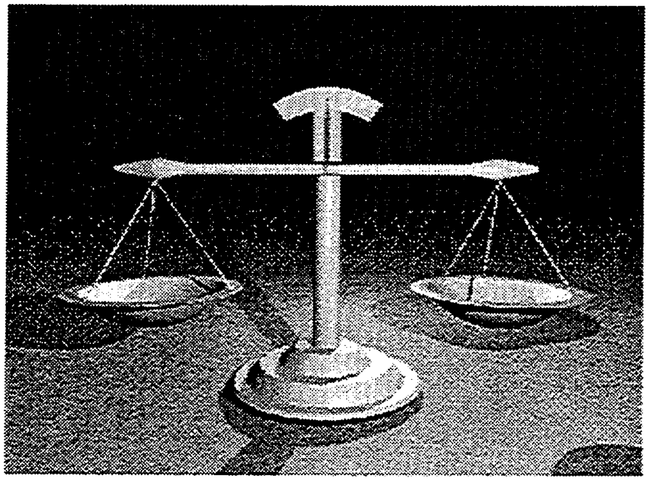
NUREG/CR-0200, Rev. 6
ORNL/NUREG/CSD-2/R6
Vol. 2, Part 2

SCALE

Version 4.4

A Modular Code System for Performing Standardized Computer Analyses for Licensing Evaluation



Functional Modules, Part 2

Oak Ridge National Laboratory

Prepared for
U.S. Nuclear Regulatory Commission

NUREG/CR-0200, Rev. 6
ORNL/NUREG/CSD-2/R6
Vol. 2, Part 2

SCALE: A Modular Code System for Performing Standardized Computer Analyses for Licensing Evaluation

Functional Modules F9 – F11

Manuscript Completed: February 2000
Date Published: May 2000

Oak Ridge National Laboratory
Managed by Lockheed Martin Energy Research Corp.
Oak Ridge, TN 37831

Prepared for
Spent Fuel Project Office
Office of Nuclear Material Safety and Safeguards
U.S. Nuclear Regulatory Commission
Washington, DC 20555-0001
NRC Job Code B0009

AVAILABILITY OF REFERENCE MATERIALS IN NRC PUBLICATIONS

NRC Reference Material

As of November 1999, you may electronically access NUREG-series publications and other NRC records at NRC's Public Electronic Reading Room at www.nrc.gov/NRC/ADAMS/index.html. Publicly released records include, to name a few, NUREG-series publications; *Federal Register* notices; applicant, licensee, and vendor documents and correspondence; NRC correspondence and internal memoranda; bulletins and information notices; inspection and investigative reports; licensee event reports; and Commission papers and their attachments.

NRC publications in the NUREG series, NRC regulations, and *Title 10, Energy*, in the Code of *Federal Regulations* may also be purchased from one of these two sources.

1. The Superintendent of Documents
U.S. Government Printing Office
P. O. Box 37082
Washington, DC 20402–9328
www.access.gpo.gov/su_docs
202–512–1800
2. The National Technical Information Service
Springfield, VA 22161–0002
www.ntis.gov
1–800–533–6847 or, locally, 703–805–6000

A single copy of each NRC draft report for comment is available free, to the extent of supply, upon written request as follows:

Address: Office of the Chief Information Officer,
Reproduction and Distribution
Services Section
U.S. Nuclear Regulatory Commission
Washington, DC 20555-0001
E-mail: DISTRIBUTION@nrc.gov
Facsimile: 301–415–2289

Some publications in the NUREG series that are posted at NRC's Web site address www.nrc.gov/NRC/NUREGS/indexnum.html are updated regularly and may differ from the last printed version.

Non-NRC Reference Material

Documents available from public and special technical libraries include all open literature items, such as books, journal articles, and transactions, *Federal Register* notices, Federal and State legislation, and congressional reports. Such documents as theses, dissertations, foreign reports and translations, and non-NRC conference proceedings may be purchased from their sponsoring organization.

Copies of industry codes and standards used in a substantive manner in the NRC regulatory process are maintained at—

The NRC Technical Library
Two White Flint North
11545 Rockville Pike
Rockville, MD 20852–2738

These standards are available in the library for reference use by the public. Codes and standards are usually copyrighted and may be purchased from the originating organization or, if they are American National Standards, from—

American National Standards Institute
11 West 42nd Street
New York, NY 10036–8002
www.ansi.org
212–642–4900

The NUREG series comprises (1) technical and administrative reports and books prepared by the staff (NUREG–XXXX) or agency contractors (NUREG/CR–XXXX), (2) proceedings of conferences (NUREG/CP–XXXX), (3) reports resulting from international agreements (NUREG/IA–XXXX), (4) brochures (NUREG/BR–XXXX), and (5) compilations of legal decisions and orders of the Commission and Atomic and Safety Licensing Boards and of Directors' decisions under Section 2.206 of NRC's regulations (NUREG–0750).

DISCLAIMER: This report was prepared as an account of work sponsored by an agency of the U.S. Government. Neither the U.S. Government nor any agency thereof, nor any employee, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for any third party's use, or the results of such use, of any information, apparatus, product, or process disclosed in this publication, or represents that its use by such third party would not infringe privately owned rights.

ABSTRACT

SCALE, a modular code system for Standardized Computer Analyses Licensing Evaluation, has been developed by Oak Ridge National Laboratory at the request of the U.S. Nuclear Regulatory Commission. The SCALE system utilizes well-established computer codes and methods within standard analysis sequences that (1) allow an input format designed for the occasional user and/or novice, (2) automate the data processing and coupling between modules, and (3) provide accurate and reliable results. System development has been directed at problem-dependent cross-section processing and analysis of criticality safety, shielding, heat transfer, and depletion/decay problems. Since the initial release of SCALE in 1980, the code system has been heavily used for evaluation of nuclear fuel facility and package designs. This revision documents Version 4.4 of the system.

CONTENTS

Volume 1, Part 1: Control Modules

Volume 1, Part 2: Control Modules

*Obsolete with SCALE-4.0 release.
**Not included in SCALE-4.4 release.
***Cancelled. Included in Section C4.

Volume 2, Part 1: Functional Modules

Volume 2, Part 2: Functional Modules

*Obsolete with SCALE-4.0 release.
**Not included in SCALE-4.4 release.
***Cancelled. Included in Section C4.

Volume 2, Part 3: Functional Modules

Volume 3: Miscellaneous

*Obsolete with SCALE-4.0 release.
**Not included in SCALE-4.4 release.
***Cancelled. Included in Section C4.

Volume 3: Miscellaneous (continued)

*Obsolete with SCALE-4.0 release.
**Not included in SCALE-4.4 release.
***Cancelled. Included in Section C4.

PREFACE

Introduction

This Manual represents Revision 6 of the user documentation for the modular code system referred to as SCALE. The previous revision documented version 4.3 of SCALE, released in October 1995. This revision documents version 4.4a of SCALE. Prior to the release of version 4.4a, SCALE 4.4 was released in September 1998. Many minor corrections and enhancements have been made since that time and are being included in SCALE 4.4a. The corrections and enhancements in versions 4.4a and 4.4 are documented separately below. All modifications in version 4.4 are included in version 4.4a.

Background

The history of the SCALE code system dates back to 1969 when the current Computational Physics and Engineering Division at Oak Ridge National Laboratory (ORNL) began providing the transportation package certification staff at the U.S. Atomic Energy Commission with computational support in the use of the new KENO code for performing criticality safety assessments with the statistical Monte Carlo method. From 1969 to 1976 the certification staff relied on the ORNL staff to assist them in the correct use of codes and data for criticality, shielding, and heat transfer analyses of transportation packages. However, the certification staff learned that, with only occasional use of the codes, it was difficult to become proficient in performing the calculations often needed for an independent safety review. Thus, shortly after the move of the certification staff to the U.S. Nuclear Regulatory Commission (NRC), the NRC staff proposed the development of an easy-to-use analysis system that provided the technical capabilities of the individual modules with which they were familiar. With this proposal, the concept of the Standardized Computer Analyses for Licensing Evaluation (SCALE) code system was born.

The NRC staff provided ORNL with some general development criteria for SCALE: (1) focus on applications related to nuclear fuel facilities and package designs, (2) use well-established computer codes and data libraries, (3) design an input format for the occasional or novice user, (4) prepare "standard" analysis sequences (control modules) that will automate the use of multiple codes (functional modules) and data to perform a system analysis, and (5) provide complete documentation and public availability. With these criteria the ORNL staff laid out the framework for the SCALE system and began development efforts. The initial version (Version 0) of the SCALE Manual was published in July 1980. Then, as now, the Manual is divided into three volumes — Volume 1 for the control module documentation (Sections C4, C6, D1, S1- S5, and H1), Volume 2 for the functional module documentation (Sections F1- F17), and Volume 3 for the documentation of data libraries, and subroutine libraries, and miscellaneous utilities (Sections M1- M17).

System Overview

The original concept of SCALE was to provide "standardized" sequences where the user had very few analysis options in addition to the geometry model and materials. Input for the control modules has been designed to be free-form with extensive use of keywords and engineering-type input requirements. The more flexible functional modules have a more difficult input logic and require the user to interface the data sets necessary to run the modules in a stand-alone fashion. As the system has grown in popularity over the years and additional options have been requested, the control modules have been improved to allow sophisticated users additional access to the numerous capabilities within the functional modules. However, the most important feature of the SCALE system remains the capability to simplify the user knowledge and effort required to prepare material mixtures and to perform adequate problem-dependent cross-section processing.

The modules available in Version 0 of SCALE were for criticality safety analysis sequences (CSAS) that provided automated material and cross-section processing prior to a one-dimensional (1-D) or multidimensional criticality analysis. Since that time the capabilities of the system have been significantly expanded to provide additional CSAS capabilities, new shielding analysis sequences (SAS) that also include depletion/decay capabilities for spent fuel characterization, and a heat transfer analysis sequence (HTAS). At the center of the CSAS and SAS sequences is the library of subroutines referred to as the Material Information Processor or MIPLIB (see Section M7). The purpose of MIPLIB is to allow users to specify problem materials using easily remembered and easily recognizable keywords that are associated with mixtures, elements, and nuclides provided in the Standard Composition Library (see Section M8). MIPLIB also uses other keywords and simple geometry input specifications to prepare input for the modules that perform the problem-dependent cross-section processing: BONAMI, NITAWL-II, and XSDRNPM. A keyword supplied by the user selects the cross-section library from a standard set provided in SCALE (see Section M4) or designates the reference to a user-supplied library. Several utility modules from AMPX[1] have been included to provide users with the capability to edit the cross-section data and reformat user-supplied libraries for use in SCALE.

Over the history of the project several modules have been removed from the system because they are no longer supported by the development staff at ORNL. Tables 1 and 2 provide a summary of the major applications of each of the control modules and functional modules currently in the SCALE code system. The control modules were designed to provide the system analysis capability originally requested by the NRC staff. The CSAS module (sometimes denoted as the CSAS4 module and documented in Section C4) is the primary control module designed for the calculation of the neutron multiplication factor of a system. Eight sequences enable general analysis of a 1-D system model or a multidimensional system model, capabilities to search on geometry spacing, and problem-dependent cross-section processing for use in executing stand-alone functional modules. CSAS6 is a newer criticality control module to provide automated problem-dependent cross-section processing and criticality calculations via the KENO-VI functional module. The SAS1 and SAS3 modules (see Sections S1 and S3, respectively) provide general 1-D deterministic and 3-D Monte Carlo analysis capabilities. The SAS2 module (see Section S2) was originally developed to perform a depletion/decay calculation to obtain spent fuel radiation source terms that were subsequently input automatically to a 1-D, radial shielding analysis in a cylindrical geometry. Over time the depletion/decay portion of the SAS2 module has been significantly enhanced and interfacing to the other shielding modules has been provided. An alternative sequence for depletion/decay calculations is ORIGEN-ARP (Section D1), which interpolates pre-generated ORIGEN-S cross-section libraries versus enrichment, burnup, and moderator density. The SAS4 module (see Section S4) enables automated particle biasing for a Monte Carlo analysis of a transportation package-type geometry. The HTAS1 module (see Section H1) is the only heat transfer control module and uses the various capabilities of the HEATING code to perform different sequences of steady-state and transient analysis that enable the normal and accident conditions of a transportation package to be evaluated. Like SAS4, the HTAS1 module is limited to a package-type geometry. The capability to perform a point-kernel shielding analysis within the SCALE system has been developed in the QADS control module.

A 238-energy-group neutron cross-section library based on ENDF/B-V[2] is the latest cross-section library in SCALE. All the nuclides that are available in ENDF/B-V are in the library. A 44-group library has been collapsed from this 238-group library and validated against numerous critical measurements.[3] These libraries are available in this version of SCALE.

Technical Assistance and Updates

To obtain technical assistance regarding the installation and use of SCALE, download software updates, or report problems, you may contact us through the following channels:

- E-mail questions to **scalehelp@ornl.gov**

- The SCALE Users Electronic Notebook on the Web:
http://www-rsicc.ornl.gov/ENOTE/enotscal.html

- SCALE Web Site (including Download, Training, Benchmarks, and Newsletter pages):
http://www.cped.ornl.gov/cad_nea/text/scale-home.html

- SCALE Newsletter:
http://www.cped.ornl.gov/cad_nea/text/scale_news.html

- FAX to SCALE Help, 815-327-6460 or 865-576-3513

Significant Updates in SCALE 4.4a

A significant number of updates have been made to SCALE since the initial release of SCALE 4.4 in September 1998. Most of these updates were minor corrections or enhancements. Because some of these updates could be important to SCALE users, this interim release of SCALE 4.4a is being made available.

SAS4 and PICTURE were enhanced to allow the generation of two-dimensional (2-D) plots when the "PARM=CHECK" option is used. This option is similar to the plotting option in the CSAS criticality sequences. Another innovation was the addition of an option that allows users to specify an X-Y, X-Z, or Y-Z plot and have the code automatically calculate the cosines used for the plot.

A discrepancy in scoring boundary crossings of surface detectors was corrected in MORSE. Contributions to user-specified surface detectors in MORSE in SCALE 4.4 could have been underestimated because of a failure to determine which surface detector to score. This failure was due to the comparison of a single precision variable to a double precision variable. Most affected cases would have a zero result for the surface detector, indicating that no particles have crossed the surface detector boundary. Detector location coordinates of four digits or less would not be expected to experience this problem.

A coding error introduced in QAD-CGGP in SCALE 4.4 has been identified and corrected. Because of inconsistent array dimensions, if more than a very limited number of bodies are input in one zone, the additional zone data are lost or stored incorrectly. This situation typically causes the code to fail. Though extremely unlikely, it might be possible for a case like this to run if the incorrectly stored geometry happened to be valid. SCALE 4.4 users should check under the "input zone data" header in the QAD-CGGP output to verify that the zone data agree with their input.

SAS2 was corrected to fix an error introduced in SCALE 4.4 that caused the PARM=OLDSAS2 option to fail. Another discrepancy introduced in SCALE 4.4 caused spent fuel isotopic data written to file FT72F001 to be incorrect in certain cases. This error, which has been corrected, occurred in cases where burnable poison rods or other inserts are removed from or inserted into the fuel assembly between fuel cycles. SAS2 can now correctly handle multiple fuel zones in the path B model. A minor discrepancy was corrected where invalid characters were being written to title records in the ORIGEN-S binary library. Some text editors could not read the SAS2 output file when invalid characters were present.

A large number of enhancements were made to XSDRNPM. The Fortran source for XSDRNPM was converted to Fortran 90 free format. The input/output units were all moved to the 0$ array. The energy of the average lethargy causing fission was added to the balance tables. The output files from the balance tables and the activities were modified and converted to ASCII files. A new ASCII file was created that contains the input and derived data from a problem. The coarse mesh generation algorithm used in rebalancing the inner iterations was modified to correct a problem that prevented convergence for a very small class of problems. The code was modified to recycle if the final iteration performed after convergence failed the convergence test. For group banding cases, convergence is now reset after initial convergence to an order of magnitude less than overall convergence to prevent looping through iterations and never converging. The default value for flux convergence tolerance, PTC, was reduced from 10^{-4} to 10^{-5}. The calculation of activities by interval, an option that was available many years ago, was reintroduced in the code.

The XSDRNPM mesh generation algorithm in MIPLIB was modified to address two problems: (a) insufficient number of mesh intervals for thick reflectors of low absorbing material and (b) too many mesh intervals for highly absorbing regions. New input options to override the automatic mesh generation were added. Although this enhancement was designed primarily for CSAS1X, it potentially affects all control modules except SAS2H that use XSDRNPM.

KENO-VI was modified to detect intersecting HOLEs in the global unit. A problem will now terminate if intersecting HOLEs are detected in the global unit. Intersecting HOLEs are illegal in KENO-VI geometry but were not detected in the global unit in SCALE 4.4. Intersecting HOLEs in units other than the global unit are detected during tracking of particles through the intersecting regions. Several corrections were made to KENO-VI to prevent a particle from becoming lost and causing the code to enter an infinite loop.

MORSE was updated to correct a problem in determining the correct day of the week for dates after December 31, 1999.

Many other minor changes included in SCALE 4.4a are listed under "SCALE 4.4a Minor Modifications."

SCALE 4.4a Minor Modifications

In addition to the major enhancements noted above, SCALE 4.4a contains many minor modifications, including corrections to errors in SCALE 4.4 and changes to improve portability to different computing platforms. Note that some of these modifications may be duplicate listings of items mentioned in the previous section.

PICTURE **MRR98-056**

Updated to handle the call by SAS4 when the "PARM=CHECK" option (added to the SAS4 control module in MRR98-057) is used. Also, added an option that allows users to specify an X-Y, X-Z, or Y-Z plot and have the code automatically calculate the cosines used for the plot.

SAS4 **MRR98-057**

Added a "PARM=CHECK" option that calls PICTURE from within the SAS4 to plot geometry but not run MORSE. SAS4 prepares or reads MARS geometry input data, reads PICTURE input; and calls PICTURE. Several other changes were made to error messages and formats.

SAS2 **MRR98-058**

Updated to correct an error introduced in SCALE 4.4 that caused the PARM=OLDSAS2 option to fail. Also corrected another problem introduced in SCALE 4.4 that caused spent fuel isotopic data written to file FT72F001 to be incorrect in certain cases where burnable poison rods or other inserts are removed from or inserted into the fuel assembly between fuel cycles.

MORSE **MRR98-059**

Corrected a discrepancy in scoring boundary crossings of surface detectors. A roundoff error caused by comparison of a double precision variable to a single precision constant resulted in boundary crossings not being scored. The epsilon value for the comparison was also increased from 0.0005 to 0.001.

SAS4 **MRR98-060**

Updated to correct the dimensions on two arrays. Also changed a test comparing 2 floating point variable names equivalenced to integer variables to use function ISET. (This test has previously caused floating point underflows on some platforms.)

QADS **MRR98-061**

Added a test on the MIPLIB error flag that terminates execution of the problem if an error occurred.

KENO-VI **MRR98-062**

Updated to allow a particle to cross from one hole directly into an adjacent hole even if the crossing is outside the allowed tolerances. This prevents some cases from entering an infinite loop.

SAS2 **MRR98-063**

Corrected minor discrepancy that resulted in invalid characters being written to title records in ORIGEN-S binary library. Some text editors could not read SAS2 output file when invalid characters were present.

MIPLIB **MRR98-064**

Updated to allow control modules to specify a sensitivity library from NITAWL and to allow number density input for an element that has multiple isotopes.

XSDRNPM **MRR98-065**

The Fortran source for XSDRNPM was converted to Fortran 90 free format. The input-output units were all moved to the 0$ array. The energy of the average lethargy causing fission was added to the balance tables. The output files from the balance tables and the activities were modified and converted to ASCII files. A new file was created which contains the input and derived data from a problem. The flux file was changed to double precision. The code was modified to not run with fluxes out of core unless explicitly requested in the input. The coarse mesh generation algorithm used in rebalancing the inner iterations was modified to correct a problem that prevented convergence a very small class of problems.

C5TOC6/K5TOK6 **MRR98-066/MRR98-067**

The input file generated for CSAS6/KENO VI incorrectly labeled regions generated to surround HOLEs if there were more than one region in a unit that contained HOLEs. Because of a change in KENO-VI, these regions should no longer need to be generated. Subroutine PUNCH_GEOM was modified to not generate these regions.

QAD-CGGP **MRR98-068**

Updated to correct an error introduced in SCALE 4.4. The dimension on one variable in the geometry was not updated when the input format was changed to match that of MARS input. This caused some jobs to fail. Also, updated to correct misspelled name of unit used for error output.

ARPLIB **MRR99-001**

Updated to accept either lower or upper case input.

PRISM **MRR99-002**
Updated to accept either lower or upper case input.

XSECLIST **MRR99-003**
Updated to accept either lower or upper case input.

SAS2 **MRR99-004**
Updated for compatibility with the newest revisions to XSDRNPM (MRR98-065). The routines that wrote the XSDRNPM input files needed to be changed to account for the changes to XSDRNPM input. Subroutine COPYNX had to be changed to add the 0$ array to the XSDRNPM input file, and to move setting the logical unit number of the flux output file from the 2$ array to the 0$ array.

UNIXLIB **MRR99-006**
Changes to update XSDRNPM required a double precision ERF function. This function is part of the Fortran intrinsic library for the DEC Alpha's and the IBM RS/6000, but is not part of that library for the HP or the SUN workstations. This modification provided an update for the necessary routines to compute the double precision ERF when it is not part of the intrinsic library.

BONAMI **MRR99-007**
Updated to correct a problem that caused cases to fail when zero number density input is used.

SAS4 **MRR99-008**
Updated to change the convergence criteria because the criteria in XSDRNPM were changed. Also added an input variable NDAB to allow the user to specify the number of direct access blocks allocated.

XSDOSE **MRR99-009**
Added option to turn off angular flux print and made the default to be no angular flux print.

MODIFY **MRR99-010**
Updated subroutine LODATA for compatibility with changes in MRR98-064(MIPLIB).

XSDRNPM **MRR99-011**
(1) The code was modified to re-cycle if the final iteration performed after convergence failed the convergence test. For group banding cases, convergence is now reset after initial convergence to an order of magnitude less than overall convergence to prevent looping through iterations and never converting. (2) The default value for flux convergence tolerance, PTC, was reduced from 10^{-4} to 10^{-5}. (3) Errors were corrected in the calculation of activities by interval. This previously undocumented option is now documented in the XSDRNPM input description.

KENO-VI **MRR99-012**
Modified KENO-VI to detect intersecting HOLEs in the global unit. A problem will now terminate if intersecting HOLEs are detected in the global unit. Intersecting HOLEs are illegal in KENO-VI geometry but were not detected in the global unit in SCALE 4.4. Intersecting HOLEs in units other than the global unit are detected during tracking of particles through the intersecting regions.

ORIGEN **MRR99-014**
A new subroutine was added to provide the user the option of more significant digits in the output tables. Unit 71 was set as the default file number for the binary file containing concentrations and spectral data.

MIPLIB **MRR99-015**

The XSDRNPM mesh generation algorithm was modified to address two problems: (a) insufficient number of mesh intervals for thick reflectors of low absorbing material and (b) too many mesh intervals for highly absorbing regions. New input options to override the automatic mesh generation were added too. Consistent with MRR99-011, the default value of PTC was reduced from 10^{-4} to 10^{-5}. Although this enhancement was designed primarily for CSAS1X, it potentially affects all control modules that use XSDRNPM.

KENO-VI **MRR99-016**

The code was corrected to define LCHK as a logical variable in subroutine POSIT. In addition, an IMPLICIT NONE statement has been added to the beginning of the subroutine. All variables have been explicitly typed as appropriate.

MODIFY **MRR99-017**

Program MODIFY was changed for consistency with the change in the direct access file made in MRR99-015(MIPLIB).

KMART **MRR99-018**

An error that resulted in calculated volumes of zero for hemicylinders and arrays (if an array number was skipped) was corrected.

CSAS6 **MRR99-019**

The argument list for the call to subroutine PRTPLT was modified for consistency with changes made to KENO-VI in MRR99-012.

SAS2 **MRR99-020**

Calls to subroutine EPSIG were changed for consistency with MIPLIB modifications in MRR99-015.

MORSE **MRR99-021**

The code was updated to correct a problem in determining the day of the week for dates after December 31, 1999.

KENO-VI **MRR99-022**

The code was updated to fix a roundoff problem that sometimes caused particles to get into an infinite loop when they transferred from one array location to another but in the process missed the unit boundary.

SUBLIB **MRR99-023**

Subroutine YREAD turns off the normal invalid character check done by the free form reading routines, but does not make any checks of its own for invalid characters. This can lead to erroneous results in some cases where a user mistypes a character when entering the array data to KENO. Checks were added to the array reading routine to give warning messages if illegal characters are read. Corrections were also made so YREAD would store correctly a double precision array.

KENO-VI **MRR99-024**

The code was updated to fix a problem where the unit boundary shares surfaces with other geometry regions and the unit is in an array. This problem could result in an infinite loop because the code fails to detect a particle crossing the boundary.

SAS2 MRR99-025

The following modifications were made: (1) The calculation of the light element concentrations in ORIGEN-S was corrected when multiple fuel zones (MX=500) are used in the PATH B model. The code previously assumed only one fuel zone was present, and did not sum the zone volumes when multiple zones were present, resulting in erroneous light element concentrations in the ORIGEN-S depletion calculations. (2) The depletion of light element nuclides with mixture numbers 50 through 59 is now permitted. (3) The use of 1$ data for MXT (input level 3) when reading a second working library in NITAWL is now permitted.

HEATING MRR99-027

The code was updated, including Fortran 90 dynamic memory allocation, to improve portability on both workstation and PC platforms.

UNIXLIB MRR99-028

Subroutine JSTIME was modified to return time to the precision supplied by the system.

SAS4 Sample Problems DRR99-001

Updated SAS4 sample problems 1, 3, and 5 to remove references to variables FR1, FR2, FR3, and FR4, which became obsolete in SCALE 4.4.

XSDOSE Sample Problem DRR99-002

The XSDOSE section of the SCALE Manual documents the output of the sample problem and includes the printing of the fluxes. The input was modified to turn on the new angular flux print option in XSDOSE (see MRR99-009).

KENO V.a Sample Problems DRR99-003

Input data for sample problems 17 and 18 were changed. The number of neutrons started in sample problem 18 was changed to agree with the number per generation. Problem 17 was changed to specify the NBK parameter because the default was not large enough.

238- and 44-Group ENDF/B-V Libraries DRRs 99-004 and 99-005

Changes were made because problems were discovered with ^{238}Np, ^{250}Cf, ^{253}Cf, ^{249}Bk, ^{242}Am, and ^{233}Pa. The corrections for ^{250}Cf, ^{233}Pa, and ^{249}Bk were very minor and should have no important effects. However, significant errors were identified for ^{238}Np, ^{253}Cf, and ^{242}Am. In addition, these three nuclides do not have fission cross sections specified in the fast region in ENDF/B-V. This omission is obviously wrong, and because it could lead to very non-conservative answers for k_{eff}, these three nuclides were removed from the library.

Major Enhancements in SCALE 4.4

Many enhancements and corrections were made to SCALE in the three years between the release of SCALE-4.3 and 4.4. SCALE 4.4 is compatible with the year 2000 (see "SCALE 4.4 is Year 2000 Compliant"). User-specified surface detectors have been added to SAS4/MORSE to improve its computational flexibility and efficiency (see "Improvements to SAS4 and MORSE"). The KENO-VI input requirements for HOLEs have been simplified and made more consistent with KENO V.a (see "KENO-VI HOLE Input is Simplified"). Additionally, some significant improvements to the speed and stability of KENO-VI have been made (see "KENO-VI Stability and Speed Improvements"). A large number of changes have been made to the SAS2H depletion module (see "SAS2H Corrections and Enhancements").

Several enhancements have been made to the PC version of SCALE 4.4. A significant effort has been made to minimize the programming differences between the PC and Unix workstation versions. Both versions will contain the same modules. The heat transfer modules HTAS1, HEATING, and the HEATING auxiliary codes are now available in the PC version for the first time. The PC version can recognize MS-DOS, Windows 95, Windows 98, and Windows NT operating systems and run under any of these systems from a single user command. CSAS can now be run directly from the CSASIN input processor.

ORIGEN-ARP, which was first released in the PC version of SCALE-4.3, has been enhanced and now runs under the SCALE driver, so it can run easily on workstations as well as PCs. ORIGEN-ARP has been improved significantly. ARP now interpolates on moderator density as well as burnup and enrichment for BWR fuel types. Several auxiliary codes have been added that enable users to generate their own ORIGEN-ARP cross-section libraries via SAS2H.

The default number of histories in KENO V.a and KENO-VI have been increased to 200,000 to produce more statistically accurate results. Color plots are now the defaults in both these codes.

PICTURE has been upgraded to generate two-dimensional (2-D) color plots of MORSE/MARS and QADS/QAD-CGGP geometry models like the color plots generated by KENO V.a and KENO-VI in SCALE-4.3. A new utility, LEGEND, has been created that adds a color/material legend and title to the color plots generated by KENO and PICTURE.

KMART is a new module to allow post-processing of a KENO V.a restart file, along with a working format cross-section library, to generate activities and/or broad-group fluxes and to compute the fission production activity if the components are available in the working cross-section library for the requested nuclide.

The group banding procedure in XSDRNPM was modified to significantly improve convergence for many large problems. Two examples of improvement include a fixed-source calculation with an 85% reduction in run-time and a k_{eff} calculation with a 50% reduction in run-time.

A correction was made to MIPLIB to allow the use of moderator in the gap region of a lattice cell calculation. Prior to this correction, if the same mixture number was specified in the moderator and the gap regions, the moderator density was incorrectly increased by a factor of two in the Dancoff factor calculation. In CSAS or CSAS6, this error results in a non-conservative calculated k_{eff} value that is approximately 0.5 to 1% low.

Other additions to SCALE 4.4 include the capability to perform a one-dimensional criticality search in CSAS1X (see "Criticality Search in CSAS1X"); the new KENO biasing weights library for 16-, 27-, 44-, 218-, and 238-group problems (see "New KENO Weights Library and Modules to Generate Weights"); the C5TOC6 and K5TOK6 conversion utilities for KENO-VI, and the Q0RDPN binary to ASCII conversion utility for functional module FIDO input files (see "New SCALE Utility Programs"); and the new zirconium hydride cross section data in the 238- and 44-group ENDF/B-V libraries (see "Zirconium Hydride Cross Sections").

The SCALE manual is distributed in electronic format on CD with the software. The manual is formatted in PDF files that can be read, searched, and printed using Adobe Acrobat Reader with Search. Users who desire a hard copy of the manual may obtain one from RSICC for an additional charge to cover reproduction costs.

Many other minor changes included in SCALE 4.4 are listed under "SCALE 4.4 Minor Modifications."

SCALE 4.4 is Year 2000 Compliant

Current and earlier versions of SCALE should calculate results correctly beyond the year 2000. However, when the year 2000 occurs, the output from some codes in these earlier versions will incorrectly display the year as 1900 instead of 2000. All known instances of this problem have been corrected in SCALE 4.4.

Improvements to SAS4 and MORSE

SAS4 and MORSE have been enhanced to allow users to specify multiple non-overlapping surface detectors on each surface (previously defaulted to 4 locations). These surface detectors can be divided into "sub-detectors" that enable the user to obtain detailed dose rate profiles. The flexibility in the use of these surface detectors makes them suitable for the substitution of point detectors, which are much less computationally efficient. Another enhancement to SAS4 was the addition of two options to pass data to PICTURE for plotting. One option generates geometry data only for the purpose of running PICTURE to view 2-D slices of the geometry. The other option provides "PARM=CHECK" option that calls PICTURE from within the SAS4 to plot geometry but not run MORSE. SAS4 prepares or reads MARS geometry input data, reads PICTURE input, and calls PICTURE.

Improvements to MORSE include orderly termination of a problem when errors in tracking to detector exceed a limit, an option to print/not print flux output after each batch, user capability to specify the number of direct-access blocks allocated on scratch units, compatibility with the year 2000, and reduction of the amount of error output in some cases.

KENO-VI HOLE Input Is Simplified

HOLE input in KENO-VI has been simplified. These changes are significant improvements requested by many users. The HOLE boundary no longer needs to be specified in the unit containing the HOLE. The HOLE boundary is automatically added by the program based on the unit specified in the HOLE record and its ORIGIN and ROTATE data. HOLEs cannot intersect. An example of the original and the new methods for adding HOLEs to a unit is given below. The input data no longer required are highlighted in the old input.

```
********** Old KENO-VI input *************
unit 1
hexprism  10  1.0  10.0 -10.0
media  1  10
boundary  10
unit 2
cuboid    10  6p20.0
hexprism  20  1.0  10.0  -10.0  origin x=5.0 y=3.0  rotate a2=90
media 2  10  -20
hole  1  20 origin x=5.0 y=3.0 rotate a2=90
boundary  10
******** New KENO-VI input ***************
unit 1
hexprism  10  1.0  10.0 -10.0
media  1  10
boundary  10
unit 2
cuboid    10  6p20.0
media  2  10
hole   1  origin x=5.0 y=3.0 rotate a2=90
boundary  10
```

Note that in addition to the lack of a geometry record which defines the HOLE boundary, the HOLE record no longer has a vector definition array. The new version of KENO-VI should be able to read most old input files correctly, but they will take longer to run.

KENO-VI Stability and Speed Improvements

Improvements have been made to KENO-VI since the last Web update to increase the stability and the speed of KENO-VI. To improve the code's stability, logic has been added to KENO-VI to check if a particle is still in the boundary region of a unit when it is no longer in any region. If this occurs, an error message is printed and the program terminates. This situation is often caused by an undefined volume in a unit and could previously lead to the program entering an infinite loop.

To improve the execution speed of KENO-VI, the following modification has been made: When a particle is in an array, the particle is tracked both in the unit where it is currently within the array and in the unit containing the array. It needs to be tracked in the unit containing the array so it knows when it crosses out of the array. Previously, the crossing distance to every surface in that unit was calculated. The code has been changed to calculate only the crossing distance to the surfaces related to the array boundary. This change will significantly reduce the running time of problems where particles spend most of the time in an array or where the array is in a complex unit containing many additional regions unrelated to the array boundary. Running times have been reduced by as much as 15% for arrays contained in complex units.

SAS2H Corrections and Enhancements

A large number of corrections and enhancements have been completed in SAS2H for the release of SCALE 4.4. They are listed below.

- Two errors were corrected for cases where there were more than three zones prior to the mixture 500 zone in the Path B model: (1) The atomic densities were not updated with depleted values in the cross-section processing/spectrum calculations when fuel was input to more than one zone (including the cell-weighted mixture 500 zone). (2) Nuclides that only appear in the moderator were depleted. An example of a model that would be affected is a BWR Path B model with Gd-poisoned fuel pin, gap, clad, moderator, and mixture 500. Usually these discrepancies cause only slight errors in the neutronics part of such BWR cases, but could significantly impact the results for some unique fuel models.

- Input checks and error messages were improved.

- A programming error that caused problems with "MXREPEATS=0" cases to fail on PCs was corrected. These cases are typically used to remove or insert burnable poison rods from one cycle to the next in a depletion.

- A modification was made to correctly calculate the fuel bundle area printed in the shipping cask geometry for the triangular-pitch lattice type of fuel.

- The FUELBNDL input parameter was changed from integer to floating point to allow fractions of fuel assemblies.

- The temperatures of the zones (except the gap) in the Path A model may now be changed for each cycle, similar to the BFRAC and H2OFRAC variables.

- The limit on the total number of libraries (NCYC*NLIB/CYC) was increased to 9,999. However, because the number of unique output file names in SCALE is currently limited to 10,000 and there are typically 11 output files per pass in SAS2H, the practical limit for users is approximately 900 total libraries.

- A significant change was implemented to enable fixed sources (volumetric source or angular flux at a boundary) to be used with INPUTLEVEL=3 cases. This change gives users the capability to model cases such as the irradiation of target materials without explicitly including the irradiation facility in the SAS2H model. Previously the driver geometry and its specific power were required as input to govern the depletion calculation. This fixed-source option is specified in the INPUTLEVEL=3 data as either a volumetric or boundary source. SAS2H then determines the flux based on this fixed source and passes it to ORIGEN-S for use in a flux-driven depletion calculation.

- SAS2H was modified to allow two zones in the Path A model to contain the same nuclide, one at a density of 10^{-20} and the other at a density of greater than 10^{-10}.

- The fixed dimension of 1000 for the Path B mixing table arrays was removed where possible and increased otherwise. The size needed for these arrays can be as large as five times the Path A mixing table size (currently a maximum of about 300 in the 44-group library) plus the number of nuclides outside the zone of mixture 500. The dimension of the arrays that remain fixed was increased to 5000. The remaining arrays were variably dimensioned to the maximum of 2000 or the sum of the Path B mixing table size plus 100 (to allow increases of at least 100 nuclides for INPUTLEVEL=3).

Criticality Search in CSAS1X

MIPLIB has been updated to add input options to MORE DATA that allow specifying an XSDRN adjoint solution, a zone width search, a unit number for the balance table file, and suppressing the cross section weighting. The addition of the zone width search option now gives CSAS1X the capability to perform one-dimensional criticality searches on the size of a geometry zone in XSDRNPM.

New KENO Weights Library and Modules to Generate Weights

Because there was a need to be able to automatically generate a set of weights for use in KENO for arbitrary group structure and material, a new control module GWAS and a new functional module GENWGTS have been added. GWAS sets up an adjoint XSDRNPM case and generates weights automatically from the fluxes. GENWGTS is called by GWAS to read the adjoint fluxes, automatically generate the KENO weighting functions from them, and write an output file for use by program WGT. The biasing weights library for KENO V.a and KENO-VI was updated using the new modules GWAS and GENWGTS. The library contains weights for paraffin, water, concrete, and graphite in 16, 27, 44, 218, and 238 energy groups. The new library was created because there were no biasing data for use with the new ENDF/B-V 44- and 238-group libraries that were released in SCALE-4.3. The old library only contained data for 16, 27, and 123 groups. Note that the 123-group library was removed in SCALE-4.3. Results using this new weights library with the 16- and 27-group cross-section libraries will be different but should agree within statistical uncertainty.

New SCALE Utility Programs

Several new utility programs have been developed for SCALE. A new utility LEGEND has been created that adds a title and legend to the color GIF files generated by KENO V.a or KENO-VI. LEGEND was released last summer with the updated version of KENO-VI (see the June 1996 issue of the Newsletter). The versions of KENO V.a and PICTURE in the next release of SCALE will use LEGEND as well.

K5TOK6 and C5TOC6 are new utilities that convert KENO V.a and CSAS input files to KENO-VI and CSAS6 input files by translating the KENO V.a geometry input to KENO-VI format. Since the converted input files are based on the KENO V.a geometry input, they are generally not the most effective in terms of the

KENO-VI geometry features. They do provide the user with a working KENO-VI input file that can be modified for improvements.

Another new utility is Q0RDPN. It converts a binary input file generated by a CSAS or SAS control sequence for one of the functional modules that use FIDO input such as BONAMI, NITAWL-II, ICE, and XSDRNPM, to an ASCII input file. The user can easily edit the ASCII input file to run a modified version of a problem. This capability allows the user to specify input parameters that are not available in the standard control sequences.

Zirconium Hydride Cross Sections

The ENDF/B-V cross-section libraries in SCALE 4.4 have been updated with thermal scattering data for zirconium hydride. New standard compositions have been added to the Standard Composition Library to allow access to these new cross sections. The new standard compositions are the following:

ZRH2 - density 5.61 g/cc, 1 zirconium to 2 hydrogen atoms
ZR5H8 - density 5.61 g/cc, 5 zirconium to 8 hydrogen atoms
H-ZRH2 - density 1.0 g/cc, the hydrogen in zirconium hydride
ZR-ZRH2 - density 1.0 g/cc, the zirconium in zirconium hydride

SCALE 4.4 Minor Modifications

In addition to the major enhancements noted above, SCALE 4.4 contains many minor modifications, including corrections to errors in SCALE-4.3 and changes to improve portability to different computing platforms. Note that some of these modifications may be duplicate listings of items mentioned in the previous sections.

18-Group Gamma Library: (1) Processed through CORECTOL to mark it as NITAWL-II compatible. Could not be processed by NITAWL-II prior to this correction. (2) Updated to replace the Henderson and Claiborne-Trubey dose factors because the data overestimated the doses by about 25%. The replacement data were taken from the 22n-18g group coupled library.

27-Group Burnup Library: Updated data on rhodium-103 so that Bondarenko factors are generated in the unresolved resonance range. A test case based on 4.5 wt % UO_2 burned to 54,585 MWD/MTU, cooled for 5 years, was run. The calculated k_{eff} increased by 0.06% with the new Rh-103 cross sections.

44-Group ENDF/B-V Library: The 44-group neutron cross-section library was recollapsed from the 238-group library using the corrected version of MALOCS. The impact of the MALOCS corrections should be negligible. See MALOCS corrections below for more information.

238-Group and 44-Group ENDF/B-V Libraries: (1) Corrected negative scattering and total cross sections for minor actinides, fission products, and beryllium metal. Also corrected thermal Bondarenko factors for potassium. Only significant impact should be on cases where potassium is important in the thermal range. (2) Updated to remove resonance parameters from specially weighted stainless steel nuclides because they were being doubly applied. Also, zirconium and hydrogen cross sections for zirconium hydride were added to both libraries.

AJAX: Corrected a portability problem in subroutine ANN caused by the array D being typed real by default, and then printing variables from it using an integer format.

ARP: Updated for optional interpolation on moderator density and made more general to handle user-created basic cross-section libraries. ARP now runs under SCALE driver on PCs and workstations.

ARPLIB: This is a new utility program that creates binary ORIGEN libraries for ARP. It extracts libraries at the desired burnups from large multi-burnup library files generated by SAS2H.

AWL: Added AWL to SCALE to convert AMPX working format libraries between ASCII and binary formats. It is required for the SCALE Criticality V&V package.

BONAMI: (1) Updated to improve error handling procedure and messages. (2) Corrected a problem that caused cases to fail when zero number density input was used.

C5TOC6/K5TOK6: The input file generated for CSAS6/KENO VI incorrectly labeled regions generated to surround HOLEs if there were more than one region in a unit that contained HOLEs. Because of a change in KENO-VI, these regions should no longer need to be generated. Subroutine PUNCH_GEOM was modified to no longer generate these regions.

COUPLE: Updated for year 2000 compatibility, PC version compatibility, uppercase or lowercase input files, and for printing the banner page only when COUPLE is first called.

COUPLE Sample Problem: Updated to change the inner radii in the 3$$ array to zero for consistency with the NITAWL-II input requirements.

CSAS/KENO V.a /KENO-VI/SAS2H Sample Problems: Updated to use the 44-group library.

CSAS and MODIFY: CSAS was updated to add additional required data to the direct access file written for a search problem. MODIFY was updated to read this file. A check for valid parameter constraints and the printing of an error message if they are invalid were also added.

H7MAP: For 1-D problems, if the number of nodes is large enough that the output exceeds one page in length, only part of the output is displayed. The output from the first page is repeated, and the rest of the output is never printed. Correcting this problem involved simply moving one statement from within a DO loop to a point before the DO loop.

H7TECPLOT and H7MONITOR: Outdated comment lines in the BLOCK DATA subroutine that are used to activate or deactivate computer-system-dependent blocks of code resulted in memory not being allocated for variably-dimensioned arrays. An additional correction was made in H7TECPLOT, where the x and y axes were reversed when a translation was done from spherical to Cartesian coordinates.

HEATING Sample Problems: The input file for the second HEATING sample problem was modified to first compile and run a simple Fortran program to convert an ASCII node connector file to binary format for use by HEATING. This modification improves installation portability on different Unix workstation platforms.

KENO V.a: (1) Updated subroutine RDPLOT to correct the format used to print the error message for incomplete input data. (2) Corrected variable type in format statements for debug prints. This discrepancy causes problems on some systems, including PCs when debug print is turned on (DBG=YES). (3) Changed default plot type to color. (4) Updated to correct an error in the k_{eff} calculation that caused a doubling of k_{eff}

when using an ICE mixed AMPX format working library. This error was introduced in SCALE-4.3. (5) Updated to allow printing the frequency distributions for 1-group problems. (6) Updated to match KENO-VI with respect to matrix calculations. The calculation of lifetime was corrected because it was not based on a fair game. These changes can cause the lifetime to be substantially different. The error in the lifetime calculation has probably been in KENO V.a since its initial release in SCALE-3.

KENO-VI: (1) Updated to correctly number error messages, replace the word PICTURE with the word PLOT throughout the program, and print plot symbol data only for character plots. (2) Updated subroutine TRACK to correctly sum fluxes. The fluxes didn't sum properly for units that were crossed by an array boundary. (3) Enhanced to allow HOLEs to be used without explicitly defining a geometry region where the HOLE was to be inserted. The code automatically adds to the unit containing the HOLE the equations that define the boundary of the unit contained within the HOLE, properly rotated and translated as specified on the HOLE record. (4) Fixed problem writing restart file on Sun workstation. (5) Modified the subroutine GEOMIN to correct an infinite loop problem. A pointer to the array that contained the unit boundary x, y, and z position was improperly specified. The pointer LBOXGM has been respecified. (6) Corrected a problem where a particle's inability to cross an array boundary due to round-off problems caused an infinite loop. (7) The code was updated to correct a problem that could cause cases containing arrays with complex boundaries to incorrectly calculate k_{eff}. (8) Corrected a discrepancy that caused the code to go into an infinite loop when boundaries consisted of a body with multiple sets of paired planes. (9) Corrected an error that prevented a restart problem from producing a readable file if it stored data in the generation before the code entered the infinite loop. (10) Corrected a problem involving nested arrays and hexprisms that sometimes caused the code to go into an infinite loop if a collision occurred very near a boundary. (11) Corrected a problem that occurred when a particle crossed a boundary and immediately had a collision that reversed its direction without traveling any distance. The particle sometimes got lost and entered an infinite loop. (12) Modified subroutine TRACK to correct a problem that occurred when an array shared a boundary with a hole that contained the array. If the distance to cross out of the array is less than EPS, the particle now exits the array instead of crossing from one unit to another within the array. (13) Corrected an error in placement of starting points for start type 6. (14) Corrected an error in the flux calculation for regions containing holes or arrays. (15) Corrected a roundoff problem with arrays offset a long distance from the origin. This problem could sometimes cause an infinite loop. (16) Corrected a variable that was misnamed and, as a result, was used without being initialized. (17) Set a lower limit for the calculated crossing tolerance to prevent the code from entering an infinite loop. Also made minor changes to the particle-tracking output when parameter TRK=YES. (18) Updated to allow starting points in a volume larger than the global unit. (19) Updated to terminate a problem if a particle in subroutine TRACK gets lost. Also, updated to allow problems that contain array data but do not reference the arrays in the GEOMETRY data block to run. (20) Updated to change the logic in calculating the array boundary crossing distance (decreases running time for some problems) and to change the default plot type to color. (21) Updated to correct a problem with non-cuboidal albedo boundaries and to add additional space for matrix data. The standard deviations for average k-effective by generation skipped are now accumulated in batches. Because of these changes, any matrix information and the table of average k-effective by generation skipped in the sample problem output will be different. (22) Corrected tracking to allow simultaneous crossing of multiple shared boundaries and to correctly sum fluxes after a collision. Also corrected error related to calculating the x-offset of an array. Changed input logic for ORIGIN and ROTATE data to sum values for an auxiliary keyword for a given geometry record rather than use the last value. This last change was made for compatibility with C5TOC6. (23) Updated to allow a particle to cross from one hole directly into an adjacent hole even if the crossing is outside the allowed tolerances. This prevents some cases from entering an infinite loop. (24) Modified to detect intersecting HOLEs in the global unit. A problem will now terminate if intersecting HOLEs are detected in the global unit. Intersecting HOLEs are illegal in KENO-VI geometry but were not detected in the global unit. Intersecting HOLEs in units other than the global unit are detected during tracking of particles through the intersecting regions.

KENO-VI Sample Problems: Sample problem 22 has been altered in the KENO-VI input file. The geometry data were changed to take advantage of the simplified method of adding HOLEs.

KMART: This new module was added to allow post processing of a KENO V.a restart file, along with a working format cross-section library, to generate activities and/or broad group fluxes and compute the fission production activity if the components are available in the working cross-section library for the requested nuclide. A resonance self-shielded value is used for the fission cross section.

MALOCS: (1) An error was corrected in weighting a coupled master library using a neutron spectrum from a neutron library combined with an explicitly specified gamma-ray spectrum. Also introduced several options for truncating upscattering terms. Changes were made to properly weight the delayed and prompt values of ν. (2) A discrepancy was corrected that caused the storage of invalid data in the temperature array. In the 44-group library this caused the data for the third temperature to be overwritten and to be used for a temperature that is effectively zero degrees Kelvin.

MARSLIB: (1) Updated to change the value of epsilon used to check for round-off errors in the geometry and, thereby, reduce the number of such errors. This modification eliminated the errors previously experienced with several of the SCALE Shielding V&V problems. (2) Variables IR in subroutine AZIP and IRET in subroutine UNIS are now initialized to 0 before they are used as arguments to function IREAD. In AZIP and in UNIS a 'CALL EXIT' was changed to a 'STOP'. In subroutine ALBERT, the nH was removed from two formats and replaced with quotes.

MIPLIB: (1) Updated to allow moderator mixture in a lattice cell to be used in the gap and to add the ability to specify the inner radius to the resonance data. (2) Updated to allow a control program to suppress certain output by setting flags. Added input options to MORE DATA to allow specifying an XSDRNPM adjoint solution, a criticality search in XSDRNPM using the zone width search option, a unit number for the balance table file, and suppressing the cross section weighting. (3) Corrected an error allowing the input of a number density for a compound or alloy. This error was introduced in SCALE-4.3. (4) Updated to allow number density input for an element that has multiple isotopes.

MIPLIB, SUBLIB, UNIXLIB, COMPOZ, MODIFY: Updated to use new direct access routines for character data and replaced references to specific intrinsic FORTRAN functions with their generic names for Fortran 90 compatibility. Also corrected an error in the Dancoff factor calculation that occurs for cylinders in a MULTIREGION problem. This error results in an error in the calculated k_{eff} value of approximately 0.1% for a cylinder the size of a typical fuel rod. Note that this error did not occur in the LATTICECELL geometry option.

MORSE: (1) Updated the limit on number of tracking errors, the unit number for surface detector results, and increased dimensions on surface detector arrays. (2) Updated to include changes to surface detectors for SAS4 cases, to correct a problem in DIREC for NDSG=17 case, to allow orderly termination of a problem when errors in tracking to detector exceed a limit, to add an option to print/not print flux output after each batch, to input the number of direct-access blocks allocated on scratch units, to change the way date is output (to handle the year 2000 and beyond) and to reduce the amount of error output in some cases.

MORSE Sample Problem 8: The 10** array was modified by adding a 22r0.0 at the end.

NITAWL: Corrected the potential cross section used for higher order resonances (L>0). The impact should be negligible in most cases.

ORIGEN-S: (1) Updated cross-section edit of binary libraries to add option to change cross-section values to quantities derived from total flux (as in ORIGEN2) instead of thermal flux. (2) Corrected calculation of printed average power. (3) Added error message if number of time steps is less than 4 for reactor startup case. (4) Updated to correct the loop index for re-normalizing the R8 array. (5) Updated for year 2000 compatibility and to correct calculation of He-3 and H-3 for long time steps and high flux. (6) Updated to allow saving concentrations and then continuing with a subcase using a new library. (7) Updated to allow the flux input value for the last time step to be zero.

ORIGEN-S Master Photon Library: The library was updated to correct the photon yield data for Ra-222 and Th-226, and the photon yields for gammas accompanying (α,n) and spontaneous fission reactions were updated to reflect small changes that occurred during the last decay data update.

OSBICO/OSBIRE: Updated for compatibility with lastest version of ORIGEN-S.

PERFUME: Improved the selection of new moments when a moment is found to be invalid and converted coding to a more standard Fortran 90.

PERFUME Sample Problem: The special cross-section data file required for the PERFUME sample problem has been added to SCALE, and the sample problem input data have been updated to use it. This problem has not been included in SCALE since SCALE was moved from the mainframe to the workstation several years ago.

PICTURE: (1) The module was updated to add option of generating 2-D color GIF plot files of the geometry model input for the SCALE shielding modules MORSE or QAD-CGGP. This capability already exists in the SCALE criticality modules KENO V.a and KENO-VI. (2) The module was also updated to handle the call by SAS4 when the "PARM=CHECK" option is used. (3) An option was added that allows users to specify an X-Y, X-Z, or Y-Z plot and have the code automatically calculate the cosines used for the plot.

PRISM: This is a new utility program for ARP that can read a single SAS2H or other type of input file and generate multiple copies by replacing generic symbols with specified values.

QADS/QAD-CGGP: (1) Updated to make the combinatorial geometry input data have the same format as the combinatorial portion of the MARS geometry input which is used in other SCALE modules. Old input files will no longer run. (2) Updated to add error checks for limits on number of compositions and elements and to fix the code to handle upper- or lower-case input. (3) Added a test on the MIPLIB error flag that terminates execution of the problem if an error occurred.

QADS and QAD-CGGP Sample Problems: Updated to change the geometry input format to agree with the changes made to QADS and QAD-CGGP.

RADE: Corrected an error in subroutine MCHEK that caused RADE to fail on a Sun workstation. A constant was passed as an argument to subroutine MCHEK to be used for dimensioning, but MCHEK later used the same variable for other purposes. The argument was renamed and used in the dimension statement.

SAS1: Scratch unit N16 was not opened when SCALE driver returned to SAS1 after cross-section processing and prior to XSDRNPM shielding calculation. This problem caused SAS1 to fail on the PC. The OPEN statement was moved to the beginning of main program so it would always be opened.

SAS2H: (1) Updated to fix a problem where the reload feature failed to reload correctly for the final cycle type. (2) Modified subroutine SZNSEG so that it would not cause the ORIGEN library creation to fail by not recognizing the cross-section library specified. The problem was an uninitialized variable ERSET. The change was to initialize the variable as "FALSE" before calling subroutine GETLIB. A change was also made so that the library name was passed to GETLIB instead of only the first 4 characters. (3) Updated to correct an error in the mass of the clad when the clad was input as an isotope and the mass was not input as a light element in Data Block 15. (4) A problem was corrected where the atomic densities were not updated with depleted values in the cross-section processing/spectrum calculations when fuel was input to more than one zone (including the cell-weighted mixture 500 zone) and there were more than three zones prior to the mixture 500 zone. For example, consider a BWR Path B model with Gd-poisoned fuel pin, gap, clad, moderator, and mixture 500, where there are four zones prior to the mixture 500 zone. Usually this discrepancy causes only slight errors in the neutronics part of such BWR cases, but could significantly impact the results for some unique fuel models. (5) The module was updated for compatibility with the newest revisions to XSDRNPM. The routines that wrote the XSDRNPM input files needed to be changed to account for the changes to XSDRNPM input. Subroutine COPYNX had to be changed to add the 0$ array to the XSDRNPM input file, and to move setting the logical unit number of the flux output file from the 2$ array to the 0$ array.

SAS3: (1) Variable IR in subroutine OAKTRE is now initialized to 0 before it is used as an argument to function AREAD. Subroutine RINPUP was updated to initialize the variables JMK and IML in COMMON JOMK because they are used when SAS3 calls MARSLIB routines and they were not being defined prior to the calls to JOMIN. (2) Updated to be compatible with the new MORSE input options and to implement the PARM=SIZE parameter which was not being passed to MORSE.

SAS4: (1) Subroutine MORINP was updated to add common JOMK and to initialize the variables JMK and IML in common JOMK because they are used when SAS4 calls MARSLIB routines and they were not being defined prior to the calls to JOMIN. (2) The code was modified to translate the user input to lowercase. This change was necessary to make SAS4 capable of handling input files in either upper or lowercase, as the other SCALE neutronic codes already do. (3) The code was updated to correct the dimensions on two arrays. Also changed a test comparing 2 floating point variable names equivalenced to integer variables to use function ISET. (This test has previously caused floating point underflows on some platforms.) (4) Added a "PARM=CHECK" option that calls PICTURE from within the SAS4 to plot geometry but not run MORSE. SAS4 prepares or reads MARS geometry input data, reads PICTURE input, and calls PICTURE. Several other changes were made to error messages and formats. (5) The convergence criteria was updated because the criteria in XSDRNPM were changed. (6) The input variable NDAB was added to allow the user to specify the number of direct access blocks allocated.

SAS4 Sample Problems: (1) A ninth sample problem was added to illustrate the new enhanced surface detector option. (2) SAS4 sample problems 1, 3, and 5 were updated to remove references to variables FR1, FR2, FR3, and FR4, which became obsolete in SCALE 4.4.

SCALE Driver: (1) Updated to allow processing the rest of the input data after invalid input data are detected. (2) The driver has been updated to obtain and act on error codes from the modules. The driver now prints error codes and stops further sequence execution.

Standard Composition Library: (1) The default density of B4C was corrected from 2.54 to 2.52 g/cc. This error was introduced in SCALE-4.3. For an LWR fuel problem with B_4C pins between fuel assemblies, the calculated k_{eff} value increased less than 0.2%. (2) Updated to reference the nuclides used for zirconium hydride which have been added to ENDF/B-V libraries and to add four new standard composition names related to zirconium hydride. (3) The densities for SS304 nuclides were made identical to the standard versions of the same nuclides. (4) Updated the standard composition ZIRC2 for consistency with current technical standard and updated densities for SS304 and SS316. (5) Mass of copper was corrected (it was in atomic mass units instead of C-12 mass units). Density of C-GRAPHITE was changed from 1.0 to 2.3 g/cc. The following compositions were added: GRAPHITE, KEROSENE, KERO(H2O), NORPAR13, NORPAR(H2O), POLYVINYLCL, PVC, PVC(H2O), TBP, TBP(H2O).

SUBLIB/UNIXLIB: (1) Updated to remove year 2000 problems. These changes basically changed the year format for the QA verification table to 4 digits. Additionally, the date format was changed to use a 3-character month abbreviation so that the date would be unambiguous. A new line was added to the QA verification table printout to identify the machine on which the program was run. (2) Updated to remove an artificial limit of 8-character-length filenames for non-standard files in subroutine OPNFIL. (3) Modified subroutines LISTQA and VERGET for consistency of the length of the string containing the executable name, the creation date, and the directory path to the executable. The directory path was increased to 256 characters. (4) Updated subroutine FINDQA to place underscores in place of the blanks in the date to simplify the automatic updating of the QA verification table. (5) Replaced the CHARACTER*8 type of variable CAT with a variable length CHARACTER type in subroutine NOTE. This corrected a problem in WAX on the Sun workstation. (6) Added comments to subroutine OPENDA indicating how to replace the Fortran 90 specific INQUIRE statement with a Fortran 77 compatible statement. (7) Replaced all STOP statements with calls to EXIT with the appropriate error return code for proper detection by the driver. (8) Modified subroutine DREAD to correctly process data following the second digit of an exponent when called by the array reading subroutine YREAD. Previously, exponents of 10 or greater sometimes caused errors in the reading of FIDO-type input arrays. This discrepancy was discovered in an ORIGEN-S case. (9) Changes to update XSDRNPM required a double precision ERF function. The necessary routines were added to compute the double precision ERF when it is not part of the intrinsic library.

XSDOSE: An option was added to turn off angular flux print and no angular flux print was made the default.

XSDOSE Sample Problem: The input was modified to turn on the new angular flux print option in XSDOSE.

XSDRNPM: (1) The special activity file and balance table file were not written correctly, and the correct file structure is not what was documented. Subroutine SETUP was changed such that it would not read or write dummy records after the files were opened. These read/writes were the only way to open the files before Fortran 77, but when the code was converted to Fortran 77 and OPEN statements were added to explicitly open the files, the extra statements were not removed. (2) Updated to correct the accumulation of zone fluxes when inner-cell weighting is selected. (3) Updated to correct the value of productions/absorptions when a direct buckling search is done.. (4) The code was also modified to collapse prompt υ and delayed υ using the same procedure as used to collapse the total υ. (5) Corrected calculation of broad group balance tables to be consistent with fine group tables. Broad group cross sections were not in balance when upscatters were collapsed. (6) Updated to print clearer messages when allocated memory is insufficient. Also, in these cases if an output file could not be written, any previously existing file was deleted to prevent subsequent calculations from reading it. (7) The Fortran source for XSDRNPM was converted to Fortran 90 free format. (8) The input/output units were all moved to the 0$ array. The energy of the average lethargy causing fission was added to the balance tables. (9) The output files from the balance tables and the activities were modified and converted to ASCII files. A new file was created which contains the input and derived data from a problem. The flux file was changed to double precision. (10) The code was modified to not run with fluxes out of core unless explicitly requested in the input. (11) The coarse mesh generation algorithm used in rebalancing the inner iterations was modified to correct a problem that prevented convergence of a very small class of problems.

XSECLIST: This is a new utility program for ARP which prints lists of absorption and fission cross sections vs burnup for nuclides from ORIGEN-S multi-burnup binary libraries.

Portability

Version 4.4a of the SCALE system has been developed to ensure portability among various computing platforms. The system is maintained and enhanced at ORNL under quality assurance and configuration management plans. The system has been routinely tested on IBM and DEC workstations. In addition, a version for personal computers (PCs) is included in the package. The PC version runs on Windows 95, 98, and NT4.0 and platforms. The system also has been installed and tested by ORNL on SUN and HP workstations. Information needed to install and run SCALE on each of these systems is included in README files with the software package distributed by the code center.

Related developments

The definition of "easy-to-use" has changed considerably since the late 1970s. As funding has allowed, the ORNL development staff has sought to develop user interfaces that provide a distinct aid to novice or occasional users of the system.

The ORIGNARP input processor is a MS-DOS PC program designed to assist a user in creating an ORIGEN-S input file. It is coupled with the ARP code, which interpolates on standard LWR ORIGEN-S binary libraries, in the ORIGEN-ARP system (Section D1).

CSPAN (Criticality Safety Input Processor for Analysis) is the Windows GUI replacement for the CSASIN input processor for the CSAS criticality sequences in SCALE. CSASIN was an MS-DOS program developed in 1990–91 to assist new and occasional SCALE users. Because CSASIN is incompatible with Pentium II and later PCs, a new easier to use and more powerful Windows program has been developed. CSPAN can be used to read and modify an existing SCALE input file or to create a new input file. CSPAN can call SCALE to execute CSAS using the input file it creates. The SCALE Standard Composition library and the selected SCALE cross-section library are read by CSPAN and the user is only allowed access to those compositions available on the selected cross-section library. The program handles the entry of basic standard compositions, solutions, and arbitrary materials, unit cell data, optional parameter data, and KENO V.a input data. CSPAN can call SCALE to execute any CSAS case. CSPAN runs under Windows 95, 98, or NT. Checks for errors are included throughout the program to verify that the input is valid. The initial version distributed with SCALE 4.4a is considered a beta test version. Help files have not been developed yet, but will be made available soon.

The initial version of a Windows-based GUI for HEATING named Visual Heating is also included in the SCALE 4.4a release. Visual Heating assists the user in preparing a HEATING input file and includes a 3-D graphics display of HEATING geometry models using OpenGL. Visual HEATING can execute the HEATING case in SCALE and display the output file in a text editor. It includes an HTML Help system similar to many commercial Windows programs. The help system is accessible both from the main menu bar and by pressing the F1 key. Most of the information in the HEATING User's Manual (Sect. F10 of the SCALE Manual) is included in the help system along with explanations of Visual HEATING input screens.

Availability

The SCALE code system and the other software designated under "Related Developments" have been packaged by the Radiation Safety Information Computational Center (RSICC). The SCALE system and the related software may be obtained by contacting

Radiation Safety Information Computational Center
Oak Ridge National Laboratory
P.O. Box 2008
Oak Ridge, TN 37831-6362
Telephone: (865) 574-6176
FAX: (865) 574-6182
E-mail: rsic@ornl.gov
Internet: http://www-rsicc.ornl.gov

Acknowledgments

The SCALE system is maintained at ORNL and enhanced to keep pace with normal technical advancements in the analysis areas of interest. Although the NRC continues its role as the controlling sponsor of the SCALE system, the U.S. Department of Energy (DOE) began assisting in the maintenance of the SCALE system in 1987. Over the years numerous individuals within these sponsoring organizations have played key roles in ensuring that the SCALE system remained a readily available, reliable system for the analysis of nuclear fuel facilities and packages. The individuals who have worked with the ORNL staff to coordinate maintenance and development activities include R. H. Odegaarden (NRC, ret.), G. H. Bidinger (NRC, ret.), C. Mauck (DOE, ret.), E. P. Easton (NRC), W. H. Lake (NRC and DOE), M. E. Wangler (DOE), and M. G. Bailey (NRC).

As demonstrated by this Manual, there are also numerous individuals from the ORNL staff who have contributed significantly to the development and enhancement of the SCALE system. Most are credited by their authorship of the sections in this Manual that correspond to their work. A few individuals have been essential to the development and maintenance of SCALE but are not credited by authorship. These individuals include: S. K. Lichtenwalter, who is responsible for implementing and controlling software system changes; C. H. Shappert, who provided the editorial review of this Manual; and L. F. Norris (ret.) and W. C. Carter, who prepared the entire manuscript. Special acknowledgement is also due to R. M. Westfall and G. E. Whitesides (ret.) who, together with R. H. Odegaarden of the NRC, developed the concept and long-range goals of the SCALE system in the late 1970s. Finally, this Project Leader will always be grateful to C. V. Parks, who served as the SCALE Project Leader for the first 15 years, and L. M. Petrie, who for 20 years has consistently provided consultation and advice on the technical direction that should be taken in development of nearly every module and cross-section library that are in the present system.

Stephen M. Bowman

Stephen M. Bowman
SCALE Project Leader
December 1999

References

1. N. M. Greene, W. E. Ford III, L. M. Petrie, and J. W. Arwood, *AMPX-77: A Modular Code System for Generating Coupled Multigroup Neutron-Gamma Cross-Section Libraries from ENDF/B-IV and/or ENDF/B-V*, ORNL/CSD/TM-283, Martin Marietta Energy Systems, Inc., Oak Ridge National Laboratory, October 1992.

2. N. M. Greene, J. W. Arwood, R. Q. Wright, C. V. Parks, *The LAW Library–A Multigroup Cross-Section Library for Use in Radioactive Waste Analysis Calculations*, ORNL/TM-12370, Martin Marietta Energy Systems, Inc., Oak Ridge National Laboratory, August 1994.

3. M. D. DeHart and S. M. Bowman, *Validation of the SCALE Broad Structure 44-Group ENDF/B-V Cross-Section Library for Use in Criticality Safety Analyses*, NUREG/CR-6102 (ORNL/TM-12460), U.S. Nuclear Regulatory Commission, Oak Ridge National Laboratory, September 1994.

Table 1 Analysis capabilities summary of the SCALE control modules

Control module	Analysis function(s)	Functional modules executed	Section reference
CSAS	1-D deterministic calculation of neutron multiplication 3-D Monte Carlo calculation of neutron multiplication Problem-dependent cross-section processing Multiplication search or spacing	BONAMI NITAWL-II XSDRNPM KENO V.a ICE	C4
CSAS6	3-D Monte Carlo calculation of neutron multiplication	BONAMI NITAWL-II XSDRNPM KENO-VI	C6
ORIGEN-ARP	Point depletion/decay of nuclear fuel and radioactive material	ARP ORIGEN-S	D1
SAS1	1-D deterministic calculation of radiation transport through shield and dose evaluation at a point Calculation of dose at detector based on leakage from critical volume	BONAMI NITAWL-II XSDRNPM XSDOSE	S1
SAS2	Point depletion/decay of nuclear fuel 1-D radial shielding analysis in cylindrical geometry	BONAMI NITAWL-II XSDRNPM COUPLE ORIGEN-S XSDOSE	S2
SAS3	Dose evaluation using MORSE Monte Carlo code	BONAMI NITAWL-II XSDRNPM MORSE-SGC	S3
SAS4	Calculation of dose outside of transportation package using MORSE code and automated biasing techniques	BONAMI NITAWL-II XSDRNPM MORSE-SGC	S4
QADS	3-D point-kernel gamma-ray shielding analysis	QAD-CGGP	S5
HTAS1	R-Z steady-state and transient analyses of a transportation package	OCULAR HEATING	H1

Table 2 Analysis capabilities summary of the SCALE functional modules

Module	Function	Section reference
BONAMI	Resonance self-shielding of cross sections with Bondarenko factors	F1
NITAWL-II	Resonance self-shielding of cross sections with resolved resonance data	F2
XSDRNPM	General 1-D, discrete-ordinates code for: • zone-weighting of cross sections • eigenvalue calculations for neutron multiplication • fixed-source calculation for shielding analysis • adjoint calculation for determining importance functions	F3
XSDOSE	Module for calculation of dose at a point based on the 1-D leakage flux from a finite shield	F4
COUPLE	Interface module for preparation of cross-section and spectral data for ORIGEN-S	F6
ORIGEN-S	General-purpose point-depletion and decay code to calculate isotopic, decay heat, radiation source terms, and curie levels	F7
ICE	Cross-section utility module for mixing cross sections	F8
MORSE-SGC	Monte Carlo code with combinatorial and array geometry features used to perform radiation shielding analysis	F9
HEATING7.2	Finite-volume, multidimensional code for conduction and radiation heat transfer	F10
KENO V.a	Monte Carlo code for calculation of neutron multiplication factors	F11
OCULAR	Calculation of radiation exchange factors	F16
KENO-VI	Monte Carlo code for calculation of neutron multiplication factors for complex geometries	F17

NUREG/CR-0200
Revision 6
Volume 2, Section F9
ORNL/NUREG/CSD-2/V2/R6

Computational Physics and Engineering Division

MORSE-SGC FOR THE SCALE SYSTEM

J. T. West*
T. J. Hoffman*
M. B. Emmett

Date Published: March 2000

Prepared for the
Office of Nuclear Material Safety and Safeguards
U.S. Nuclear Regulatory Commission
Washington, DC 20555
Under Interagency Agreement DOE 1886-C000-9B
NRC JCN No. B0009

Prepared by the
OAK RIDGE NATIONAL LABORATORY
managed by
LOCKHEED MARTIN ENERGY RESEARCH CORP.
for the
U.S. DEPARTMENT OF ENERGY
under contract DE-AC05-96OR22464

*Formerly with Oak Ridge National Laboratory.

ABSTRACT

MORSE-SGC was originally developed to provide a version of the popular MORSE-CG code with supergroup cross-section storage. When updated for inclusion within the SCALE system, a new, improved geometry system (MARS, Multiple ARray System) for nested rectangular array analysis was incorporated. MORSE-SGC is written to solve a wide variety of shielding and criticality problems without user-required subroutines.

CONTENTS

Page

CONTENTS (continued)

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

MORSE-SGC is a member of the Oak Ridge National Laboratory (ORNL) MORSE Monte Carlo family of codes. The original developers of MORSE were E. A. Straker, P. N. Stevens, D. C. Irving, V. R. Cain, and M. B. Emmett. S. K. Fraley was responsible for writing and developing the supergroup cross-section storage and tracking as it appears in MORSE-SGC. S. K. Fraley wrote the ICE program to mix cross sections external to MORSE-SGC. The development of MORSE-SGC occurred under the guidance and consultation of L. M. Petrie and M. B. Emmett.

F9.1 INTRODUCTION

MORSE-SGC is a member of the MORSE[1] family of Monte Carlo programs developed at Oak Ridge National Laboratory (ORNL). SGC,[2] which was originally developed prior to the existence of SCALE, was incorporated into the SCALE system to run in the SAS3 Control Module (see Sect. S3), but is capable of running "stand alone" for either shielding or criticality problems.

F9.1.1 MORSE-SGC FEATURES

1. Cross-section storage is supergrouped. MORSE-SGC does not store all cross sections in memory at any given time. Cross sections are stored in available computer memory and are transferred to and from mass disk storage as needed. This feature allows MORSE-SGC to run fine-group calculations with many different media and high-order Legendre expansions for the scattering cross sections with a minimum amount of computer memory requirements.

2. Particle tracking is supergrouped (see Sect. F9.2.1). Individual particles are tracked through one supergroup at a time. If a particle survives the supergroup and enters the next supergroup; the particle is banked for later processing and retrieval.

3. The Multiple ARray System (MARS) geometry package (see Sect. M9) in MORSE-SGC allows complicated rectangular arrays to be modeled with a minimum of computer memory requirements and a minimum of user input. This feature greatly enhances the lattice modeling capabilities of the code. (See Sample Problem, Sect. F9.C.)

4. Combinatorial geometry is integrated into the MARS geometry system, thereby giving the user a great deal of flexibility in modeling irregular geometric shapes. It simplifies user input descriptions by using combinatorial logic to describe material and shapes. (See Sect. F9.2.2.)

5. MORSE-SGC allows immediate calculational capability without user-supplied subroutines. User flexibility is enhanced by a general volumetric source sampling subroutine, a generalized array flux editing capability, a point detector flux option, and several problem-independent biasing schemes.

MORSE-SGC is unique from other versions of MORSE since it has supergrouped cross-section storage and tracking features. The original MORSE-CG package (which ORNL no longer maintains) does not have the free-form combinatorial input, the MARS array capability, or the supergroup storage and tracking features. It does have routines for coupling to discrete-ordinates DOT[3] calculations via the DOMINO[4] code, and it has a new Klein-Nishina Gamma Estimator option which was also added to MORSE-SGC in 1988. MORSE-CG has the BREESE[5] peripheral routines for using special albedo boundaries defined from library data. In some respects, the coding in the original MORSE-CG is easier to modify for special problems than MORSE-SGC. This is due largely to the complexity of array tracking with MARS. MORSE-CGA[6] is the latest in the MORSE-CG family of codes, having been first distributed in 1985. CGA does contain the MARS array capability and free-form input. All three versions of MORSE have been benchmarked. MORSE-SGC was used to study the reactivity effects from reactor core disruptive accidents related to Three Mile Island (TMI). This study involved extensive benchmarking of the MARS lattice system, comparison with TMI plant data and KENO-IV (see Sect. F5).

The basic theory behind all of the MORSE codes is identical and has been documented in Ref. 1. The main portion of this theory is repeated in this document as Sect. F9.D. Ref. 1 also contained instructions for the user on combining multiple runs in order to get lower fractional standard deviations; this discussion is included in this document in Sect. F9.F.

F9.1.2 REFERENCES

1. M. B. Emmett, *The MORSE Monte Carlo Radiation Transport System,* ORNL-4972, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., February 1975.

2. S. K Fraley, *Users Guide to MORSE-SGC,* ORNL/CSD-7, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., March 1978.

3. W. A. Rhoades, D. B. Simpson, R. L. Childs, and W. W. Engle, Jr., *The DOT-IV Two-Dimensional Discrete Ordinates Transport Code With Space-Dependent Mesh and Quadrature,* ORNL/TM-6529, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., January 1979.

4. M. B. Emmett, C. E. Burgart, and T. J. Hoffman, *DOMINO, A General Purpose Code for Coupling Discrete Ordinates and Monte Carlo Radiation Transport Calculations,* ORNL-4853, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., July 1973.

5. V. R. Cain and M. B. Emmett, *BREESE-II. Auxiliary Routines for Implementing the Albedo Option in the MORSE Monte Carlo Code,* ORNL/TM-6807, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., July 1979.

6. M. B. Emmett, *MORSE-CGA, A Monte Carlo Radiation Transport Code With Array Geometry Capability,* ORNL-6174, Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., April 1985.

F9.2 MORSE-SGC TRACKING AND MODELING CAPABILITIES

F9.2.1 MORSE-SGC SUPERGROUP TRACKING

After reading most of the problem input, MORSE determines the minimum number of supergroups required for a problem and the maximum number of energy groups possible in each supergroup based on available computer memory. Then the cross sections are either mixed or read in from a previously mixed ICE Monte Carlo tape (see Sect. F8). Each block of cross-section data is placed on a direct-access file for efficient retrieval.

Source particles are initially sorted in supergroup order. A source particle is retrieved from the particle bank, and tracking is initiated. The particle is tracked until it is either killed, escapes, or enters another supergroup. If the particle enters another supergroup, then the particle tracking information is banked for future retrieval. When tracking through a complex array model, the particle coordinate banked is always the global-level zero coordinate. This arrangement eliminates the necessity of banking the particle's nesting table with the particle. The nesting table is regenerated when the particle is retrieved from the bank. All particles in a given supergroup are tracked before particles in the next supergroup are processed.

The supergrouping of cross-section storage and tracking allows fine-group cross-section libraries, high-order Legendre scattering expansions, and many different material mixtures to be run on moderate-to-small computer systems. This arrangement is accomplished with a minimum of input-output requirements.

The segmentation of MORSE complements the supergrouping capability by requiring a smaller amount of computer memory for the program itself. For many problems only a few supergroups are required, since most of the computer memory is available for data storage.

F9.2.2 COMBINATORIAL MARS GEOMETRY MODELING

The geometry system in MORSE-SGC is an enhanced version of the combinatorial geometry system used in earlier versions of MORSE. Combinatorial geometry is an integral part of the new MARS geometry system. MARS is now called by MORSE-SGC, instead of combinatorial geometry. MARS determines whether a particle is located inside of an array, and if so, the lattice universe type. MARS then calls combinatorial geometry to handle zone tracking. Combinatorial geometry is a technique of describing material zones as the union or the intersection of simple geometric bodies. A body in combinatorial geometry is an enclosed volume composed of several surfaces. The body descriptions determine the equations for each of the individual geometric surfaces composing the combinatorial body. This combinatorial approach to modeling makes generating complicated models containing irregular geometric shapes easier.

The modeling capability in MORSE-SGC is further enhanced by the lattice modeling ability provided by MARS. This feature enables the user to model rectangular cells of arbitrary content called universes. The user may combine universes to describe arrays, and he may construct other arrays containing smaller arrays. In MARS the user may nest arrays without limit. Furthermore, arrays may contain arbitrary vacancies either external or internal to the array. Arrays may be arbitrarily repeated in space by both rotation and translation. Array repetition and universe repetition do not require additional computer memory. MARS is described in detail in Sect. M9 of the SCALE document.

Model verification of a MARS geometry is available with PICTURE, which provides a printed view of arbitrary two-dimensional (2-D) slices through the geometry (see Sect. M13).

F9.3 BIASING TECHNIQUES IN MORSE-SGC

A Monte Carlo calculation consists of a statistical solution to a radiation transport problem. As such, the calculated radiation effect has a standard deviation associated with it. The purpose of biasing a Monte Carlo calculation is to reduce this standard deviation. For many practical problems, meaningful results (i.e., those with a standard deviation that is less than 20% of the calculated radiation effect) can only be obtained by proper biasing.

Briefly, biasing consists of altering the natural distributions from which particle parameters (position, energy, direction, and time) are randomly selected in a Monte Carlo calculation. The goal is to spend more computation time tracking those particles that are more likely to penetrate the shield. To account for this bias, weight corrections, called "statistical weights," are applied to the calculated radiation effects.

MORSE-SGC has several biasing options available to the user: splitting and Russian roulette, the exponential transform, source energy and direction biasing, and energy biasing at collision sites. To use these options, the user must furnish appropriate biasing parameters. Default values are not provided.

Since it is anticipated that MORSE-SGC will be used primarily on shielding problems,* the discussion in this section will emphasize shielding applications. Biasing parameters for shielding problems tend to be highly problem dependent. Therefore, only general guidelines for the specification of these parameters are possible.

F9.3.1 SPLITTING AND RUSSIAN ROULETTE

Probably the most widely used biasing technique for deep-penetration problems is that of splitting accompanied by Russian roulette. To illustrate its use, consider the calculation of radiation leakage from a spent fuel shipping cask. Source particles, are incident on the inner surface of the shield. At a specified distance into the shield, a particle is split into two identical particles, each with half the statistical weight of the original particle. Clearly, weight is preserved, but twice as many particles are processed beyond the specified distance. The standard deviation of the calculated leakage should improve since the number of particles that reach the cask surface will roughly double.

If it is desirable to split particles when they penetrate deeper into a shield, or, in general, when they enter a more important region of the problem, then it is usually appropriate to decrease the number of particles when they enter a less desirable region. This reduction is achieved by Russian roulette. In the previous example, if a particle returns to the inner portion of the cask, Russian roulette may be played by allowing the particle to survive with probability 0.5. If the particle survives, its weight is increased by a factor of 2. To preserve the weight, the weight is set to zero if the particle fails to survive Russian roulette (i.e., the particle history is terminated). Again, weight is conserved, but about half as many particles will be processed in this less important region.

In MORSE-SGC, the user specifies the importance regions on the Region Card (A.5) of the MARS input. For each importance region, three energy-dependent weight standards are required to control splitting and Russian roulette: WTHI, WTLO, and WTAV. These parameters are input in the 8* array. Particles with statistical weights above WTHI will be split. Russian roulette will be played when a particle's weight (WATE) is below WTLO. If a particle survives Russian roulette with probability WATE/WTAV, it will be given a weight equal to WTAV.

*KENO, which usually requires less setup and computer time than MORSE-SGC, is recommended for criticality calculations.

The amount of information required to implement splitting and Russian roulette should not discourage the user from using these techniques. Even crude values for these parameters will save computation time if the following guidelines are used:

1. For the importance region that contains the source, the value of WTAV should be consistent with the source biasing that is being used (see Sect. F9.3.3); if no source biasing is used, WTAV should be set to 1.0 for all energy groups in the importance region that contains the source.

2. WTAV should be decreased by about a factor of 2 for each mean-free path as one moves from source to detector.

3. For all regions and energy groups, WTLO should be about equal to WTAV/5 and WTHI should be about equal to 5*WTAV.

These guidelines are based in part on experience at ORNL and LANL and, in part, on considerations of the behavior of the adjoint flux. They are demonstrated in Sect. F9.C.1 by application to a spent fuel shipping cask problem.

F9.3.2 THE EXPONENTIAL TRANSFORM

The exponential transform is frequently called "path-length stretching." The idea is to artificially reduce the number of mean-free paths between source and detector. To preserve the weight, particles with longer travel distances are assigned appropriately smaller statistical weights. In this manner more particles will penetrate the shield and therefore, with the increased number of contributions, the standard deviation of the calculated radiation effect should be reduced.

As implemented in MORSE-SGC, the number of mean-free paths will be modified by the following factor:

1.0/(1.0-PATH(IG,NREG)*DIREC).

The appropriate weight correction is calculated and applied to the particle's contribution by the code; however, the user must supply the biasing parameters DIREC and PATH. DIREC is usually the cosine of the angle between the flight direction and the preferred direction of travel; thus it is restricted to the range -1 to 1. Note that particles with DIREC < 0 are moving away from the important direction; thus, the number of mean-free paths is decreased rather than increased. PATH is a measure of the degree to which the path is stretched and varies from 0 (no stretching) to 1 (infinite stretching).

In the older versions of MORSE, the user had to supply a subprogram DIREC. DIREC was to compute the angle between the particle trajectory and the preferred direction of travel. It returned the cosine of this angle. The current version of MORSE-SGC gives the user many options for biasing the path length without requiring a user-supplied subprogram. The options are discussed in Sect. F9.A.2 (see parameter NDSG of the 3$ array).

Some caution must be used in arbitrarily assigning values to PATH. If these values are too large, one can "overbias" the problem. What this means is that one can obtain a low standard deviation for a very poor estimate of the detector response. What happens is that one continually obtains good estimates of the first few terms in the Neumann series solution to the problem, but, due to the "overstretching," the particles are not in the system long enough to provide estimates of the higher order (and perhaps most important) terms of the series. The net result is a consistent underestimate of the radiation effect.

Although much has been written about overbiasing, little has been said about "underbiasing." Underbiasing is a problem that frequently occurs when a next-event estimator, such as RELCOL or RELCOA (see Table F9.A.1), is used. This problem occurs because the next-event estimator will often produce consistent (i.e., low standard deviation) underestimates of the radiation effect if few particles have collisions in the vicinity of the detector. One of the reasons the next-event estimator is so popular is that it "always gives an answer" because it estimates from every collision site. However, this answer may be meaningless if undersampling of the important regions of the problem has occurred. Therefore, if the user decides to use RELCOL or RELCOA (i.e., sets IFLAG(9) of the 5$ array to +1 or −1) he should make sure that adequate sampling results. Proper sampling can be done by noting the number of escapes in the general vicinity of the point detector. For example, if a detector is located above a shipping cask and the upper axial leakage consists of fewer than 100 particles, the detector response should be reviewed with skepticism regardless of how low the standard deviation is or how many contributions were made to the detector. A low standard deviation associated with low leakage in the vicinity of the detector means the user should probably increase PATH and/or lower the weight standards. (Note: If a detector response is desired at points other than a void, the versions of RELCOL and RELCOA that are in MORSE-SGC should not be used.)

Unlike the use of splitting and Russian roulette that are "easy to use and difficult to abuse," the improper use of path-length stretching can lead to very "accurate" calculations of the wrong answer. Since overbiasing is more difficult to detect than underbiasing, the user is advised to use low values of PATH and the previously mentioned checks for adequate sampling. Based on the above observations, personal experience, and work done by J. Spanier,[1] ORNL, and LANL,[2] the following guidelines are offered to the wary users:

1. For regions within one mean-free path of the detector, set PATH to zero for all energy groups.

2. For more distant regions, set PATH to 0.5 for the higher energy neutron groups (E ≥ 1 MeV), and to 0.0 for the lower energy neutron groups.

3. For gamma-ray energy groups, somewhat larger values, around 0.7, for regions more than a mean-free path from the detector can be safely used.

4. Always accompany path-length stretching with the weight standards previously described.

F9.3.3 SOURCE BIASING

Source energy biasing is available in MORSE-SGC. Source energy biasing is implemented when IEBIAS of the 4$ array is set to 1, and it requires input of the 18* array.

In source energy biasing, the user inputs the relative importance of each source energy group in the 18* array, BFS(IG) (see Sect. F9.A.4). MORSE-SGC multiplies these values by the natural source distribution, FS(IG) of the 17* array, normalizes the product by dividing by the sum over all energy groups, and uses the normalized distribution as the source distribution. Again, the weight corrections are calculated and applied to the histories by the code. BFS(IG) must be furnished by the user.

High-energy-source particles tend to be more penetrating than low-energy particles and, therefore, a bias to the high-energy source particles is usually appropriate in shielding calculations. However, if fissile material is present in or near the source region or secondary gamma-ray responses are being calculated, low-energy neutrons may be more important than high-energy neutrons. For example, for the sample problem described in F9.C.2, a neutron in the source region with energy less than 1 eV contributes more to the detector response than a neutron at the same position with an energy of 3 MeV. The reason, of course, is not that the 1-eV neutron has a greater chance of penetrating the shield, but rather that it can produce neutrons through fission that do have a higher leakage probability than does a 3-MeV neutron. The point is that simply knowing the number of mean-free paths from source to detector for each energy group is not sufficient information to determine values of BFS.

An approach to source energy biasing that will improve results for most problems is to obtain a one-dimensional (1-D) model that consists of the materials and thicknesses encountered as one moves directly from source to detector. The 1-D model is then represented in another SCALE code, XSDRNPM and an adjoint calculation is performed. The adjoint fluxes output by XSDRNPM (labelled "TOTAL FLUX" in the summary tables) for the zone that contains the source are excellent values to use for BFS. This approach is not valid if voids that provide streaming paths are present in the shield. In this case, source energy biasing is best avoided and weight standards as described in Sect. F9.3.1 used.

If source energy biasing is used, it is essential to adjust the weight standards accordingly. If this is not done, the source biasing will be ineffective. MORSE-SGC will simply play Russian roulette and split the source particles until their weights lie within the weight window. The adjustment of the weight standards to yield values that are consistent with source biasing is illustrated in Sect. F9.C.2 for source energy biasing.

F9.3.4 ENERGY BIASING AT COLLISION SITES

This type of biasing is very similar to source energy biasing. It is activated by setting IEBIAS of the 1$ array to one and furnishing the 9* array (i.e., EPROB) for all energy groups and all regions.

The energy of a particle emerging from a collision is normally selected by sampling from the downscatter probabilities for the incident energy group. (These downscatter probabilities are printed out by MORSE-SGC before tracking under the title "CROSS SECTIONS FOR MEDIA NUMBER N.") When energy biasing is used, the probability of selecting a given energy group is proportional to the product of EPROB as furnished by the user and the normal downscatter probability. Similar to source energy biasing, the adjoint fluxes obtained from an XSDRNPM calculation are excellent values to use for EPROB.

F9.3.5 REFERENCES

1. J. Spanier, *A New Multi-Stage Procedure for Systematic Variance Reduction in Monte Carlo,* CONF-710302,1971.

2. W. L. Thompson, O. L. Deutsch, and T. E. Booth, *Deep-Penetration Calculations,* ORNL/RSIC-44, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., 1980.

F9.4 SOURCE SAMPLING AND ANALYSIS TECHNIQUES

In running a Monte Carlo calculation, many factors must be taken into consideration to describe a problem. The source sampling technique, the biasing games played, and the analysis methods applied are several important considerations in using any general Monte Carlo transport code. This section describes the source sampling technique available in MORSE-SGC and several of the analysis methods operational in MORSE-SGC.

F9.4.1 VOLUMETRIC SOURCE SAMPLING

In MORSE-SGC the user inputs a source volume space defined with bounds XMIN, XMAX, YMIN, YMAX, ZMIN, and ZMAX. The source subroutine in MORSE uniformly samples locations within this defined volume. If the user specified a source media, the code only accepts points found in the source media. If a source media is not given, then all points sampled in the defined source volume are assumed to be valid source points. For shielding calculations this process occurs at the beginning of every batch. For criticality calculations this process only occurs at the beginning of the first batch. Subsequent batch source particles are generated by fission collision sites from the previous batch.

The user is given the option of having the source particles started with a uniform isotropic angular distribution or having the initial source direction sampled from the biased distribution described in Sect. F9.3.3.

Point source problems may be run by setting XMAX equal to XMIN, YMAX equal to YMIN, and ZMAX equal to ZMIN. Likewise, planar source problems may be run in a similar manner, although hemispherical angular sampling is not available in the current source routine. Also nonuniform sampling distributions are not available in the current version of MORSE-SGC.

F9.4.2 POINT DETECTOR ESTIMATES

MORSE-SGC allows the user two methods of computing the reaction rate at a point detector. Either a single or multiple estimate may be made to each point detector from each collision site. These estimates are made by subroutine RELCOL or RELCOA. For some problems, making multiple estimates from each possible downscatter group at a collision site to the point detector will improve the standard deviation of a response. The user decides which method to use by setting IFLAG(9) in the MORSE input. Subroutine RELCOA makes multiple estimates from a collision site. It is called when IFLAG(9) is set to −1. Subroutine RELCOL makes a single estimate to each point detector from each collision site when IFLAG(9) is set to +1. When IFLAG(9) is set to zero, then no point detector estimates are made from collision sites.

On some problems it is possible to compute negative fluxes with the collided point detector estimators. When this occurs, the problem has yielded erroneous results. Negative estimates result from negative scattering probabilities. When the point detector results are negative, the user should modify his problem and/or his biasing schemes (see Sect. F9.3.2 also).

When making point detector estimates, subroutine SDATA makes uncollided flux estimates from source locations and subroutine SGAM makes uncollided flux estimates for secondary-gamma births. The total flux response printed out by MORSE is the sum of the collided flux response and the uncollided flux response.

F9.4.3 THE KLEIN-NISHINA OPTION IN MORSE

The MORSE-SGC code has been modified to allow for reading, storing, and processing gamma-ray pair production and Compton scattering cross sections. This modification is necessary in order to use the Klein-Nishina formula for making estimates in RELCOA.

The MORSE-SGC routines that required modification were XSEC3, XSEC4, XSEC8, RINPU1, RINPU2, and RELCOA. Both temporary and permanent cross-section storage had to be allocated for the two new cross sections-pair production and Compton scattering. Since there are no fission gammas in the available cross-section libraries, the gamma portion of the $\nu\Sigma_f$ array was used for pair production data. The Compton-scattering cross section is stored in the gamma portion of the gamma production data.

The user indicates the use of this option by setting NCOEF = -N, where N is the number of coefficients.

A sample RELCOA has been written, but some of the burden of making sure it works for the user's particular problem falls on the user. For example, users who run combined neutron-gamma problems as all primary particles must specify internal to RELCOA just how many groups are actually neutrons. This step is necessary in order to ensure that the Klein-Nishina method is used only for gamma rays and not for neutrons. Estimates for neutrons do not involve the Klein-Nishina formula but are done as the usual point-detector next--flight estimator. RELCOA is somewhat like two separate routines because it uses two entirely different methods for making estimates for neutrons and gammas. For gamma groups, a choice is made between pair production and Compton estimates.

A description of RELCOA follows:

Subroutine RELCOA(D,XD,YD,ZD,VEL,EXTRG,EXTRD,NMTG,ND)

The fluence estimate for neutron groups, from the normal collisions in RELCOA, is given by

$$\frac{\text{WATE} \times e^{\text{ARG}} \times \text{p(THETA,IG)}}{R^2},$$

where WATE is the statistical weight of the particle leaving the collision, ARG is the negative of the number of mean-free paths from collision to detector, R is the distance from collision to detector, P(THETA,IGO) is the probability, per steradian, for a particle with energy in the incoming group IGO scattering to a lower energy group through angle $\cos^{-1}$ (THETA) and THETA is the cosine of the angle between the incoming direction and the line from the collision site to the point detector. PTHETA generates the P array.

The fluence estimate for gamma groups, from the normal collisions in RELCOA, has several possible forms. For a forward problem, a choice is made between Compton and pair production estimates by selecting a random number, R1, and using a Compton scattering estimate if

$$R1 < \frac{\Sigma_c}{2\Sigma_{pp} + \Sigma_c};$$

otherwise, pair production is used. The Compton scattering estimate is

$$\frac{\text{WATE} \times e^{\text{ARG}} \times \text{PMU}}{R^2},$$

where WATE, ARG, and R are as described above, and PMU is the probability of scattering calculated by the usual Klein-Nishina formulas.

The pair production estimate is $\frac{\text{WATE} \times e^{\text{ARG}}}{4\pi R^2}$ and is made to group IZ, containing 0.511 MeV.

Called by: SUPGRP when IFLAG(9) = - 1.

Routines called:
SQRT, EXP
PTHETA
EUCLID
PTPT
FLUXST
FLTRN

Commons required: POINT, QDET, NUTRON, APOLL, PERM, INPUT, GOMLOC, PTBIAS, SCALOP, NGAMGP, Blank.

F9.4.4 MARS ARRAY ANALYSIS

The MARS geometry system in MORSE-SGC allows the user to model very complicated lattice arrays with minimum difficulty. In Monte Carlo tracking, the user should be aware of where particles are tracked and the number of collisions occurring throughout the model. This information is necessary to properly determine if the calculational results are realistic or in error. MORSE-SGC contains a collided flux editor for the MARS system. This editor allows the user to obtain reaction rates throughout his array geometry as a function of array or universe (cell type) and as a function of media. The collided flux editor for MARS is described in detail in Sect. M9.4. The Collided Flux Edit System (CFE) provides the user with valuable information without adding additional computation time. The MARS user in MORSE-SGC should utilize the CFE feature where possible to obtain the maximum amount of information from his calculation.

F9.A MORSE-SGC INPUT INSTRUCTIONS

F9.A.1 MORSE DATA BLOCK 1

Card A (20A4)

Title Card

Card B (2Z8)

RANDOM - starting random-number seed. Seed should start in position 5 (i.e., first 4 characters should be blank).

F9.A.2 MORSE DATA BLOCK 2 (FIDO INPUT)

1$ (14)

1. IADJM set >0 for an adjoint problem. All input data should still be in the forward mode; the program will adjoint it whenever IADJM > 0.

2. NSTRT number of particles per batch.

3. NMOST maximum number of particles allowed for in the bank(s); NSTRT + 1 is the minimum value for NMOST (if no splitting, fission, and secondary generation). If NSTRT = NMOST, NMOST will be reset to NSTRT + 1.

4. NITS number of batches.

5. NQUIT number of sets of NITS batches to be run without new input data.

6. INB option to print (=1) or not print (=0) responses at end of each batch. Default 0.

7. ISTAT set >0 to store Legendre coefficients.

8. NSPLT set >0 if splitting is allowed.

9. NKILL set >0 if Russian roulette is allowed.

10. NPAST set >0 if exponential transform is invoked (subroutine DIREC is called).

11. NOLEAK set >0 if nonleakage is invoked.

12. IEBIAS set >0 if energy biasing is allowed.

13. NKCALC the number of the first batch to be included in the estimate of k; if ≤0 no estimate of k is made.

14. NORMF the weight standards and fission weights are unchanged if ≤0; otherwise, fission weights will be multiplied at the end of each batch by the latest estimate of k, and the weight standards are multiplied by the ratio of fission weights produced in previous batch to the average starting weight for the previous batch. For time-dependent subcritical systems, NORMF should be >0.

2$ (5)

1. MEDIA number of cross-section media.

2. NMIX number of mixing operations (elements times density operations) to be performed.

3. MEDALB = 0 if no specular reflection is used; >0 if specular reflection is used; value of MEDALB is used as the reflecting media.

4. MXREG number of regions described by the geometry input.

5. MFISTP set >0 if fissions are allowed.

3$ (20)

1. NNGA number of neutron groups to be analyzed.

2. NGGA number of gamma groups to be analyzed.

3. NNGTP set >0 if a completely coupled neutron-gamma problem is desired (both neutrons and gammas are treated as primary particles). NNGA and NGGA must be equal to the number of neutron and gamma groups on the input cross-section tape if this option is used.

4. NDAB the number of direct access blocks allocated (default is 1000).

5. NOT USED set = 0

6. NDSN number of array analysis collision edits; see input discussion.

7. NDSG if NPAST is >0, NDSG determines the direction of PATHLENGTH STRETCHING used in subroutine DIREC. The options available are the following:

= 1 pathlength stretched in the +X direction (local)

= 2 pathlength stretched in the +Y direction (local)

= 3 pathlength stretched in the +Z direction (local)

= 4 pathlength stretched in the -X direction (local)

= 5 pathlength stretched in the -Y direction (local)

= 6 pathlength stretched in the -Z direction (local)

= 7 pathlength stretched cylindrically outward on a radial vector perpendicular to the X axis; nonrotated systems only.

= 8 pathlength stretched cylindrically outward on a radial vector perpendicular to the Y axis; nonrotated systems only.

= 9 pathlength stretched cylindrically outward on a radial vector perpendicular to the Z axis; nonrotated systems only.

= 10 pathlength stretched spherically outward.

= 11 through 16 are the same as options 1 through 6, except the coordinate direction cosine used is the global coordinate direction cosine; for systems containing rotated arrays, use options 11 through 16; otherwise, these will have the same effect as options 1 through 6 for nonrotated systems.

= 17 pathlength stretched toward the coordinate of the first point detector.

8. NCOEF number of coefficients for each mixture, including P_o. Set NCOEF<0 to invoke Klein-Nishina options.

9. NSCT number of discrete angles, set =0 for isotropic scattering (usually NCOEF/2 integer).

10. MAXGP group number of last group for which Russian roulette, splitting, or exponential transform is to be performed.

11. IRDSG switch to print the cross sections as they are read if >0

12. ISTR switch to print the cross sections as they are stored if >0.

13. IFMU switch to print intermediate results of μ's calculation if >0.

14. IMOM switch to print moments of angular distribution if >0.

15. IPRIN switch to print the angles and probabilities if >0.

16. IPUN switch to print results of bad Legendre coefficients if >0.

17. IXTAPE = 0 cross sections are to be read from an ICE TAPE.
= the value of IXTAPE is the logical tape unit for the cross-section input data (AMPX working tape).

18. JXTAPE = 0 if AMPX Working Tape is to be read.
= the value of JXTAPE is the logical tape unit for reading an ICE-created cross-section tape.

19. MDIM Maximum array size to be used by MORSE. (Default is 150000.)

20. MSM = 0 no media-dependence for the source selection; all points sampled with the source volume specified will be accepted.

= SOURCE MEDIA NUMBER; only points sampled in the source specified volume in media MSM will be accepted as valid source starting coordinates.

4$ (4)

1. ISOUR source energy group if >0; if ≤0, source input cards will be required.

2. NGPFS number of groups for which the source spectrum is to be defined; must equal NNGA + NGGA for an adjoint problem.

3. ISBIAS set not equal to zero if the source energy is to be biased.

4. NSOUR set ≤0 for a fixed-source problem; otherwise, the source is from fission generated in a previous batch.

5$ (28)

1. ND number of detectors (set = 1 if ≤0).

2. NNE number of primary particle (neutron) energy bins to be used (must be ≤NE).

3. NE total number of energy bins (set = 0 if ≤1).

4. NT number of time bins for each detector (set = 0 if NT ≤1).

5. NA number of angle bins (set = 0 if ≤1).

6. NRESP number of energy-dependent response functions to be used for point detector analysis and for the volume average collided flux edits (set = 1 if ≤0).

7. NEX number of extra arrays of size NNGA + NGGA to be set aside; set NEX = 4 + NRESP for point detector analysis.

8. NEXND number of extra arrays of size ND to be set aside (useful, for example, as a place to store detector-dependent counters). For the point detector analysis, set NEXND to 4.

9-28. IFLAG (20) (see Table F9.A.1).

T (Block 1 must be terminated with a T.)

F9.A.3 FREE-FORM COMBINATORIAL MARS INPUT INSTRUCTIONS

The following ten cards denoted by A.1 through A.10 correspond exactly to the MARS input described in Sect. M9.A. The input is repeated here for convenience to the user.

A.1 **Title Card**

FORMAT (15A4)

A.2 **Options Card (four entries required)**

IVOPT - Volume option not implemented - enter 0.

IDBG - Debug print option if positive; otherwise, enter 0.

IBOD - Body numbers are assigned by the user if IBOD >0; otherwise, enter 0.

NAZ - Number of zones to be added to the data storage for next zone of entry memory table. Enter any large number if extra storage required. Default value allows for five zones to be entered from any single code zone. This option is normally not required, enter 0.

A.3 **Body Definition Cards**

Each new body must start on a new card. The allowable body types are given in Table F9.A.2, along with the required input variables to describe each body. An END card must be used to signify the end of the body definition cards. For each body the following input is required:

ITYPE - Specifies the alphanumeric body type or END to terminate reading of body data (e.g., BOX, RPP, ARB, RCC, etc.)

IALP - Body number assigned by the user if IBOD > 0; otherwise, not entered.

FPD(I) - Real data required for the given body as shown in Table F9.A.1. These data must be in cm.

Table F9.A.1 Subroutine called by IFLAG(I) setting

Subroutine		Initiating Event	Called by	Comment
1	Not Used			
2	Not Used			
3	Not Used			
4	STRUN	Beginning of run	MORSE	User-supplied routine
5	SDATA	Primary Source Birth	MSOUR	Makes uncollided flux estimate to each point detector
6	SPLIT	Particle SPLIT	SUPGRP	User-supplied routine; otherwise, not available
7	FISSIN	Fission	FPROB	User supplied; otherwise, not available
8	SGAM	Secondary particle	GSTORE	Makes uncollided generation site flux estimate to each point detector; identical to SDATA
9	= −1 RELCOA	Real collision	SUPGRP	Multiple estimates to each point detector from each collision site
	= +1 RELCOL		SUPGRP	One estimate to each point detector from each collision site
10	ALBEDO	Albedo reflection	SUPGRP	User-supplied routine; otherwise, not available
11	BDRYX	Boundary crossing across dissimilar media or region	NXTCOL	User supplied or user modification of BDRYX in MORSE required
12	ESCAPE	Particle escape	NXTCOL	User supplied; otherwise, not available
13	ECUT	Energy cut-off	SUPGRP	User supplied; otherwise, not available

Table F9.A.1 (continued)

Subroutine		Initiating Event	Called by	Comment
14	TKILL	Time kill	SUPGRP	User supplied; otherwise, not available
15	RRKILL	Russian roulette kill	SUPGRP	User supplied; otherwise, not available
16	RRSURV	Russian roulette survival	SUPGRP	User supplied; otherwise, not available
17	GAMLST	Secondary particle generation, no room in banks	GSTORE	User supplied; otherwise, not available
18	= 0 COLISN	Collision site	SUPGRP	MORSE standard routine for treating collisions
	= 1 FCOLN	Collision site	SUPGRP	MORSE standard routine for benchmarking rotated arrays with explicit modeling approach
19-20	NOT USED...			

A.4 Input Zone Description Cards

Each new zone must start on a new card. A three-character title should be given for each new input zone (not necessarily unique) which must start with an alpha-type character. An END card must be used to signify the end of the input zone description cards. For each input zone, the data needed are the title and zone data. Input zone numbers are assigned sequentially.

IALP - The three-character title for the zone where the first character is a letter.

IIBIAS(I) - Specify the "OR" operator if required for the JTY(I) body.

JTY(I) - Body number with the (+) or (-) sign as required for the zone description.

Table F9.A.2 Input required for each body type

Body type	ITYPE	IALP	Real data defining particular body					
Box	BOX	IALP is assigned by the user or by the code if left blank.	Vx	Vy	Vz	H1x	H1y	H1z
			H2x	H2y	H2z	H3x	H3y	H3z
Right Parallelepiped	RPP		Xmin	Xmax	Ymin	Ymax	Zmin	Zmax
Sphere	SPH		Vx	Vy	Vz	R	–	–
Right Circular Cylinder	RCC		Vx	Vy	Vz	Hx	Hy	Hz
			R	–	–	–	–	–
Right Elliptical Cylinder	REC		Vx	Vy	Vz	Hx	Hy	Hz
			R1x	R1y	R1z	R2x	R2y	R2z
Ellipsoid	ELL		V1x	V1y	V1z	V2x	V2y	V2z
Truncated Right Cone	TRC		Vx	Vy	Vz	Hx	Hy	Hz
			R1	R2	–	–	–	–
Right-Angle Wedge	WED or RAW		Vx	Vy	Vz	H1x	H1y	H1z
			H2x	H2y	H2z	H3x	H3y	H3z
Arbitrary Polyhedron	ARB		V1x	V1y	V1z	V2x	V2y	V2z
			V3x	V3y	V3z	V4x	V4y	V4z
			V5x	V5y	V5z	V6x	V6y	V6z
			V7x	V7y	V7z	V8x	V8y	V8z
			Face Descriptions (see note below)					
Alternative Body	BPP		Xmin	Xmax	Ymin	Ymax	Zmin	Zmax
			θ_1	θ_2	θ_3			
Descriptions	WPP		Xmin	Ymax	Ymin	Ymax	Zmin	Zmax
			θ_1	θ_2	θ_3			
Termination of Body Input Data	END							

NOTE: The arbitrary polyhedron input contains a four-digit number for each of the six faces of an ARB body.

Example:

```
PEL +1
CLD +2 -1
H2O +3 -2
END
```

A.5 Region Card

One entry is required for each input zone. This specifies the importance region each input zone is inside. This determines which set of weights for splitting, Russian roulette, and pathlength stretching to use in each zone during tracking.

A.6 Universe Card

This array specifies which universe each input zone is inside. One entry is required for each input zone. The entry must be either a zero or a positive integer. A negative entry is not valid. Each universe, with the exception of the null universe, must contain one and only one zone with a -1000 media. The null universe cannot contain any -1000 media zone.

A.7 Media Card

This array specifies the media contained in each input zone. One entry is required for each input zone. If the entry is positive, it references a valid cross-section mixture or a reflected boundary, MEDALB. If the entry is negative, it references a valid array number as the absolute value of the entry. If the entry is -1000, it references a universe external boundary media. If the entry is 1000, it references an internal void. If the entry is 0, it references an external void.

A.8 Array Size Specification Input

An array is a regular rectangular lattice composed of rectangular cells of arbitrary content. The size of each array should be entered as NXMAX, NYMAX, by NZMAX. The arrays are sequentially named as they are entered, starting with 1. The array size entered should include any vacant cells in the array if any are present. After the size of the last array has been entered, a zero should be entered to terminate the entries. Zero is an illegal entry for an array size. After the zero terminator has been entered, a single integer parameter is entered to determine the means of entering the array specification list. If no array is to be described, only the zero terminator is required.

$NXMAX^i$ - The number of cells along the x-axis of array i

$NYMAX^i$ - The number of cells along the y-axis of array i

$NZMAX^i$ - The number of cells along the z-axis of array i followed by a "0 terminator"

IOP - Array specification input option (required)
= 0 free-form input, FFREAD, type specification
= 1 standard KENO mixed cell (BOX) orientation cards
= 2 standard FIDO integer array input specification

Example: 15 15 1 6 5 2 7 7 1 0 0

This example describes three arrays. Array 1 will be 15 × 15 × 1; Array 2, 6 × 5 × 2; and Array 3, 7 × 7 × 1. Zero terminates the array size entries. The last zero entered selects the free-form input specification method of describing the array contents.

--End of geometry input if no arrays are modeled--

A.9 **Array Content Description**

The contents of each cell of each array must be defined. All contents of Array 1 are defined, then Array 2, etc. The method of entering these data is determined by IOP in A.8 input. There are three possibilities for each cell entry. These options are distinguished by either a positive, zero, or negative entry. A positive entry is a universe number. The universe must fit snugly in the lattice cell position that references it. The contents of a universe are completely arbitrary. A negative entry is an array entry. The absolute value of the entry is the array being referenced. It cannot contain any vacancies in its lattice cell positions. Repeating a subarray in a larger array in this manner does not require any additional input. The array must, however, fit snugly in the lattice cell position. The means of entering these data is selected by the user to give flexibility in describing his arrays. The options are the following:

IOP = 0 - Free-Form Input Option

Free-form input is entered using the FFREAD notation. This step allows an "*" repeat feature. Data are entered as:

```
DO 10 M =1, NAR (NAR is the number of arrays entered)

DO 10 K = 1, NZMAX

DO 10 J = 1, NYMAX

DO 10 I = 1, NXMAX

... enter the contents of the $i^{th}$, $j^{th}$, and $z^{th}$ cell location for array m...

10 CONTINUE
```

All entries must be separated by a blank, and data may be entered in all columns 1 through 80. Entries of the form, "L*N," means enter the value N into the input L times. This step could also be done with the "R" option by entering "LRN." In both cases blanks between entries are not allowed.

Example: 2 1 2 2 1 2 2 1 2

This could be the description of a 3 × 3 array of the form,

2 1 2

2 1 2

2 1 2

IOP = 1 - Mixed-Cell Orientation Cards

The first field contains the entry followed by three sets of three fields that are treated like FORTRAN DO loops, followed by a field that indicates whether another set of data is to be read. The arrangement of lattice cells may be considered as consisting of a three-dimensional (3-D) matrix of numbers, with the cell position increasing in the positive X, Y, and Z directions, respectively. Each set of orientation data consists of the following parameters, separated by one or more blanks.

LTYPE The cell entry. LTYPE may be negative (array #), zero (empty cell), or positive (universe #).

IX1 The starting point in the X direction. IX1 must be at least 1 and ≤NXMAX.

IX2 The ending point in the X direction. IX2 must be at least 1 and ≤NXMAX.

INCX The number of cells by which increments are made in the positive X direction. INCX must be >0 and ≤NXMAX.

IY1 The starting point in the Y direction. IY1 must be at least 1 and ≤NYMAX.

IY2 The ending point in the Y direction. IY2 must be at least 1 and ≤NYMAX.

INCY The number of cells by which increments are made in the positive Y direction. INCY must be >0 and ≤NYMAX.

IZ1 The starting point in the Z direction. IZ1 must be at least 1 and ≤NZMAX.

IZ2 The starting point in the Z direction. IZ2 must be at least 1 and ≤NZMAX.

INCZ The number of cells by which increments are made in the positive Z direction. INCZ must be >0 and ≤NZMAX.

ISTP = 0, read another set of data,
≠ 0, do not read any more mixed-cell orientation data.

An important feature of this type of data description is that, if any portion of an array is defined in a conflicting manner, the last card to define that portion will be the one that determines the array's cell type configuration. To utilize this feature, one can fill an entire array with the most prevalent cell type and then superimpose the other cell types in their proper places to accurately describe the array. The last set of mixed-cell orientation data must have a nonzero entry in the last field.

IOP = 2 - Standard FIDO Array Input

The array being entered is integer; therefore, it is a "$" or "$$" array. The array may be entered in the standard free-form FIDO format. The description for each lattice array is entered as a single array block with FIDO. The FIDO integer array number entered is the array number being described plus 100. The data are entered, and each array description is terminated with a 'T.' All standard FIDO repeat options are available for entering the data. Array 1 would be entered as the "101$$" FIDO array terminated with a "T." The process would continue until all array descriptions have been entered. The format for the data entry is the same as the description for free-form input. All x entries for the first y row and first z level are entered, then all x entries for the second y row and first z level are entered. This process continues until the entire first z level has been described. Then the second z level is described until the entire array has been described. Then the geometry array description or a given array is terminated with a "T."

A.10 Universe Type

One entry is required for each universe modeled in the combinatorial geometry, starting with Universe 1. The entries should be either a zero or a 1: a zero if the universe is "combinatorial" and a 1 if the universe is "simple." A "simple" universe is a universe composed of concentric zones, where every zone completely surrounds the zone inside of it. Furthermore, input zones in a simple universe may be only one code zone and may be described by only one or two bodies. Tracking through "simple" universes is about 30% or more faster than through regular combinatorial geometry tracking, although the modeling capability is limited. Simple universes may be combined with regular combinatorial universes in arrays without any problems.

--- End of Geometry Input ---

F9.A.4 MORSE DATA BLOCK 3 (FIDO INPUT)

6* (Ten entries)

1. TMAX Maximum cpu time in minutes allowed for the problem on the computer.

2. TCUT Age in seconds at which particles are retired; if TCUT = 0, no time kill is performed.

3. WTSTRT Input starting weight for source particles (set = 1.0 if read as zero).

4. AGSTRT Starting age for source particles.

5. XMIN

6. XMAX

7. YMIN

SOURCE SAMPLING VOLUME BOUNDARIES (absolute coordinates). Source coordinates are sampled uniformly within the bounds input by user. If the user has specified a source media, MSM, then only points in the specified

8. YMAX media will be accepted.

9. ZMIN
10. ZMAX

7* (NNGA)

1. (VELTH(I),I=1,NNGA) The neutron velocities for all energy groups being analyzed. If not input, calculated values will be used based on the group energy limits.

8* (4*MAXGP*MXREG) required only if NSPLT+NKILL+NPAST > 0

1. ((WTHI(I,J),I=1,MAXGP),J=1,MXREG) weight above which splitting will occur.
2. ((WTLO(I,J),I=1,MAXGP),J=1,MXREG) weight below which Russian roulette is played.
3. ((WTAV(I,J),I=1,MAXGP),J=1,MXREG) weight parameters for particles surviving Russian roulette.
4. ((PATH(I,J),I=1,MAXGP),J=1,MXREG) pathlength stretching parameters for use in exponential transform (usually $0 \leq PATH < 1$)

9* ((NNGA+NGGA)*MXREG) required only if IEBIAS > 0.

1. ((EPROB(I,J),I=1,NNGA+NGGA,J=1,MXREG) values of the relative energy importance of particles leaving a collision in region J going to group I.

10* (MXREG+((NNGA+NGGA)*MEDIA) required only if MFISTP > 0. Initialized values will be used if not input.

1. (FWLO(I),I=1,MXREG) values of the weight to be assigned to fission neutrons. (Initialized to 1.0 if not input.)
2. ((FSE(I,J),I=1,NNGA+NGGA,J=1,MEDIA) fraction of fission-induced source particles in group I for medium J. (Uses fission spectrum of element with largest $\nu\Sigma_f$ if not input.)

11* NNGA*MXREG if IADJM ≤ 0 required only if coupled neutron-gamma
or
NGGA*MXREG if IADJM > 0 ray problem, but not completely coupled.

1. ((GWLO(I,J),I=1,NNGA or NGGA),J=1,MXREG) probability of generating a gamma at each collision. Alternatively (depending on GPROB), the values of the weight to be assigned to the secondary particles being generated. NNGA groups are read for each region in a forward problem and NGGA for an adjoint.

12$ (NMIX)

1. (KM(I),I=1,NMIX) MORSE media number.

13$ (NMIX)

1. (KM(I),I=1,NMIX) element or ICE identifiers.

14* (NMIX) (14*Array not input if using an ICE tape.)

1. (RHO(I),I=1,NMIX) density.

15$ NOT USED

16$ NOT USED

17* (NGPFS) required only if ISOUR ≤ 0

1. (FS(I),I=1,NGPFS) the unnormalized fraction of source particles in each group. Uses FSE of media number 1 if not input and MFISTP > 0.

18* (NGPFS) required only if ISOUR ≤ 0 and ISBIAS ≠ 0.

1. (BFS(I),I=1,NGPFS) the relative importance of a source in group 1.

T (Block 3 must be terminated with a T.)

F9.A.5 MORSE DATA BLOCK 4 (ANALYSIS DATA)

Card AA (20A4)

Title or units for total response for all detectors.

Card AB (20A4*NRESP)

NRESP* title or units for each total response for all detectors.

Card AC (20A4) required only if NE > 1

Units for energy-dependent fluence for all detectors.

Card AD (20A4) required only if NT > 1

Units for time-dependent total response for all detectors.

Card AE (20A4) required only if NT > 1 and NE > 1

Units for time- and energy-dependent fluence for all detectors.

Card AF (20A4) required only if NA > 1

Units for angle- and energy-dependent fluence for all detectors.

F9.A.6 MORSE DATA BLOCK 5 (FIDO INPUT)

19* (ND*3)

((X(I),Y(I),Z(I)),I=1,ND) (X,Y,Z) coordinates of i^{th} detector.

20* ((NNGA+NGGA)*NRESP)

((RFV(I,J),I=1,NNGA+NGGA),J=1,NRESP) response function values.

21$ (NE) required only if NE > 1

Energy group numbers defining the lower limit of energy bins.

22* (NT*ND) required only if NT > 1

((T(I,J),I=1,NT),J=1,ND)

NT values of upper limits of time bins for each detector (in order of increasing time and detector number.)

23* (NA) required only if NA > 1

Values of upper limits of angle (cosine) bins.

T (Block 5 must be terminated with a T.)

F9.B MORSE-SGC SUBROUTINE DESCRIPTION

ABEND is called when an abnormal termination is required. It calls RCOVER and then returns.

ACCNRM is a utility routine to normalize an N by M matrix. It will generate cumulative probability tables for each N row of the matrix.

ADJNT is called to invert many different arrays for an adjoint calculation. The arrays inverted depend on user input and problem description. ADJNT calls a utility routine INVERT to perform the inversion.

ALBDO is called when an albedo media (MEDALB) has been encountered. It is called from SUPGRP and performs a simple specular reflection using UNORM, VNORM, and WNORM (direction cosines normal to the reflecting surface) calculated by NORML which was called by GOMST.

ALOCAT is called to dynamically allocate core memory for data storage.

ANGLES is called to compute the angular-scattering probability tables, that is, the probability of scattering through a given angle for a given energy change. First the moments are computed from the Legendre expansion of the scattering cross sections, then the moments are used to compute tables of angles and probabilities. ANGLES is called for each group transfer for each material by XSEC5.

ARCT is called from SUPGRP to process collisions for the array editing. The collision location criteria and media criteria are tested in ARCT. Edit responses are computed in ARCT.

AZIRN returns the sine and cosine of a random angle uniformly distributed between 0 and 360 degrees.

BADMOM is called from ANGLES to print out bad moments, and associated error messages. A moment is rejected if it implies negativity in the angular distribution or if it falls outside the acceptable range.

BATCH controls the program flow from the beginning of a batch through the random walk process (by calling SUPGRP) until the end of batch processing. It terminates a calculation when all batches are completed or when a user-specified cpu time has been used.

BDRYX is called from NXTCOL as an input option. It computes the flux at an input zone boundary. The subroutine must be user-written and will be called (optionally) at media boundaries or region boundaries corresponding to input zone boundaries.

COLISN is called from SUPGRP at collision sites to determine the particle's new emerging energy, weight, and direction cosines. If energy biasing is being used, it calls GTIOUT to determine the new energy group and correct the particle weight for the biasing. The new direction cosines are computed by performing two 2-D rotations on the old direction cosines. The first rotation is through the scattering angle (FM). The second rotation is a random azimuthal rotation in the scattering cone.

DIREC is called to determine the amount of pathlength stretching when the exponential transform option is invoked. The version of DIREC in MORSE-SGC offers many options for direction biasing. This feature reduces the necessity of the user supplying his own version of DIREC for many cases.

ENDRUN is called by NRUN to provide special problem-dependent output. The default version is a dummy routine.

EUCLID is called from SGAM, SDATA, and RELCOL to track from a point source or collision location to a point detector. EUCLID calls PILOT and NSIGTA to determine the number of mean-free paths between the two locations. PILOT returns to EUCLID on input zone boundaries.

EXPRN returns an exponential random number between zero and positive infinity.

FBANK is the fission bank routine called by FPROB. The direction cosines and energy group of the fission neutron are determined in FBANK. STORNT is called to store the new parameters in the appropriate arrays for later retrieval.

FCOLN is optionally called by SUPGRP for rotated coordinate tracking. FCOLN calls CLEV to obtain the particle direction cosines in the nonrotated coordinate system before calling COLISN. This step is not necessary except to benchmark comparisons between runs using a rotated coordinate system with runs using nonrotated coordinate systems.

FIDAS is a standard ORNL array input subroutine. It calls FFREAD to read free-form input for either floating-point or integer data.

FIND is called from ANGLES to find the roots to the orthogonal polynomials between - 1 and + 1. FIND calls Q to evaluate the polynomials.

FISGEN returns $\nu\Sigma_f/\Sigma_T$ for group IG and media IMED.

FLTRN calls assembler random-number generator and returns a random number between zero and 1.

FLUXST banks the flux estimates made by RELCOL, SDATA, SGAM, and BDRYX. The estimate is multiplied by user input response functions.

FPROB is called from SUPGRP in criticality problems at collision sites in fissile media to determine if a fission will take place. FBANK is called by FPROB to bank the fission data. Fission counters and fission weight counters are incremented in FPROB.

FSOUR is called from MSOUR to move particles from the fission bank arrays into the source bank array prior to starting a batch. FSOUR retrieves the fission data directly from the arrays and calls STORNT to store the data into the source data arrays for retrieval by GETNT.

GAMGEN is called from GPROB. It returns the probability of generating a secondary particle in group IG and media IMED. It determines the energy of the secondary particle by selecting a random number and comparing it to the neutron-to-gamma transfer cumulative probability table.

GDATE is called by INITL and by NRUN to get the current date and time.

GETETA is called from NXTCOL to determine the extent of pathlength stretching. It uses function DIREC to compute the BIAS factor. This factor is used to compute the extent of path stretching and the amount of weight correction to apply. If the NOLEAK option for pathlength stretching is invoked, then EUCLID is called.

GETMUS is called by ANGLES to compute the angular moments from the Legendre coefficients. These values are used in the recurrence relations for the orthogonal polynomials. Further detail on the derivation of the methods used in MORSE to compute the moments, orthogonal polynomial coefficients, angles and probabilities may be found in Sect. F9.E.2.

GETNT is called to retrieve particle coordinates, direction cosines, and other variables such as weight, energy group, etc., from a particle bank. All data stored in the particle banks are in global coordinates. Local coordinates are returned along with the particle nesting table upon retrieval. CALI is called to locate the particle and return the local coordinates. STORNT is an entry point in GETNT. It performs the inverse function of storing the data in the particle bank. STORNT calls CLEV to generate global coordinates from local coordinates and the particle's nesting table. VAR3 is called to compute the fractional standard deviation on each edit. The final results are printed out by GIFINA for each edit.

GIFINA is called from BATCH for the end-of-run processing.

GTMED is called by COLISN, FBANK, FPROB, GAMGEN, NSIGTA, PTHETA, and SUPGRP to select the cross-section media number corresponding to a particular geometry media number. The default version assumes the two are the same.

GOMST is called by NXTCOL to follow the particle track to either the next boundary crossing or to the next collision. GOMST is the main interface subroutine between MORSE's random walk and the MARS geometry system in MORSE. It calls CALI after an albedo reflection. It calls PILOT for the particle track. It calls NORML when an albedo media (MEDALB) has been encountered.

GPROB is called from SUPGRP when a coupled neutron-secondary-gamma problem is being run. It calls GAMGEN and compares the probability of secondary generation with the GWLOW array to determine if a secondary particle will be generated by the collision.

GSTORE is called from GPROB to store the secondary gammas. It determines if room exists in the bank to store secondaries. It saves the particle coordinates and direction cosines in common NUTRON. If the secondary is produced inside an array, GSTORE only has the local coordinate of the particle. The coordinate of the particle in the global level of the geometry model is required for banking. This is obtained by calling CLEV, to convert from a local to a global coordinate. STORNT is called for the data storage, then label common NUTRON is restored. Secondary counters are incremented before the routine returns.

GTIOUT is called from COLISN at a collision site where energy biasing is being performed. GTIOUT determines the outgoing energy from the collision and the corrected weight after biasing. Energy biasing is important for fully coupled problems, where the nonabsorption probability becomes greater than 1.

GTISO returns random isotropic direction cosines U, V, and W.

IASN is called from BDRYX for SAS4 cases to calculate the azimuthal detector number.

INITL is called by program O0O006 or O0O106, or MORSE, depending on which setup of SGC is being used. It does some initialization of I/O units, reads the first block of data, and calls ALOCAT, which calls MPROG.

INSCOR is called by SCORIN to provide special problem-dependent input data. The default version is a dummy routine.

INVERT inverts a 1-D array.

INVRT1 inverts a 2-D array.

INVRT2 is called by XSEC4 to invert 1-D cross-section arrays, where the beginning of each group is determined by an array of pointers.

JOMINB is not used in this version of MORSE. It will read the binary geometry file written by JOMINI from logical unit ND30 and copy it to logical unit ND17.

MAMENT is called by BADMOM to generate Legendre coefficients from moments. The routine requires label common MOMENT.

MGCWRD is called from MWLIST, XSEC2, and XSEC3 to unpack the magic word integer used in the AMPX cross sections. The purpose of the magic word is to determine the starting and ending location of the group transfer arrays to eliminate requiring a square transfer matrix with zero fill. The AMPX magic word saves considerable computer memory, since the scattering matrix does not have to be full.

MORSE is the main program when running SGC stand-alone.

MPROG is a subroutine called by INITL. It calls BATCH. This subroutine is the controlling routine for the MORSE/SGC execution.

MSOUR is called from BATCH at the beginning of a particle batch. It calls FSOUR to obtain the fission bank for criticality problems. It calls SOURCE for fixed-source problems. It calls GETNT and STORNT to update the neutron particle bank.

MTIMER is a utility routine maintaining local and global clocks for MORSE during execution.

MWLIST is called from XSEC2. It prints out the transfer matrices in a variable format.

NDBARC is called from BATCH for the end of batch processing. NDBARC updates the array edits at the end of each batch and normalizes the result. The mean and the square of each batch estimate is computed in NDBARC.

NDBTCH is called from BATCH for the end of batch processing. It adds the average responses for each point detector at the end of a batch to the results for the run. The square and the result are tabulated for later computation of the mean and the fractional standard deviation at the end of the run.

NETLEV is called from BDRYX and NXTCOL by option when MORSE is being run as part of SAS4. NETLEV converts x, y, z, u, v, and w between local and global coordinate systems.

NORML is called by GOMST to compute the normal direction to the surface of an albedo media. It obtains the body number, NASC, from the combinatorial common PAREM. The body encountered must be referenced in the albedo zone description or the code will print a fatal error. The body surface number is LSURF. This routine is the only one in MORSE using LSURF.

NRUN is called from MPROG to perform analysis processing at the completion of the run. It works with the arrays prepared by NDBTCH. VAR2 and VAR3 are called to compute the fractional standard deviations. ne point detector results and the fluence results for the point detectors are printed out by NRUN.

NSIGTA is called from NXTCOL and EUCLID to look up the total cross section for a given energy group and media. The media is checked to verify its validity. The total cross section for an internal void is returned as zero.

NXTCOL is called from SUPGRP to track a particle to its next collision site. It calls GOMST to control the tracking. It returns when the particle has had a collision, leaked from the system, or encountered an albedo boundary. It will optionally call BDRYX for boundary crossing estimates.

OUTPT is called by BATCH at the end of a batch and by MPROG at the end of a run. It prints out the summary information for the collisions, boundary crossings, splittings, leaks, etc., for the end of a batch. At the end of a run it prints out the neutron death table and the region counters; scattering, fission, Russian roulette kills, survivals, etc. The region counters and the Russian roulette and splitting arrays require considerable memory, especially when many regions are modeled. K-effective for criticality runs is computed and printed out by OUTPT.

OUTPT2 is called from OUTPT to print out 2-D arrays.

PRT1D is called from XSEC2 to print out 1-D cross sections.

PRT1D2 is called from XSEC7 to print out 2-D cross-section arrays.

PTHETA is called from RELCOL and RELCOA to determine the probability of scattering through angle 0 for each of the possible energy transfers. All possible downscatters and upscatters are computed.

PTPT is called from RELCOL, SDATA, and SGAM to compute the distance from the collision site to the point detector. The point detector location is in global coordinates, while the collision location is in local coordinates. CLEV is called by PTPT to compute the global coordinate of the collision site.

Q a function called from ANGLES, BADMOM, and FIND to compute the value of an orthogonal polynomial at a position X.

O0O006 and O0O106 are the two entry points to initiate MORSE execution. The entry point taken determines whether the input is read from cards or from binary files in SCALE.

RANNO is the assembler random number generator.

RDU96 is called by O0O106 to read unit 96 when a SAS4 case is being run.

RELCOL is called from SUPGRP at a collision site. It calculates collided flux estimates to the point detectors by calling PTHETA, EUCLID, and FLUXST. RELCOL makes only one estimate to each point detector from each collision site, as opposed to RELCOA which makes multiple estimates.

RELCOA is used to make estimates to point-detector sites from each collision site. It calls PTHETA, EUCLID, PTPT, and FLUXST. Estimates are made to each possible group transfer given the fixed angle of scatter to travel to a point detector. Estimates for gamma groups are done using the Klein-Nishina formula if NCOEF < 0.

RESET is called to reset array tracking variables destroyed by point-detector estimation. During tracking, estimates may be made from collision sites to point detectors. After the collided flux estimate has been made, the random walk is continued. It is necessary to reset the particle nesting table to its previous condition at the collision site before continuing the particle track.

RINPU1 is called from MPROG. It prints out most of the MORSE input edit and reads in a large portion of the user's input. When MORSE is running outside of SCALE, RINPU1 calls

FIDAS to read the input. When MORSE is running as a part of SCALE, RINPU1 reads a binary input file by calling QOREAD. ITYPE in labelled common SCALOP determines which way MORSE is running.

RINPU2 is called from MPROG. It computes the array pointers for data storage and the amount of supergrouping required. Many data arrays are shifted around in memory by RINPU2 to their final locations. XSEC3, XSEC4, XSEC5, XSEC6, and XSEC8 are called by RINPU2.

RITNUT is a utility routine, which may be called to dump out the contents of labelled common NUTRON containing particle tracking information, coordinate, weight, direction cosines, etc.

RNDIN loads a seed into the random-number generator RANNO.

RNDOUT gives the last random number returned by RANNO, without changing the random-number sequence.

SCORIN is called by RINPU2. It prints out the point-detector coordinates, response functions, energy bank boundaries, and angle bank boundaries related to point-detector scoring.

SDATA is called from MSOUR to compute the uncollided flux estimate to a point detector from a source point. It calls PTPT, EUCLID, and FLUXST. The switch flag in the call to FLUXST is set to add the uncollided flux estimate to the collided flux estimate.

SGAM is called from FBANK and GSTORE to compute the uncollided flux estimate to point detectors from secondary-particle birthsites (captured gamma or fission neutron). SGAM is nearly identical to SDATA except the switch flag in the call to FLUXST is set to score only the total flux.

SHIFT is called to move data from one area of memory to another area of memory.

SHUFLE is called from RINPU2. It computes the multigroup cross-section pointers after the cross-section storage has been supergrouped.

SIZEX is called from RINPU2. It computes the space required for cross-section storage. This calculation determines the amount of supergrouping and the size of each supergroup. This is a function of available core memory, the number of energy groups, and the number of media.

SORIN is called from RINPU2. It computes the cumulative distribution for source energy spectrum for both forward and adjoint problems. The distributions are normalized, and are ordered depending on whether a forward or adjoint case is being run. The cumulative distribution is increasing to 1 for a forward problem, and decreasing from 1 for an adjoint problem.

SORK sorts the neutron bank by supergroup at the beginning of each supergroup. During a batch, particles from the most populous supergroup will be tracked, then the next most populous supergroup, etc. SORK calls STORNT and GETNT in ordering the neutron bank. SORK is called from BATCH.

SOURCE is called from MSOUR. It calls CALI to determine the location of a source particle. The version of SOURCE in MORSE-SGC samples uniformly from XMIN to XMAX, YMIN to YMAX, and ZMIN to ZMAX. If the user has specified a source media, only those points sampled in the media will be saved as a valid source particle. Angular biasing is provided as an option in SOURCE. Isotropic selection of initial starting cosines is available by default in SOURCE by calling GTISO.

SOURS4 is called from SOURCE when MORSE is being run as part of SAS4. SOURS4 generates source particle parameters for a shipping cask according to the source region geometry option specified.

STBTCH is called by BATCH. It clears the response and detector analysis arrays at the beginning of each batch by calling CLEAR.

STRUN is called from MPROG. The user may supply his own version of STRUN. The version supplied does nothing. It is called at the start of a new run.

SUPER1 is called from RINPU2. This routine calculates the lengths of each of the supergroups stored in memory.

SUPER2 is called from RINPU2. This routine constructs a table giving the supergroup number for each energy group number.

SUPGRP is called from BATCH. This routine controls the random walk for each particle in a supergroup. It calls NXTCOL for particle tracking. It calls routines to handle the secondary particle generation and banking. Russian roulette and splitting is performed in SUPGRP. COLISN is called from SUPGRP for collision processing.

SURFAC is called by NRUN to print the surface detector results to the standard output unit and to unit 75 which is named 'detout'.

TRNPSE is called from XSEC4 to transpose an array.

VAR2 is called to compute the fractional standard deviation of the analysis arrays. VAR2 works on 2-D arrays and can independently weight batches. VAR2 is called from NRUN.

VAR3 is called from NRUN. It is the same as VAR2 except it can work on 3-D arrays.

XNPUT is called from BATCH. This routine retrieves cross-section data from disk for a given supergroup during batch processing.

XSEC1 is called from RINPU1. It reads the ID record and energy group structure from an AMPX working tape.

XSEC2 is called from RINPU1. It reads the cross-section data off the AMPX working tape for elements in the mixing table and stores them on disk. It also calculates the storage space that will be needed to store the mixed cross sections.

XSEC3 is called from RINPU2. It mixes the cross sections and stores the mixed cross sections on disk. It also selects FSE from cross-section data if it wasn't read in.

XSEC4 is called from RINPU2. This routine will invert the cross sections for an adjoint problem. It sums the neutron to gamma transfer arrays into 1-D gamma production cross sections. It accumulates and normalizes the 2-D cross sections as needed. It inverts FSE array also. Not called if previously mixed cross sections (JXTAPE > 0).

XSEC5 is called from RINPU2. It calculates the angles and probabilities for collision processing. Cumulative angular probability distributions are constructed for each energy transfer. These data are stored by supergroup on disk.

XSEC6 is called from RINPU2 before XSEC5. This routine reads cross sections off disk, reorders them into supergroups, and stores them back on disk.

XSEC7 is called from RINPU1. It reads a preprocessed, premixed ICE (see Sect. F8) tape, and retrieves the mixtures requested.

XSEC8 is called from RINPU2. It performs similar functions as XSEC5 and XSEC6, except for an ICE premixed tape. It selects pair production and Compton scattering cross sections when the KleinNishina option is being used.

Some of the routines described above are part of other subroutine libraries, such as the SCALE subroutine library or MARSLIB, but are called from MORSE.

In addition to the above routines, dummy versions of the following routines are provided:

ALBEDO, ECUT, ENDRUN, ESCAPE, FISSIN, GAMEST, HELP, HELPER, RRKILL, RRSURV, SPLITN, TKILL, and XSCHLP.

Users may write versions of these routines to do analyses specific to their application.

F9.C MORSE-SGC SAMPLE PROBLEMS

F9.C.1 INTRODUCTION

MORSE-SGC has eight sample problems distributed with the code package. The first three problems are neutron-only shipping cask shielding problems that are intended to demonstrate use of biasing parameters rather than solution of a practical problem; the geometry of the cask is from Ref. 1. Problem 1 was run using only splitting and Russian roulette: problem 2 added the path-length-stretching option; problem 3 uses source-energy-biasing parameters from an adjoint XSDRNPM calculation.

Problems 4 through 7 come from the MORSE-CGA code package (see Ref. 2). They test several additional options of the MORSE-SGC system. Problem 4 (originally CGA sample problem 1) is a calculation of fast neutron fluence for a point, isotropic, fission source in an infinite medium of air. Sample problem 5 (originally CGA sample problem 2) is a calculation of secondary-gamma dose rate due to neutrons of energies greater than 0.011 MeV at several radial distances. It is similar to problem 4 except that gamma rays are transported.

Problem 6 (originally CGA sample problem 6) is a gamma-ray problem (no neutron transport) which calculates gamma-ray dose rate for a point isotropic 4-5 MeV source. Problem 7 (originally sample problem 8 from CGA) is a collision density calculation that uses several types of estimation including boundary crossing, collision density, collision density fluence averaged over regions, and a tracklength/unit-volume estimator.

Problem 8, which illustrates the array capabilities of the MARS geometry package, is nearly identical to problem 3. The only difference is in the modeling of the fuel assemblies. Each fuel rod in the 7 assemblies is explicitly modeled. Each assembly is assumed to consist of an array of 17 × 17 fuel rods without any control rods or water holes.

Descriptions of the eight problems, input data, and output results are contained in the following sections. Problems 4 through 7 have been through a verification process using MORSE-SGC on a UNICOS system on a CRAY computer (see Ref. 3). They have been part of the sample problem package for MORSE-CGA (originally MORSE-CG) for many years, and, as a result, they have been tested on several generations of main-frame computers and multiple versions of MORSE. Results have been checked against discrete-ordinates calculations and compared with other Monte Carlo calculations. Additional information on Problems 4-7 appears in Refs. 2 and 3. Although this set of sample problems tests many of the options in the MORSE code system, it is not all inclusive, nor is it intended to be. The sample problems are representative of the types of problems that can be run. It may be beneficial to add other problems at some future date.

The sample problem results printed in this report are from an IBM RISC-6000 series workstation and are for illustrative purposes only. Users should refer to the sample problem results that are distributed with the SCALE code system because they will be the current results as run with the version being distributed.

F9.C.2 PROBLEM DESCRIPTION FOR SHIPPING CASK SHIELDING PROBLEMS

This sample problem is intended more to demonstrate specification of biasing parameters for MORSE-SGC than to solve a practical shielding problem. The problem is to calculate the upper axial neutron leakage from a dry shipping cask. The geometry of the cask was taken from Ref. 1. Basically, it is a cylindrically symmetric model of the proposed IF300 Shipping Cask containing seven PWR fuel assemblies.

A decay time of 120 days was assumed for the fuel. The 22 neutron energy groups used in the analysis are from the 22N-18 COUPLE library in SCALE.

Again, it should be pointed out that biasing of a Monte Carlo calculation is more of an art form than an exact science. Judgment, rules-of-thumb, and luck are all part of "proper biasing."

The first step involves the specification of the importance region geometry. The user should try to define these regions in such a manner that particles of equal importance are grouped together. This means that the importance regions are determined by the detector geometry and not by the source geometry. An additional consideration is the detail one uses in defining these regions. Many small regions will increase computation time because geometry tracking, the major consumer of computation time in Monte Carlo calculations, will increase as the number of bodies in the geometric description increases. A few large regions, however, may also increase computation time because of the lack of detail in the biasing information. A rough rule of thumb is to make importance regions around one mean-free path in thickness.

Using material boundaries for importance region boundaries does not increase computation time because no new bodies are added to the geometric description. Therefore, these boundaries serve as a starting point for region specification. For the upper axial shield, these boundaries result in a rather large importance region for the depleted uranium. Therefore, this region was broken down into three regions, each with an optical thickness of about one mean-free path. In addition, the outer stainless steel was divided into two regions. The source was assigned to region 1. The material to the side of the source and below the axial shield was designated region 8. The resulting region description is shown in Fig. F9.C.1.

The weight standards, based on the guidelines for Russian roulette and splitting (see Sect. F9.3. 1), are shown in Table F9.C.1. These weight standards are only appropriate if no source biasing is used. Region 8, not covered by the guideline in Sect. F9.3.1, is less important than the source region because leakage from the side of the cask is more probable. Therefore, WTAV should be higher for region 8 than for region 1. Arbitrarily, WTAV was assigned a value for region 8 that was twice that of the source region.

F9.C.3 RESULTS FOR SAMPLE PROBLEMS 1-3

This problem was run with MORSE-SGC using the weights of Table F9.C.1. Russian roulette and splitting were the only biasing techniques used in the first calculation. The input is shown in Fig. F9.C.2. The calculated upper axial leakage is 0.0261 neutron/source neutron. The calculated standard deviation for the 20,000 history run is 3.74% of the mean (i.e., 3.74% of 0.0261—considerably less than the 20% used as a figure of merit for a "meaningful" calculation). A large number of contributions, 2437, were made to the detector. The latter number is determined by multiplying detector 2 results by the number of source particles (NSTRT*NITS) in the run. A "last-event" or "death" estimator was used, so the problem of underbiasing was avoided.

This problem was also solved using the guidelines for path-length stretching (see Sect. F9.3.2). DIREC was set to the cosine of the angle between the particle direction and the Z-axis by setting NPAST of the 1$ array to 1 and NDSG of the 3$ array to 3. The input is shown in Fig. F9.C.3. An upper axial leakage of 0.0266 was obtained with a fractional standard deviation of 3.8% for 20,000 histories. The number of contributions made to the detector was 2682.

A third calculation, using source energy biasing parameters obtained from an adjoint XSDRNPM calculation was also performed. The XSDRNPM input is shown in Fig. F9.C.4. The resulting adjoint fluxes are shown in Table F9.C.2.

In order to do a source energy biasing calculation with Russian roulette and splitting, the weight standards must be modified. The weights in Table F9.C.1 reflect the spatial dependence of the importance

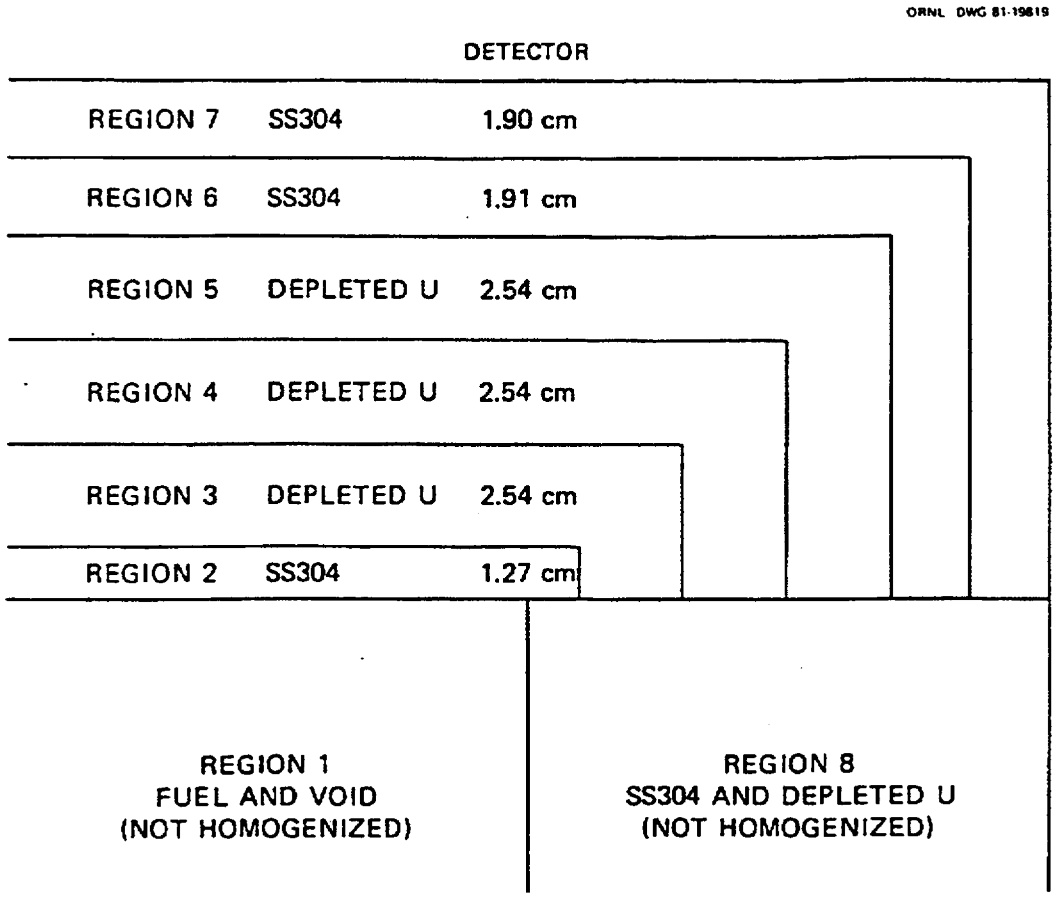


Figure F9.C.1 Sample problem description

Table F9.C.1 Weight standards

Region	WTAV	WTHI	WTLO
1	1.0	5.0	0.2
2	0.71	3.55	0.142
3	0.45	2.25	0.090
4	0.24	1.20	0.048
5	0.13	0.65	0.026
6	0.078	0.39	0.0156
7	0.054	0.27	0.0108
8	2.0	10.0	0.4

```
=nitawl
0$$ 85 e  1$$ a2 11 e 1t
2$$ 8016 24000 25055 26000 28000 40000 42000 92235 92238 94239 94240 2t
end
=morse
morse-sgc/s sample problem #1
    13579bdfdb97
1$$ 0 100 800 200 1 0 0 1 1 0 0 0 0 0
2$$ 3 13 88 8 1
3$$ 22 5r0 0 4 2 22 0 1 4r0 4 0 0 1
4$$ 0 14 0 0
5$$ 2 0 0 0 0 1 0 2 11r0 1 8r0
  t
2d dry cask model
 0 0 0 0
 rcc 0 0 -1 0 0 183.88 12.1
 rcc 0 0 -1 0 0 184.88 21.16
 rcc 0 0 -1 0 0 183.88 36.42
 rcc 0 0 -1 0 0 229.60 44.55
 rcc 0 0 -1 0 0 230.87 45.82
 rcc 0 0 -1 0 0 233.41 48.36
 rcc 0 0 -1 0 0 235.95 50.90
 rcc 0 0 -1 0 0 238.49 53.44
 rcc 0 0 -1 0 0 242.30 57.25
 rcc 0 0 -2 0 0 2.0000 60.00
 rcc 0 0 -1 0 0 240.39 55.34
 rcc 0 0 -1 0 0 229.50 58.00
   end
 sfl 1 -10 or 3 -2 -10
 inv 4 -3 -10 or 2 -1 -10
 ss1 5 -4 -10 -12
 du1 6 -5 -10 -12
 du2 7 -6 -10 -12
 du3 8 -7 -10 -12
 ss2 9 -11 -10 -12
 alb 10 9
 exv -9
 ss3 11 -8 -10 -12
 ss4 5 12 -4 -10
 du4 8 12 -5 -10
 ss5 9 12 -8 -10
   end
 1 1 2 3 4 5 6 1 1 7 8 8 8
 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 1000 2 3 3 3 2 88 0 2 2 3 2
 0
6** 10.0 3r0.0 -36.42 36.42 -36.42 36.42 0.0 182.88
8** 22r5.0 22r3.55 22r2.25 22r1.2 22r.65 22r.39 22r.27 22r10.
    22r.2 22r.142 22r.09 22r.048 22r.026 22r.0156 22r.0108 22r.4
    22r1. 22r.71 22r.45 22r.24 22r.13 22r.078 22r.054 22r2.
    176r0.
10** 1. .71 .45 .24 .13 .078 .054 2.
 4.9338-4 1.9473-3 5.8223-3 1.9193-2 3.9752-2 5.1325-2 1.0834-1
 8.7299-2 2.1106-2 1.1541-1 2.0886-1 1.9255-1 1.3361-1 1.4261-2
 2.1043-5 1.5248-6 1.0053-7 5r0.0
 22r0.0
 3.3607-4 1.4566-3 4.7086-3 1.6604-2 3.6095-2 4.8189-2 1.0487-1
 8.6328-2 2.0664-2 1.1546-1 2.1259-1 1.9902-1 1.3943-1 1.4248-2
 8r0.0
12$$ 6r1 5r2 2r3
 13$$ 92235 92238 94239 94240 8016 40000 24000 25055 26000 28000
 42000 92235 92238
14** 5.2666-5 6.1687-3 3.2769-5 1.4961-5 1.2538-2 1.7929-3
 1.662-2 1.2-3 5.775-2 7.52-3 1.1-4 1.07-4 4.77-2
17** 8.611-5 7.295-4 2.007-3 1.009-2 2.516-2 5.07-2 .2123 .1581
  .03157 .1331 .1737 .1386 .0636 6.131-6
  t
 neutron leakage
 neutron leakage
19** f0.0
20** f1.0
  t
end
```

Figure F9.C.2 MORSE-SGC sample problem 1 input – splitting and Russian roulette

```
=nitawl
0$$ 85 e  1$$ a2 11 e 1t
2$$ 8016 24000 25055 26000 28000 40000 42000 92235 92238 94239 94240 2t
end
=morse
morse-sgc/s sample problem #2
    13579bdfdb97
1$$ 0 100 800 200 1 0 0 1 1 1 0 0 0 0
2$$ 3 13 88 8 1
3$$ 22 5r0 3 4 2 22 0 1 4r0 4 0 0 1
4$$ 0 14 0 0
5$$ 2 0 0 0 0 1 0 2 11r0 1 8r0
  t
2d dry cask model
 0 0 0 0
 rcc 0 0 -1 0 0 183.88 12.1
 rcc 0 0 -1 0 0 184.88 21.16
 rcc 0 0 -1 0 0 183.88 36.42
 rcc 0 0 -1 0 0 229.60 44.55
 rcc 0 0 -1 0 0 230.87 45.82
 rcc 0 0 -1 0 0 233.41 48.36
 rcc 0 0 -1 0 0 235.95 50.90
 rcc 0 0 -1 0 0 238.49 53.44
 rcc 0 0 -1 0 0 242.30 57.25
 rcc 0 0 -2 0 0 2.0000 60.00
 rcc 0 0 -1 0 0 240.39 55.34
 rcc 0 0 -1 0 0 229.50 58.00
   end
 sfl 1 -10 or 3 -2 -10
 inv 4 -3 -10 or 2 -1 -10
 ss1 5 -4 -10 -12
 du1 6 -5 -10 -12
 du2 7 -6 -10 -12
 du3 8 -7 -10 -12
 ss2 9 -11 -10 -12
 alb 10 9
 exv -9
 ss3 11 -8 -10 -12
 ss4 5 12 -4 -10
 du4 8 12 -5 -10
 ss5 9 12 -8 -10
   end
 1 1 2 3 4 5 6 1 1 7 8 8 8
 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 1000 2 3 3 3 2 88 0 2 2 3 2
 0
6** 10.0 3r0.0 -36.42 36.42 -36.42 36.42 0.0 182.88
8** 22r5.0 22r3.55 22r2.25 22r1.2 22r.65 22r.39 22r.27 22r10.
    22r.2 22r.142 22r.09 22r.048 22r.026 22r.0156 22r.0108 22r.4
    22r1. 22r.71 22r.45 22r.24 22r.13 22r.078 22r.054 22r2.
    11r.5 11r0. 11r.5 11r0. 11r.5 11r0. 11r.5 11r0. 44r0.
     11r.5 11r0. 11r.5 11r0.
10** 1. .71 .45 .24 .13 .078 .054 2.
 4.9338-4 1.9473-3 5.8223-3 1.9193-2 3.9752-2 5.1325-2 1.0834-1
 8.7299-2 2.1106-2 1.1541-1 2.0886-1 1.9255-1 1.3361-1 1.4261-2
 2.1043-5 1.5248-6 1.0053-7 5r0.0
 22r0.0
 3.3607-4 1.4566-3 4.7086-3 1.6604-2 3.6095-2 4.8189-2 1.0487-1
 8.6328-2 2.0664-2 1.1546-1 2.1259-1 1.9902-1 1.3943-1 1.4248-2
 8r0.0
12$$ 6r1 5r2 2r3
13$$ 92235 92238 94239 94240 8016 40000 24000 25055 26000 28000
 42000 92235 92238
14** 5.2666-5 6.1687-3 3.2769-5 1.4961-5 1.2538-2 1.7929-3
 1.662-2 1.2-3 5.775-2 7.52-3 1.1-4 1.07-4 4.77-2
17** 8.611-5 7.295-4 2.007-3 1.009-2 2.516-2 5.07-2 .2123 .1581
  .03157 .1331 .1737 .1386 .0636 6.131-6
  t
 neutron leakage
 neutron leakage
19** f0.0
20** f1.0
  t
end
=xsdrn
adjoint
0$$ a3 4 e
1$$ 1 6 74 1 0 5 17 8 3 0 10 4 0 1 3
3$$ 0 1 e
5** a3 0 e
 t
```

Figure F9.C.3 MORSE-SGC sample problem 2 input – path-length stretching

```
13$$ 7r1 2r2 5r3 2r4 5
14$$ 92235 92238 94239 94240 8016 40000 1001 1001 8016
  24000 25055 26000 28000 42000 92235 92238 92238
15** 2.89622-5 3.39277-3 1.80229-5 8.2285-6 0.68958-2 9.86078-4
  0.0 0.0 0.0 .01662 .0012 .05775 .00752 .00011
  1.07-4 4.77-2 0
  t
30$$ 73r0 1
31** 22r1. f0
 t
33** f0 t
35** 39i 0 14i 182.88 228.6 14i 229.87 1i 237.49 241.3 242.3
36$$ 40r1 15r2 3 15r4 2r5 6
39$$ 1 2 3 4 3 5
40$$ f3
  t
end
```

Figure F9.C.4 XSDRNPM input to obtain BFS parameters for MORSE-SGC

Table F9.C.2 XSDRNPM adjoint fluxes for source region

Energy group	Total flux
1	106.9
2	96.99
3	93.24
4	78.56
5	64.03
6	58.65
7	56.01
8	53.86
9	53.43
10	48.61
11	38.65
12	33.14
13	25.26
14	8.891

function. The source energy biasing parameters reflect the energy variation of the importance function in the source region. By multiplying the weight standards in each region by the source weights for the source region, the user is implementing an importance function that is separable in energy and position. Such an importance function is not optimal, but should improve the calculation over using only source energy biasing or only Russian roulette and splitting.

The actual calculation of the weight standards is tedious. It involves first calculating the weights for the source region by multiplying BFS(IG) (i.e., the adjoint fluxes from Table F9.C.2) by the natural distribution [i.e., FS(IG) of the 17* array]. These weights are summed over energy, and the sum is divided by the adjoint fluxes to obtain WTAV(IG) for the source region. All other weights are then multiplied by WTAV(IG). The routine shown in Fig. F9.C.5 performs this operation.

The input for this calculation is shown in Fig. F9.C.6. An upper axial leakage of 0.0253 was obtained with a fractional standard deviation of 4.46% for 20,000 histories. Fourteen-hundred-eighty-seven contributions were made to the detector.

All three problems require the use of the special SUBROUTINE ESCAPE which is provided. The number of contributions mentioned above is determined by multiplying the total number of source particles by the result in the detector 2 position of the response which is a tabulation of the fraction of source particles that contribute to the detector.

F9.C.4 SAMPLE PROBLEM 4: CGA1 (NEUTRON ONLY)

The fast-neutron fluence at several radial distances is calculated by MORSE-SGC for a point, isotropic, fission source in an infinite medium of air. The air was assumed to be made up of only oxygen and nitrogen with a total density of 1.29 g/ℓ. The MARS array geometry package was used to describe the concentric spherical shells of air surrounding the point source. Although the entire medium was air, the geometry medium numbers alternate between each of the shells for use with the boundary-crossing estimator (see part 4, Sect. 4.5.3, p. 4.5-28 of Ref. 3). This estimator requires that each detector lie on a boundary separating two media. The cross sections for air used in this calculation were processed with LAVA (see Ref. 4) and were for 22 neutron groups with 5 Legendre coefficients used for the angular expansion. Only the top 13 neutron groups were analyzed. The group structure with the corresponding fraction of particles emitted in each group is given in Table F9.C.3. Splitting, Russian roulette, and path-length stretching were also implemented. Figure F9.C.7 contains the SGC input data.

For this problem, the standard SOURCE was modified to set WATE = DDF and versions of routines DIREC, GTMED, SDATA, and BDRYX were converted from the corresponding CGA routines. SDATA is a routine for analysis of uncollided fluence, and BDRYX is for analysis of all boundary crossings (equivalent to path length/unit volume) (see part 4, Sect. 4.6.4 of Ref. 3). Figure F9.C.8 is a listing of the modified routines. CGA1 sample problem results from MORSE-SGC compared well with the original MORSE-CGA runs. The detector responses differed about the same amount as they would if you ran a different random number sequence (i.e., statistically, they were equivalent). The random walk produced the same type of events and equivalent numbers of scatterings, Russian roulette kills, energy cut-offs, boundary crossings, etc. Table F9.C.4 shows the results.

```
      subroutine sdata(d,xd,yd,zd,vel)
      common/input/iadj(17),mxreg,mfis(10),maxgp,irds(11),ngpfs
      common/point/lfp(20)
       dimension d(1)
       data icall/0/
       if(icall.ne.0) return
       do 10 ig=1,ngpfs
       ifs=lfp(8)+ig-1
       write(6,1000) ig,d(ifs)
       do 10 nreg=1,mxreg
       iwthi=lfp(16)+maxgp*(nreg-1)+ig-1
       d(iwthi)=d(iwthi)*d(ifs)
       iwtlo=iwthi+maxgp*mxreg
       d(iwtlo)=d(iwtlo)*d(ifs)
       iwtav=iwtlo+maxgp*mxreg
       d(iwtav)=d(iwtav)*d(ifs)
 10    write(6,2000) ig,nreg,d(iwthi),d(iwtlo),d(iwtav)
       icall=1
      return
 1000 format(i10,2e15.5)
 2000 format(2i10,3e15.5)
      end
```

Figure F9.C.5 MORSE-SGC routine to modify weight standards for source energy biasing

```
=nitawl
0$$ 85 e  1$$ a2 11 e 1t
2$$ 8016 24000 25055 26000 28000 40000 42000 92235 92238 94239 94240 2t
end
=morse
morse-sgc/s sample problem #3
    13579bdfdb97
1$$ 0 100 800 200 1 0 0 1 1 1 0 0 0 0
2$$ 3 13 88 8 1
3$$ 22 5r0 3 4 2 22 0 1 4r0 4 0 0 1
4$$ 0 14 1 0
5$$ 2 0 0 0 0 1 0 2 4r0 1 6r0 1 8r0
  t
2d dry cask model
 0 0 0 0
 rcc 0 0 -1 0 0 183.88 12.1
 rcc 0 0 -1 0 0 184.88 21.16
 rcc 0 0 -1 0 0 183.88 36.42
 rcc 0 0 -1 0 0 229.60 44.55
 rcc 0 0 -1 0 0 230.87 45.82
 rcc 0 0 -1 0 0 233.41 48.36
 rcc 0 0 -1 0 0 235.95 50.90
 rcc 0 0 -1 0 0 238.49 53.44
 rcc 0 0 -1 0 0 242.30 57.25
 rcc 0 0 -2 0 0 2.0000 60.00
 rcc 0 0 -1 0 0 240.39 55.34
 rcc 0 0 -1 0 0 229.50 58.00
   end
 sfl 1 -10 or 3 -2 -10
 inv 4 -3 -10 or 2 -1 -10
 ss1 5 -4 -10 -12
 du1 6 -5 -10 -12
 du2 7 -6 -10 -12
 du3 8 -7 -10 -12
 ss2 9 -11 -10 -12
 alb 10 9
 exv -9
 ss3 11 -8 -10 -12
 ss4 5 12 -4 -10
 du4 8 12 -5 -10
 ss5 9 12 -8 -10
   end
 1 1 2 3 4 5 6 1 1 7 8 8 8
 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 1000 2 3 3 3 2 88 0 2 2 3 2
 0
6** 5.0 3r0.0 -36.42 36.42 -36.42 36.42 0.0 182.88
8** 22r5.0 22r3.55 22r2.25 22r1.2 22r.65 22r.39 22r.27 22r10.
     22r.2 22r.142 22r.09 22r.048 22r.026 22r.0156 22r.0108 22r.4
    22r1. 22r.71 22r.45 22r.24 22r.13 22r.078 22r.054 22r2.
    11r.5 11r0. 11r.5 11r0. 11r.5 11r0. 11r.5 11r0. 44r0.
    11r.5 11r0. 11r.5 11r0.
10** 1. .71 .45 .24 .13 .078 .054 2.
 4.9338-4 1.9473-3 5.8223-3 1.9193-2 3.9752-2 5.1325-2 1.0834-1
 8.7299-2 2.1106-2 1.1541-1 2.0886-1 1.9255-1 1.3361-1 1.4261-2
 2.1043-5 1.5248-6 1.0053-7 5r0.0
 22r0.0
 3.3607-4 1.4566-3 4.7086-3 1.6604-2 3.6095-2 4.8189-2 1.0487-1
 8.6328-2 2.0664-2 1.1546-1 2.1259-1 1.9902-1 1.3943-1 1.4248-2
 8r0.0
12$$ 6r1 5r2 2r3
13$$ 92235 92238 94239 94240 8016 40000 24000 25055 26000 28000
 42000 92235 92238
14** 5.2666-5 6.1687-3 3.2769-5 1.4961-5 1.2538-2 1.7929-3
 1.662-2 1.2-3 5.775-2 7.52-3 1.1-4 1.07-4 4.77-2
17** 8.611-5 7.295-4 2.007-3 1.009-2 2.516-2 5.07-2 .2123 .1581
  .03157 .1331 .1737 .1386 .0636 6.131-6
18** 107. 97.0 93.2 78.6 64.0 58.6 56.0 53.9 53.4 48.6 38.6 33.1
     25.3 8.89
  t
 neutron leakage
 neutron leakage
19** f0.0
20** f1.0
  t
end
```

Figure F9.C.6 MORSE sample problem 3 input - source energy biasing, path-length stretching, splitting, and Russian roulette

Table F9.C.3 Fission spectrum in 14-group structure

Group No.	Energy limits (MeV)	Fraction of source neutrons
1	15.0-12.21	1.5579(-4)*
2	12.21-10.0	8.9338(-4)
3	10.0-8.187	3.4786(-3)
4	8.187-6.36	1.3903(-2)
5	6.36-4.966	3.4557(-2)
6	4.966-4.066	3.5047(-2)
7	4.066-3.012	1.0724(-1)
8	3.012-2.466	8.8963(-2)
9	2.466-2.350	2.3186(-2)
10	2.350-1.827	1.2030(-1)
11	1.827-1.108	2.1803(-1)
12	1.108-0.5502	1.9837(-1)
13	0.5502-0.1111	1.4036(-1)
14	0.1111-0.3355	1.5489(-2)

*Read as 1.5579×10^{-4}.

```
=morse
morse sample problem #4 point fission source in air
    343277244615
 1$$ 0  200  400    10   1   0     0   1   1   1 4z
 2$$   1    1    0 1 0
 3$$  13  0 5z  6 3 13 0 1 4z 2  0 2z
 4$$ 0 14 0 0
 5$$  7 4z 1 0 1 4z 1 5z 1 f0 t
  sample prob. #4 for morse
  0  0  0  10
  sph           0.       0.         0.      3.0e+03
  sph           0.       0.         0.      5.0e+03
  sph           0.       0.         0.      7.5e+03
  sph           0.       0.         0.      1.0e+04
  sph           0.       0.         0.      1.5e+04
  sph           0.       0.         0.      2.0e+04
  sph           0.       0.         0.      3.0e+04
  sph           0.       0.         0.      6.0e+04
  sph           0.       0.         0.      7.0e+04
  sph           0.       0.         0.      9.0e+04
  sph           0.       0.         0.      1.2e+05
  sph           0.       0.         0.      1.5e+05
  sph           0.       0.         0.      1.0e+06
  sph           0.       0.         0.      1.0e+07
  end
  air          +1
  air          +2     -1
  air          +3     -2
  air          +4     -3
  air          +5     -4
  air          +6     -5
  air          +7     -6
  air          +8     -7
  air          +9     -8
  air         +10     -9
  air         +11    -10
  air         +12    -11
  air         +13    -12
  air         +14    -13
  end
     1    1    1    1    1    1    1    1    1    1    1    1    1    1
 14*0
     1    2    1    2    1    2    1    2    1    2    1    2    1    0
 0
 6**  3. 9z
 8**  13r10.0  13r1.0e-2  13r1.0e-1  13r5.0e-1
  12$$  1
  13$$  7014
  14**  1.16
  17**
 1.5579e-4 8.9338e-4 3.4786e-3 1.3903e-2 3.4557e-2 3.5047e-2 1.0724e-1
 8.8963e-2 2.3186e-2 1.2030e-1 2.1803e-1 1.9837e-1 1.4036e-1 1.5489e-2
  t
sambo analysis input data
4 pi r**2 fluence
 19**
     0.        0.       1.0e+4
     0.        0.       2.0e+4
     0.        0.       3.0e+4
     0.        0.       6.0e+4
     0.        0.       7.0e+4
     0.        0.       9.0e+4
     0.        0.      12.0e+4
 20**
1.0       1.0       1.0       1.0       1.0       1.0       1.0
1.0       1.0       1.0       1.0       1.0       1.0
  t
end
```

Figure F9.C.7 MORSE-SGC sample problem 4 (CGA1) input data

F9.C.5 SAMPLE PROBLEM 5: CGA2 (COMBINED NEUTRON-GAMMA)

The secondary gamma-ray dose rate due to neutrons of energies greater than 0.011 MeV at several radial distances is calculated by MORSE for a point, isotropic, 12.2 to 15 MeV source in an infinite medium of air. The air was assumed to be made up of only oxygen and nitrogen with a total density of 1.29 g/ℓ. The Multiple ARray System (MARS) combinatorial geometry package was used to describe the concentric between each of the shells for use with the boundary-crossing estimator (see part 4, Sect. 4.5.3, p. 4.5-28 of Ref. 3). This estimator requires that each detector lie on a boundary separating two media. The cross sections for air used in this calculation were processed with LAVA and were for 22 neutron groups and 18 gamma groups with 5 Legendre coefficients used for the angular expansion. All 22 neutron groups were analyzed because of the inability of MORSE-SGC to process only part of the groups, and the top 17 gamma groups were analyzed. The group structure and the response functions are given in Table F9.C.5. Splitting, Russian roulette, and path-length stretching were also implemented. Figure F9.C.9 contains the SGC input.

This problem is similar to problem number 4 except that the gamma rays must be produced and transported. The same user routines (SOURCE, BANKR, SDATA, BDRYX) written for problem number 4 are used here (see Fig. F9.C.8). Also, additional input is required for time-dependent analysis and for the GWLOW array. CGA2 sample problem results from MORSE-SGC compared well with the results obtained by running the modified version of this problem with MORSE-CGA. For neutron dose rates, the uncollided response was identical through several decimal places since it was a one-group source, and no random numbers were used to calculate the uncollided; and the total response was statistically equivalent. The gamma dose rate was also in good agreement. The random walk events (e.g., scatterings, boundary crossings, gamma production, particle deaths) differed by no more than the amount a second run with a different random number would produce with either code. Table F9.C.6 shows the results.

F9.C.6 SAMPLE PROBLEM 6: CGA6 (GAMMA ONLY)

The gamma-ray dose rate for a point, isotropic, 4–5 MeV source in an infinite medium of air is calculated by MORSE. The air is assumed to be made up of only oxygen and nitrogen with a total density of 1.29 g/ℓ. The cross-section data, as processed by LAVA, had 22 neutron/18 gamma groups, but MORSE uses only the gamma groups.

This problem is similar to problem No. 4 except it is a gamma-only problem. The same user routines (SDATA, BDRYX, SOURCE, DIREC, and GTMED) written for problem No. 4 are used here (see Fig. F9.C.8).

Figure F9.C.10 contains SGC input data.

CGA6 sample problem No. 6 results from MORSE-SGC compared well with the original MORSE-CGA results. The uncollided gamma dose rate was the same through several decimal places, and the total gamma response was statistically equivalent. The random-walk events differed by no more than the amount that a second run using a different random number would produce. Table F9.C.7 shows the results.

F9.C.7 SAMPLE PROBLEM 7: CGA8 - COLLISION DENSITY PROBLEM

The collision density sample problem is similar to sample problem No. 4 except in the types of estimates calculated. First, a boundary-crossing estimator is used for 5 spherical shells (NDC = 5). Second, a collision-density estimator is used in RELCOL for nine detector regions. Third, subroutine ENDRUN, as

```
      function direc(d,ld)
      double precision r1,r2,cos
      double precision x,y,z,u,v,w,xold,yold,zold,uold,vold,wold
      common /nutron/ name,iq7,namex,ig,igo,nmed,medold,nreg,u,v,w,uold
     * ,vold,wold,x,y,z,xold,yold,zold,wate,oldwt,wtbc,blznt,blzon,age ,
     * oldage,lnp1,lnp2,lnp3,lnp4,lnp5,lnp6,lnp7,lnp8,lnp9,lnp10,lnp11,
     * lnp12,lnp13,lnp14,lnp15,lnp16,lnp17,lnp18,lnp19,lnp20,lnp21
      dimension d(*),ld(*)
      r1=uold*xold+vold*yold+wold*zold
      r2=sqrt(xold**2+yold**2+zold**2)
      if(r2-1.e-6) 10,10,5
  5   cos=r1/r2
      direc=cos
      return
  10  direc=1.
      return
      end
      subroutine gtmed(mdgeom,mdxsec)
      data med1e/1/,med2e/2/
      if(mdgeom-med2e) 10,5,10
  5   mdxsec=med1e
      return
  10  mdxsec=mdgeom
      return
      end
      subroutine sdata( d,xd,yd,zd,vel)
c
c    subroutine sdata  calculates uncollided quantities of
c    interest at each detector position for each batch
c
      dimension d(*),xd(*),yd(*),zd(*),vel(*)
      double precision x,y,z,u,v,w,xold,yold,zold,uold,vold,wold
      common /nutron/ name,iq7,namex,ig,igo,nmed,medold,nreg,u,v,w,uold
     * ,vold,wold,x,y,z,xold,yold,zold,wate,oldwt,wtbc,blznt,blzon,age ,
     * oldage,lnp1,lnp2,lnp3,lnp4,lnp5,lnp6,lnp7,lnp8,lnp9,lnp10,lnp11,
     * lnp12,lnp13,lnp14,lnp15,lnp16,lnp17,lnp18,lnp19,lnp20,lnp21
      common/input/iadjm,nstrt,nmost,nits,nquit,ncoltp,istat,nsplt,
     * nkill,npast,noleak,iebias,nkcalc,normf, media,nmix,medalb,mxreg,
     * mfistp,nnga,ngga,nngtp, ngopt,iggopt,ndsn,ndsg,ncoef,nsct,maxgp,
     * irdsg,istr, ifmu,imom,iprin,ipun,ixtape,jxtape,io6r,igqpt, isour,
     * ngpfs,isbias,nsour, nd,nne,ne,nt,na,nresp,nex,nexnd,iflag(20),
     * tmax,tcut,wtstrt,agstrt,xstrt,ystrt,zstrt, uinp,vinp,winp,nxpm,
     * nhistr,nhismx,nbind(36),ncolls(13), ng,iftg,igg,nnuc,idt,nrp,n1m,
     * n2m,nsgps,title(20),dat(8) ,jftg,kftg,lftg
      common/qdet/locrsp,locxd,locib,locco,loct,locud,locsd, locqe,
     * locqt,locqte,locqae,lmax,efirst,egtop
      common /point/lfp1,lfp2,lfp3,lfp4,lfp5,lfp6,lfp7,lfp8,lfp9, lfp10,
     * lfp11,lfp12,lfp13,lfp14,lfp15,lfp16,lfp17,lfp18,lfp19, lfp20,
     * lfp21,lfp22,lfp23,lfp24,lap1,lap2,lap3,lap4,lap5, lap6,lap7,lsp1,
     * lsp2,lsp3, lfp25,lfp26,lfp27,lfp28,lfp29,lfp30,lfp31,lfp32, lx1,
     * lx2,lx3,lx4,lt1,lt9
      nmtg=nnga+ngga
      call nsigta(ig,ig,nmed,pnab,d(lfp11),d(lfp10),tsig,nmtg,media)
      ia= nd
      do 5 i=1,nd
      ia=ia+1
      xi=zd(ia)
      con = wate*exp(-tsig*xi)
      call fluxst(i,ig,ig,ta,cosi,-1, d(locib+3*ne), d(locqe),  d(loct),
     *     d(loct+nd*nt), d(locco), d(locqte), d(locqae), d(locud),
     *     d(locsd), d(locqt), d(locrsp),con,nmtg,ne,nd,nt,na,nresp)
c     call fluxst(i,ig,con,ta,cos,-1)
c * * switch=-1  - store array ud only
  5   continue
      return
      end
```

Figure F9.C.8 MORSE-SGC user routines for sample problems 4, 5, and 6

```
      subroutine bdryx(d,ra,extrd,nd,nmtg)
c ** this version is for morse-sgc with morse-cga sample prob ****
c     for use in spherical geometry only
c
c     identifies detector position with a boundary crossing and then
c       calculates and sums quantities of interest for each batch.
c
      common /perm/ inn,iout,nlft,lci,n12,n14,n16,n17,n81,n90,n91,n92
     1 ,n95,n96,n97,n98
      common/qdet/locrsp,locxd,locib,locco,loct,locud,locsd, locqe,
     * locqt,locqte,locqae,lmax,efirst,egtop
      double precision x,y,z,u,v,w,xold,yold,zold,uold,vold,wold
      common /nutron/ name,iq7,namex,ig,igo,nmed,medold,nreg,u,v,w,uold
     * ,vold,wold,x,y,z,xold,yold,zold,wate,oldwt,wtbc,blznt,blzon,age ,
     * oldage,lnp1,lnp2,lnp3,lnp4,lnp5,lnp6,lnp7,lnp8,lnp9,lnp10,lnp11,
     * lnp12,lnp13,lnp14,lnp15,lnp16,lnp17,lnp18,lnp19,lnp20,lnp21
      common/input/iadjm,nstrt,nmost,nits,nquit,ncoltp,istat,nsplt,
     * nkill,npast,noleak,iebias,nkcalc,normf, media,nmix,medalb,mxreg,
     * mfistp,nnga,ngga,nngtp, ngopt,iggopt,ndsn,ndsg,ncoef,nsct,maxgp,
     * irdsg,istr, ifmu,imom,iprin,ipun,ixtape,jxtape,io6r,iggpt, isour,
     * ngpfs,isbias,nsour, nz,nne,ne,nt,na,nresp,nex,nexnd,iflag(20),
     * tmax,tcut,wtstrt,agstrt,xstrt,ystrt,zstrt, uinp,vinp,winp,nxpm,
     * nhistr,nhismx,nbind(36),ncolls(13), ng,iftg,igg,nnuc,idt,nrp,nlm,
     * n2m,nsgps,title(20),dat(8) ,jftg,kftg,lftg
      double precision xdum,ydum,zdum,udum,vdum,wdum
      double precision r21,r2,r22,cos,era
      dimension d(*),extrd(nd,1),ra(*)
      integer extrd
      xdum=x
      ydum=y
      zdum=z
      udum=u
      vdum=v
      wdum=w
      call netlev(2,d)
      r21 = sqrt (x**2 + y**2 + z**2)
      r2 = r21*0.99
      r22 = r21*1.01
      do 5 i=1,nd
      if (r2-ra(i)) 15,15,5
 5    continue
      go to 10
 15   if (r22-ra(i)) 10,20,20
 20   era = u*x + v*y + w*z
      cos = era/r21 - 1.e-10
      if (cos)  30,25,30
 25   write (iout,1000)
 1000 format(1h0,14h cos=0.,return)
      go to 10
 30   abcos=abs (cos)
      if (abcos-1.0001) 40,40,35
 35   write (iout,1010) abcos
 1010 format(1h0,'abcos.gt.1. = ',e10.4)
      call errtra
 40   if (abcos-0.01) 45,50,50
 45   abcos = 0.005
 50   con=wate/abcos
       call fluxst(i,ig,ig,ta,cosi,0, d(locib+3*ne), d(locqe),  d(loct),
     *     d(loct+nd*nt), d(locco), d(locqte), d(locqae), d(locud),
     *     d(locsd), d(locqt), d(locrsp),con,nmtg,ne,nd,nt,na,nresp)
c     call fluxst(i,ig,con,age,cos,0)
c  *  *  switch =  0  -- store in all relevant arrays except ud
c     inn = locxd + 6*nd + i
c * * this store is in the first of the nexnd arrays set aside by scorin
      extrd(i,1) = extrd(i,1) + 1
 10   x=xdum
      y=ydum
      z=zdum
      u=udum
      v=vdum
      w=wdum
      return
      end
```

Figure F9.C.8 (continued)

```
     subroutine source(d,ld,fse,fs,bfs,mirz,mmiz,nmtg,media,b)
c
      double precision x,y,z,u,v,w,xold,yold,zold,uold,vold,wold
      double precision b,u1,v1,w1,fltrn
      double precision xb,wb,wp,xp,rin,rout,pinf,dist,dist0
      integer blznt,blzold
c
c   if itstr=0, must provide ig,x,y,z,u,v,w,wate and ag if desired to be
c     different from card values (which are the values input to source)
c   if itstr=1, ig is the grp no. causing fission, must provide new ig
c
      common/input/iadjm,nstrt,nmost,nits,nquit,ncoltp,istat,nsplt,
     * nkill,npast,noleak,iebias,nkcalc,normf, mmdia,nmix,medalb,mxreg,
     * mfistp,nnga,ngga,nngtp, ngopt,iggopt,ndsn,ndsg,ncoef,nsct,maxgp,
     * irdsg,istr, ifmu,imom,iprin,ipun,ixtape,jxtape,io6r,igqpt, isour,
     * ngpfs,isbias,nsour, nd,nne,ne,nt,na,nresp,nex,nexnd,iflag(20),
     * tmax,tcut,wtstrt,agstrt,xstrt,ystrt,zstrt, uinp,vinp,winp,nxpm,
     * nhistr,nhismx,nbind(36),ncolls(13), ng,iftg,igg,nnuc,idt,nrp,n1m,
     * n2m,nsgps,title(20),dat(8) ,jftg,kftg,lftg
      common /point/lfp1,lfp2,lfp3,lfp4,lfp5,lfp6,lfp7,lfp8,lfp9, lfp10,
     * lfp11,lfp12,lfp13,lfp14,lfp15,lfp16,lfp17,lfp18,lfp19, lfp20,
     * lfp21,lfp22,lfp23,lfp24,lap1,lap2,lap3,lap4,lap5, lap6,lap7,lsp1,
     * lsp2,lsp3, lfp25,lfp26,lfp27,lfp28,lfp29,lfp30,lfp31,lfp32, lx1,
     * lx2,lx3,lx4,lt1,lt9
      common /perm/ inn,iout,nlft,lci,n12,n14,n16,n17,n81,n90,n91,n92
     1 ,n95,n96,n97,n98
      common/apoll/dff,ddf,deadwt(5),eta,etath,etausd,xtra(10), iters,
     * itime,itstr,maxtim,mgpreg,inalb,ndead(5),newnm,ngeom, nlast,nmem,
     * nmgp,mmtg, npscl(13),nsigl,nxtra(10),nsup,ninf
      common /nutron/ name,iq7,namex,ig,igo,nmed,medold,nreg,u,v,w,uold
     * ,vold,wold,x,y,z,xold,yold,zold,wate,oldwt,wtbc,blznt,blzon,age ,
     * oldage,lnp1,lnp2,lnp3,lnp4,lnp5,lnp6,lnp7,lnp8,lnp9,lnp10,lnp11,
     * lnp12,lnp13,lnp14,lnp15,lnp16,lnp17,lnp18,lnp19,lnp20,lnp21
      common/parem/xb(3),wb(3),wp(3),xp(3),rin,rout,pinf,dist,ir,idbg ,
     1             irprim,nasc,lsurf,nbo,lri,lro,kloop,loop,itype,n0a
      common/gomloc/ kma,kfpd,klcr,knbd,kior,kriz,krcz,kmiz,kmcz,kkr1,
     1 kkr2,knsr,kvol,nadd,ldata,ltma,lfpd,numr,irtru,numb,nir,kbiz,kbcz
      common /mgomv/  mus,muz,l1,ipret,iflow,iect,nlo,igx
      common /orgi/ dist0,markg,nmedg,nblz,blzold,irpold
      common/repeat/jp(20)
      common/arar/nby,nlev,nar,nq,iaw,iay,nf,nx1(3)
      common/qdet/locrsp,locxd,locib,locco,loct,locud,locsd, locqe,
     * locqt,locqte,locqae,lmax,efirst,egtop
      common/save/dum,icall,du2(9)
      common/scalop/ itype1,jtype
c
      dimension fs(*),bfs(*),d(*),mirz(*),mmiz(*),fse(nmtg,media)
      dimension ld(*)
      dimension b(3)
c
c
  50  continue
      age=agstrt
      ilg=igqpt
      if (icall) 110,110,100
  100 icall = 0
      write(iout,10300) xstrt,ystrt,zstrt
      if(isour.gt.0) ddf=1.0
  110 continue
      if (itstr.ne.0) go to 130
      wate=ddf
      nty=0
 120 continue
      x = xstrt
      y = ystrt
      z =  zstrt
  130 l1=0
      j1=0
      j1u=0
      b(1)=x
```

Figure F9.C.8 (continued)

```
   b(2)=y
      b(3)=z
      call dgtiso(u1,v1,w1)
      u = u1
      v = v1
      w = w1
135   wb(1)= u
      wb(2)= v
      wb(3)= w
      kloop=kloop+1
      nix=6*nlev+4
      if (nlev.le.0) go to 150
      do 140 i=1,nix
         ix1=jp(12)+i-1
         ld(ix1)=0
  140    continue
  150 continue
      jp4=jp(4)
      jp5=jp(5)
      jp7=jp(7)
      jp8=jp(8)
      jp0=jp(10)
      call cali(jl,jlu,b,wb,jp,d(jp4),d(jp5),d(jp7),d(jp8),d(jp0),
     1 d(kfpd),d(kma),d(klcr),d,d)
      blznt=nblz
      if (irprim.le.0) call abend(ld)
      nmed=mmiz(irprim)
      nreg=mirz(irprim)
      if (itstr.ne.0) go to 210
      if (ilg.le.0) go to 170
      if (nmed.eq.ilg) go to 170
  160 continue
      nty = nty+1
      if (nty.ge.1000) go to 250
      go to 120
  170 continue
      r=fltrn()
      if (isour.gt.0) go to 190
      do 180 i=1,ngpfs
         if ( isbias.eq.0 ) go to 185
         if ( r.le.bfs(i) ) go to 200
         go to 180
  185    continue
         if ( r.le.fs(i) ) go to 200
  180    continue
      write(iout,10500) (fs(k),k=1,nmtg)
      call errtra
  190 continue
      isbias=0
      i=isour
  200 continue
      ig=i
      if (isbias.ne.0) wate = wate*fs(i)
  210 continue
      return
  250 continue
      write(iout,10600) nty
      wate = 0.0
      return
10200 format(1x,10e12.4,/)
10300 format(//,5x,'source location is',/,5x,1p,3e15.5  )
10500 format(//,10x, 36h error in fs array in source routine,//, 100(
     * 10x,1p,10e12.5,/),//)
10600 format(//,10x, 46ha source starting position was not found after,
     *  i5, 6h tries,/)
      end
```

Figure F9.C.8 (continued)

Table F9.C.4 Results from sample problem 4 (CGA1)

	$4\pi r^2$ fluence			
Detector No.	Uncoll. response	FSD uncoll.	Total response	FSD total
1	3.4192E-01	0.00659	1.8103E+0	0.06953
2	1.2887E-01	0.01253	1.8715E+0	0.05715
3	5.1554E-02	0.01822	1.6242E+0	0.05508
4	4.1993E-03	0.03357	5.4490E-01	0.06567
5	1.9259E-03	0.03798	3.5945E-01	0.10577
6	4.2804E-04	0.04595	1.3052E-01	0.07288
7	4.9218E-05	0.05680	3.8628E-02	0.20537
Neutron deaths	No.		Weight	
Killed by Russian roulette	238		0.16344E+01	
Escaped	0		0	
Reached energy cutoff	1792		0.14556E+04	
Reached time cutoff	0		0	
Number of scatterings			No.	
Medium 1			37727	

Table F9.C.5 Neutron and gamma-group structure and response functions

Group No.	Upper neutron energy (eV)	Henderson tissue dose	Upper gamma energy (eV)	Henderson tissue dose[a]
1	14.0(+6)[b]	5.4579(-9)	10.0(+6)	2.7727(-9)
2	12.2(+6)	5.1339(-9)	8.0(+6)	2.3180(-9)
3	10.0(+6)	4.8409(-9)	6.5(+6)	1.9373(-9)
4	8.18(+6)	4.6175(-9)	5.0(+6)	1.6091(-9)
5	6.36(+6)	4.4454(-9)	4.0(+6)	1.3389(-9)
6	4.96(+6)	4.3144(-9)	3.0(+6)	1.1299(-9)
7	4.06(+6)	4.0126(-9)	2.5(+6)	9.8058(-10)
8	3.01(+6)	3.3938(-9)	2.0(+6)	8.4739(-10)
9	2.46(+6)	3.1489(-9)	1.66(+6)	7.3278(-10)
10	2.35(+6)	3.0892(-9)	1.33(+6)	6.0565(-10)
11	1.83(+6)	2.6435(-9)	1.0(+6)	4.6558(-10)
12	1.11(+6)	1.9751(-9)	0.8(+6)	3.6869(-10)
13	5.50(+5)	1.1236(-9)	0.6(+6)	2.8622(-10)
14	1.11(+5)	2.2958(-10)	0.4(+6)	1.9841(-10)
15	3.35(+3)	0	0.3(+6)	1.3585(-10)
16	5.83(+2)	0	0.2(+6)	7.3791(-11)
17	1.01(+2)	0	0.1(+6)	3.6825(-11)
18	2.90(+1)	0	0.05(+6)	
19	1.07(+1)	0		
20	3.06(+0)	0		
21	1.12(+0)	0		
22	0.414(+0)	0		

[a] Units of rad (γ/cm^2).

[b] Read as 14.0×10^6.

```
=morse
sample problem 5  (neutron-gamma)
    137012466105
 1$$ 0  500 1000   10    1   0   0   1    1   1 4z
 2$$   1    1    0 1 0
 3$$  22 17 5z  6 3 40 6z 2  0 2z
 4$$ 1 0 0 0
 5$$ 10    4z  2 0  1 4z 1  5z 1 f0 t
sample prob. #5 for morse
 0 0 0 10
  sph           0.       0.          0.      3.0e+03
  sph           0.       0.          0.      5.0e+03
  sph           0.       0.          0.      7.5e+03
  sph           0.       0.          0.      1.0e+04
  sph           0.       0.          0.      1.5e+04
  sph           0.       0.          0.      2.0e+04
  sph           0.       0.          0.      3.0e+04
  sph           0.       0.          0.      6.0e+04
  sph           0.       0.          0.      7.0e+04
  sph           0.       0.          0.      9.0e+04
  sph           0.       0.          0.      1.2e+05
  sph           0.       0.          0.      1.5e+05
  sph           0.       0.          0.      1.0e+06
  sph           0.       0.          0.      1.0e+07
  end
  air          +1
  air          +2     -1
  air          +3     -2
  air          +4     -3
  air          +5     -4
  air          +6     -5
  air          +7     -6
  air          +8     -7
  air          +9     -8
  air         +10     -9
  air         +11    -10
  air         +12    -11
  air         +13    -12
  air         +14    -13
  end
    1    1    1    1    1    1    1    1    1    1    1    1    1    1
 14*0
    1    2    1    2    1    2    1    2    1    2    1    2    1    0
 0
 6** 6. 1. 2z 6r0.0
 8**
  19r10.0 3r2.0 13r5.0 5r2.0  40r1.0e-3  19r0.5 3r0.2 13r1.0 5r0.5
  13r0.5 9z 13r0.5 5z
 11**
      .7        .7        .5        .3        .1        .05      .005
      .01     6z
     .0       5.00e-4    5.0e-4    5.0e-3    1.0e-3    2.0e-3    2.0e-3
    1.0e-2
 12$$  1
```

Figure F9.C.9 MORSE-SGC sample problem 5 (CGA2) input data

```
 13$$ 7014 14**  1.16
  t
(cm**2 rad/source)
4 pi r**2 neutron dose rate
4 pi r**2 gamma dose rate
 19**
     0.         0.         5.0+3
     0.         0.         7.5+3
     0.         0.         1.0+4
     0.         0.         1.5+4
     0.         0.         3.0+4
     0.         0.         6.0+4
     0.         0.         7.0+4
     0.         0.         9.0+4
     0.         0.        1.20+5
     0.         0.        1.50+5
 20**
 5.4579e-9 5.1339e-9 4.8049e-9 4.6175e-9 4.4454e-9 4.3144e-9 4.0126e-9
 3.3938e-9 3.1489e-9 3.0892e-9 2.6435e-9 1.9751e-9 1.1236e-9 2.2958e-10
 25r0.0
  22r0.0
  2.7727e-09 2.3180e-09 1.9373e-09 1.6091e-09 1.3389e-09 1.1299e-09
9.8058e-10 8.4739e-10 7.3278e-10 6.0565e-10 4.6558e-10 3.6869e-10
 2.8622e-10 1.9841e-10 1.3585e-10 7.3791e-11 3.6825e-11
 t
end
```

Figure F9.C.9 (continued)

Table F9.C.6 Results from sample problem 5 (CGA2)

	$4\pi r^2$ neutron dose rate (cm^2 rad/source)			
Detector	Uncoll. response	FSD uncoll.	Total response	FSD total
1	3.6315E-09	0	5.9368E-09	0.03125
2	2.9621E-09	0.00017	5.9944E-09	0.02030
3	2.4162E-09	0.00007	6.0837E-09	0.01161
4	1.6076E-09	0.00003	5.9268E-09	0.03035
5	4.7353E-10	0.00015	4.5111E-09	0.01677
6	4.1084E-11	0.00015	1.9427E-09	0.06627
7	1.8188E-11	0.00005	1.4891E-09	0.04963
8	3.5645E-12	0.00011	6.3351E-10	0.05068
9	3.0927E-13	0.00005	2.2177E-10	0.19275
10	2.6832E-14	0.00011	4.6922E-11	0.06199

	$4\pi r^2$ gamma dose rate (cm^2 rad/source)	
Detector	Total response	FSD total
1	5.8971E-10	0.07463
2	8.0956E-10	0.05328
3	9.9445E-10	0.04316
4	1.1589E-09	0.02569
5	1.1439E-09	0.02597
6	6.4773E-10	0.02692
7	4.9373E-10	0.03021
8	2.7453E-10	0.03264
9	1.1823E-10	0.06700
10	4.9335E-11	0.08268

Neutron deaths	No.	Weight
Killed by Russian roulette	5472	0.49019E+01
Escaped	0	0
Reached energy cutoff	12534	0.52371E+04
Reached time cutoff	0	0

Number of scatterings	No.
Medium 1	896906

```
=morse
sample problem 6 gamma only
    343277244615
 1$$ 0  1000 1000    5    1    0  0 3r1 4z
 2$$ 1  1 0 1  0
 3$$ 0 17 5z 6 3 18 0 5z 2 3z
 4$$ 4 3z
 5$$ 10 4z 1 0 1   4z 1 5z 1 f0  t
sample prob. #6 for morse
 0 0 0 10
  sph           0.       0.          0.      3.0e+03
  sph           0.       0.          0.      5.0e+03
  sph           0.       0.          0.      7.5e+03
  sph           0.       0.          0.      1.0e+04
  sph           0.       0.          0.      1.5e+04
  sph           0.       0.          0.      2.0e+04
  sph           0.       0.          0.      3.0e+04
  sph           0.       0.          0.      6.0e+04
  sph           0.       0.          0.      7.0e+04
  sph           0.       0.          0.      9.0e+04
  sph           0.       0.          0.      1.2e+05
  sph           0.       0.          0.      1.5e+05
  sph           0.       0.          0.      1.0e+06
  sph           0.       0.          0.      1.0e+07
  end
  air          +1
  air          +2     -1
  air          +3     -2
  air          +4     -3
  air          +5     -4
  air          +6     -5
  air          +7     -6
  air          +8     -7
  air          +9     -8
  air         +10     -9
  air         +11    -10
  air         +12    -11
  air         +13    -12
  air         +14    -13
  end
    1    1    1    1    1    1    1    1    1    1    1    1    1    1
 14*0
    1    2    1    2    1    2    1    2    1    2    1    2    1    0
 0
 6**  3. 3z 6r0.0
 8** 18r10.      18r1.e-3     18r.5          18r0.5
  12$$  1  13$$  7014
 14** 1.16         t
sambo analysis sample problem 6
4 pi r**2 gamma dose rate
 19**
     0.        0.         5.0+3
     0.        0.         7.5+3
     0.        0.         1.0+4
     0.        0.         1.5+4
     0.        0.          3.0+4
     0.        0.          6.0+4
     0.        0.          7.0+4
     0.        0.          9.0+4
     0.        0.         1.20+5
     0.        0.         1.50+5
 20**
  2.7727e-09 2.3180e-09 1.9373e-09 1.6091e-09 1.3389e-09 1.1299e-09
9.8058e-10 8.4739e-10 7.3278e-10 6.0565e-10 4.6558e-10 3.6869e-10
 2.8622e-10 1.9841e-10 1.3585e-10 7.3791e-11 3.6825e-11
  t
end
```

Figure F9.C.10 MORSE-SGC sample problem 6 (CGA6 gamma only) input data

Table F9.C.7 Results from sample problem 6

	$4\pi r^2$ gamma dose rate			
Detector	Uncoll. response	FSD uncoll.	Total response	FSD total
1	1.3379E-09	0.00013	1.4827E-09	0.01092
2	1.2199E-09	0.00010	1.4169E-09	0.01102
3	1.1124E-09	0.00008	1.3725E-09	0.01082
4	9.2490E-10	0.00011	1.2388E-09	0.01386
5	5.3162E-10	0.00015	8.9717E-10	0.01824
6	1.7563E-10	0.00010	4.0302E-10	0.01778
7	1.2142E-10	0.00002	3.0910E-10	0.01254
8	5.8026E-11	0.00007	1.7035E-10	0.03963
9	1.9171E-11	0.00007	7.0052E-11	0.04644
10	6.3336E-12	0.00010	2.7444E-11	0.08996
Neutron deaths		No.	Weight	
Killed by Russian roulette		125	0.87152E-01	
Escaped		0	0	
Reached energy cutoff		4953	0.38459E+04	
Reached time cutoff		0	0	
Number of scatterings				No.
Medium 1				88024

described in Ref. 5, is used to obtain estimates of the collision-density fluence averaged over regions specified by the geometry input. Also, in versions of MORSE containing the track-length/unit-volume calculation, ENDRUN calculates track-length-per-unit-volume estimators for the geometry regions. At the present time, MORSE-SGC does not have this option, and the coding is deactivated.

This problem uses special versions of subroutines SDATA, BDRYX, INSCOR, RELCOL, RGTOMD, ENDRUN, and GTVOL, which are listed in Fig. F9.C.11, as well as SOURCE and GTMED routines from sample problem 4. INSCOR reads in NDC, the number of detectors to be stored in by BDRYX and SDATA; RELCOL uses detector numbers NDC+1 to ND. Input is given in Fig. F9.C.12.

CGA collision density problem results from MORSE-SGC compared well with the results obtained by running the modified version of this problem with MORSE-CGA. The uncollided neutron fluence was the same through several decimal places, and the total neutron and total gamma-ray fluences were statistically equivalent. The random-walk events are comparable in the two runs. Table F9.C.8 shows the results.

An additional fluence estimate per group and region is calculated in ENDRUN by using real collision weights. Although it is known that these results are not very dependable statistically, the two codes produced comparable results. For instance, region 1 primary (neutron) fluence estimate was 2.982E-9 for SGC and 3.090E-9 for CGA, while secondary (gamma-ray) fluence was 1.554E-9 for SGC and 1.548E-9 for CGA. No FSDs are associated with these values.

F9.C.8 SAMPLE PROBLEM 8 RESULTS: ILLUSTRATES USE OF MARS ARRAYS

This problem was run with MORSE-SGC using the source energy biasing parameters obtained from the same adjoint XSDRNPM calculation used by Problem 3. The XSDRNPM input is shown in Fig. F9.C.4; and the resulting adjoint fluxes are shown in Table F9.C.2. See Sect. F9.C.3 of this document for more details.

The input data for this calculation are shown in Fig. F9.C.13. This problem requires the same subroutines (SDATA and ESCAPE) used by Problem 3 (See Fig. F9.C.5 and Sect. F9.C.3). An upper axial leakage of 0.0265 was obtained with a fractional standard deviation of 4.29% for 20,000 histories. Sixteen-hundred-sixty-five contributions were made to the detector.

The result of this problem shows a slightly higher neutron leakage than problem 3; however, the difference is within the statistical uncertainty of the calculations. Higher leakage in the heterogeneous fuel rod model is to be expected. Due to the additional detail in the geometry modeling, the total cpu time for this problem is nearly a factor of 5 higher than for problem 3.

F9.C.9 REFERENCES

1. *IF300 Shipping Cask – Consolidated Safety Analysis Report,* NEDO-10084-2, General Electric Company, October 1979.
2. M. B. Emmett, *The MORSE Monte Carlo Radiation Transport Code System*, ORNL-4972 (1975), ORNL-4972/R1 (1983), ORNL-4972/R2 (1984), Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., 1984.
3. N. M. Greene et al., *AMPX-77: A Modular Code System for Generating Coupled Multigroup Neutron-Gamma Cross-Section Libraries from ENDF/B-IV and/or ENDF/B-V*, ORNL/CSD/TM-283, Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., October 1992.
4. E. A. Straker and M. B. Emmett, *Collision Site Plotting Routines and Collision Density Fluence Estimates for the MORSE Monte Carlo Code*, ORNL/TM-3585, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., October 1971.

```
      subroutine sdata( d,xd,yd,zd,vel)
      dimension d(*),xd(*),yd(*),zd(*),vel(*)
c *** this version for collision density sample problem morse-sgc ****
c     subroutine  sdata  calculates uncollided quantities of interest at
c       each detector position for each batch.
c
      double precision x,y,z,u,v,w,xold,yold,zold,uold,vold,wold
      common /nutron/ name,iq7,namex,ig,igo,nmed,medold,nreg,u,v,w,uold
     * ,vold,wold,x,y,z,xold,yold,zold,wate,oldwt,wtbc,blznt,blzon,age ,
     * oldage,lnp1,lnp2,lnp3,lnp4,lnp5,lnp6,lnp7,lnp8,lnp9,lnp10,lnp11,
     * lnp12,lnp13,lnp14,lnp15,lnp16,lnp17,lnp18,lnp19,lnp20,lnp21
      common/input/iadjm,nstrt,nmost,nits,nquit,ncoltp,istat,nsplt,
     * nkill,npast,noleak,iebias,nkcalc,normf, media,nmix,medalb,mxreg,
     * mfistp,nnga,ngga,nngtp, ngopt,iggopt,ndsn,ndsg,ncoef,nsct,maxgp,
     * irdsg,istr, ifmu,imom,iprin,ipun,ixtape,jxtape,io6r,igqpt, isour,
     * ngpfs,isbias,nsour, nd,nne,ne,nt,na,nresp,nex,nexnd,iflag(20),
     * tmax,tcut,wtstrt,agstrt,xstrt,ystrt,zstrt, uinp,vinp,winp,nxpm,
     * nhistr,nhismx,nbind(36),ncolls(13), ng,iftg,igg,nnuc,idt,nrp,nlm,
     * n2m,nsgps,title(20),dat(8)
      common/qdet/locrsp,locxd,locib,locco,loct,locud,locsd, locqe,
     * locqt,locqte,locqae,lmax,efirst,egtop
      common /point/lfp1,lfp2,lfp3,lfp4,lfp5,lfp6,lfp7,lfp8,lfp9, lfp10,
     * lfp11,lfp12,lfp13,lfp14,lfp15,lfp16,lfp17,lfp18,lfp19, lfp20,
     * lfp21,lfp22,lfp23,lfp24,lap1,lap2,lap3,lap4,lap5, lap6,lap7,lsp1,
     * lsp2,lsp3, lfp25,lfp26,lfp27,lfp28,lfp29,lfp30,lfp31,lfp32, lx1,
     * lx2,lx3,lx4,lt1,lt9
      common/detcut/ndc
      nmtg=nnga+ngga
      call nsigta(ig,ig,nmed,pnab,d(lfp11),d(lfp10),tsig,nmtg,media)
      ia= nd
      do 5 i=1,ndc
      ia=ia+1
c   ia  corresponds to   rad
      xi=zd(ia)
      con=wate*exp (-tsig*xi)
      call fluxst(i,ig,ig,ta,cosi,-1, d(locib+3*ne), d(locqe),  d(loct),
     *     d(loct+nd*nt), d(locco), d(locqte), d(locqae), d(locud),
     *     d(locsd), d(locqt), d(locrsp),con,nmtg,ne,nd,nt,na,nresp)
c  *  * switch=-1  - store array ud only
 5    continue
      return
      end
      subroutine bdryx(d,ra,extrd,nd,nmtg)
c     this version for collision density sample prob - morse-sgc * *
c    it stores in detector numbers 1 thru ndc ***
c     for use in spherical geometry only
c
c     identifies detector position with a boundary crossing and then
c       calculates and sums quantities of interest for each batch.
c
      common /perm/ inn,iout,nlft,lci,n12,n14,n16,n17,n81,n90,n91,n92
     1 ,n95,n96,n97,n98
      common/qdet/locrsp,locxd,locib,locco,loct,locud,locsd, locqe,
     * locqt,locqte,locqae,lmax,efirst,egtop
      double precision x,y,z,u,v,w,xold,yold,zold,uold,vold,wold
      common /nutron/ name,iq7,namex,ig,igo,nmed,medold,nreg,u,v,w,uold
     * ,vold,wold,x,y,z,xold,yold,zold,wate,oldwt,wtbc,blznt,blzon,age ,
     * oldage,lnp1,lnp2,lnp3,lnp4,lnp5,lnp6,lnp7,lnp8,lnp9,lnp10,lnp11,
     * lnp12,lnp13,lnp14,lnp15,lnp16,lnp17,lnp18,lnp19,lnp20,lnp21
      common/input/iadjm,nstrt,nmost,nits,nquit,ncoltp,istat,nsplt,
     * nkill,npast,noleak,iebias,nkcalc,normf, media,nmix,medalb,mxreg,
     * mfistp,nnga,ngga,nngtp, ngopt,iggopt,ndsn,ndsg,ncoef,nsct,maxgp,
     * irdsg,istr, ifmu,imom,iprin,ipun,ixtape,jxtape,io6r,igqpt, isour,
     * ngpfs,isbias,nsour, nz,nne,ne,nt,na,nresp,nex,nexnd,iflag(20),
     * tmax,tcut,wtstrt,agstrt,xstrt,ystrt,zstrt, uinp,vinp,winp,nxpm,
     * nhistr,nhismx,nbind(36),ncolls(13), ng,iftg,igg,nnuc,idt,nrp,nlm,
     * n2m,nsgps,title(20),dat(8)
      common/detcut/ndc
      dimension d(*),ra(*),extrd(nd,1)
```

Figure F9.C.11 MORSE-SGC user routines for sample problem 7 (CGA8)

```
      integer extrd
      xdum=x
      ydum=y
      zdum=z
      udum=u
      vdum=v
      wdum=w
      call netlev(2,d)
      r21 = sqrt (x**2 + y**2 + z**2)
      r2 = r21*0.99
      r22 = r21*1.01
      do 5 i=1,ndc
      if (r2-ra(i)) 15,15,5
 5    continue
      go to 10
 15   if (r22-ra(i)) 10,20,20
 20   era = u*x + v*y + w*z
      cos = era/r21 - 1.e-10
      if (cos)  30,25,30
 25   write (iout,1000)
 1000 format(1h0,14h cos=0.,return)
      return
 30   abcos=abs (cos)
      if (abcos-1.0001) 40,40,35
 35   write (iout,1010) abcos
 1010 format(1h0,'abcos.gt.1. = ',e10.4)
      call errtra
 40   if (abcos-0.01) 45,50,50
 45   abcos = 0.005
 50   con=wate/abcos
       call fluxst(i,ig,ig,ta,cosi,0, d(locib+3*ne), d(locqe),  d(loct),
     *     d(loct+nd*nt), d(locco), d(locqte), d(locqae), d(locud),
     *     d(locsd), d(locqt), d(locrsp),con,nmtg,ne,nd,nt,na,nresp)
c  *  *  switch =  0  -- store in all relevant arrays except ud
c * * this store is in the first of the nexnd arrays set aside by scorin
      extrd(i,1) = extrd(i,1) + 1
 10   x=xdum
      y=ydum
      z=zdum
      u=udum
      v=vdum
      w=wdum
      return
      end
      subroutine inscor (rad,fact)
c    this version for collision density sample problem morse-sgc * * *
c calculate geom. factors  fact(i)=1/(4 pi/3*(r3(i+1)-r3(i)))
c   store them in fact(i)
c    spherical geometry
      common/qdet/locrsp,locxd,locib,locco,loct,locud,locsd, locqe,
     * locqt,locqte,locqae,lmax,efirst,egtop
      common/input/iadjm,nstrt,nmost,nits,nquit,ncoltp,istat,nsplt,
     * nkill,npast,noleak,iebias,nkcalc,normf, media,nmix,medalb,mxreg,
     * mfistp,nnga,ngga,nngtp, ngopt,iggopt,ndsn,ndsg,ncoef,nsct,maxgp,
     * irdsg,istr, ifmu,imom,iprin,ipun,ixtape,jxtape,io6r,igqpt, isour,
     * ngpfs,isbias,nsour, nd,nne,ne,nt,na,nresp,nex,nexnd,iflag(20),
     * tmax,tcut,wtstrt,agstrt,xstrt,ystrt,zstrt, uinp,vinp,winp,nxpm,
     * nhistr,nhismx,nbind(36),ncolls(13), ng,iftg,igg,nnuc,idt,nrp,n1m,
     * n2m,nsgps,title(20),dat(8)
      common/detcut/ndc
      common /perm/ inn,iout,nlft,lci,n12,n14,n16,n17,n81,n90,n91,n92
     1 ,n95,n96,n97,n98
      dimension fact(*),rad(*)
      read(inn,101) ndc
 101  format(i5)
      do 11 id=1,nd
      if(id .gt. 14) go to 11
      if(id.gt.ndc) go to 5
```

Figure F9.C.11 (continued)

```
      fact(id)= 1.
      go to 11
   5  ri=rad(id)
      if(id.eq.nd) ri=1.5e+5
c **  1.5e+5 is assumed to be effective outer edge of system  ****
      if(id-ndc-1) 11,10,9
  10  rim=0.
      go to 12
   9  rim=rad(id-1)
   12 fact(id)=0.23873/(ri*ri*ri-rim*rim*rim)
   11 continue
      return
      end
      subroutine relcol(d,xd,yd,zd,vel,extrg,extrd,nmtg,nd)
c    this version for collision density sample problem - morse-sgc
c *  stores estimate in detector numbers ndc+1 to nd  * * * * *
c    spherical geometry
      dimension d(*),xd(*),yd(*),zd(*),vel(*)
c     sample calling sequence
c     call relcol(d,d(locxd),d(locxd+nd),d(locxd+2*nd),d(lfp1),
c    1 d(locrsp+nresp*nmtg),d(locxd+6*nd))
c
      dimension extrg(nmtg,*),extrd(nd,1)
      double precision x,y,z,u,v,w,xold,yold,zold,uold,vold,wold
      common /nutron/ name,iq7,namex,ig,igo,nmed,medold,nreg,u,v,w,uold
     * ,vold,wold,x,y,z,xold,yold,zold,wate,oldwt,wtbc,blznt,blzon,age ,
     * oldage,lnp1,lnp2,lnp3,lnp4,lnp5,lnp6,lnp7,lnp8,lnp9,lnp10,lnp11,
     * lnp12,lnp13,lnp14,lnp15,lnp16,lnp17,lnp18,lnp19,lnp20,lnp21
      common/qdet/locrsp,locxd,locib,locco,loct,locud,locsd, locqe,
    * locqt,locqte,locqae,lmax,efirst,egtop
      common /perm/ inn,iout,nlft,lci,n12,n14,n16,n17,n81,n90,n91,n92
     1 ,n95,n96,n97,n98
      common/detcut/ndc
      common /point/lfp1,lfp2,lfp3,lfp4,lfp5,lfp6,lfp7,lfp8,lfp9, lfp10,
     * lfp11,lfp12,lfp13,lfp14,lfp15,lfp16,lfp17,lfp18,lfp19, lfp20,
     * lfp21,lfp22,lfp23,lfp24,lap1,lap2,lap3,lap4,lap5, lap6,lap7,lsp1,
     * lsp2,lsp3, lfp25,lfp26,lfp27,lfp28,lfp29,lfp30,lfp31,lfp32, lx1,
     * lx2,lx3,lx4,lt1,lt9
      common/input/iadjm,nstrt,nmost,nits,nquit,ncoltp,istat,nsplt,
     * nkill,npast,noleak,iebias,nkcalc,normf, media,nmix,medalb,mxreg,
     * mfistp,nnga,ngga,nngtp, ngopt,iggopt,ndsn,ndsg,ncoef,nsct,maxgp,
     * irdsg,istr, ifmu,imom,iprin,ipun,ixtape,jxtape,io6r,igqpt, isour,
     * ngpfs,isbias,nsour, nz,nne,ne,nt,na,nresp,nex,nexnd,iflag(20),
     * tmax,tcut,wtstrt,agstrt,xstrt,ystrt,zstrt, uinp,vinp,winp,nxpm,
     * nhistr,nhismx,nbind(36),ncolls(13), ng,iftg,igg,nnuc,idt,nrp,nlm,
     * n2m,nsgps,title(20),dat(8)
      data isig/1/
      common bc(1)
      if(isig.ne.1) go to 999
      isig=0
      nd1= ndc+1
  999  continue
      rsq=x*x+y*y+z*z
      r=sqrt(rsq)
      call nsigta(igo,igo,nmed,pnab,d(lfp11),d(lfp10),tsig,nmtg,media)
      rsq=12.566*rsq
      con=wtbc *rsq/tsig
      nd1= ndc+1
      do 11 id=nd1,nd
      if(id.eq.nd) go to 12
      rdx=d(locxd+3*nd+id-1)
      if(r.le.rdx   ) go to 12
   11 continue
      go to 20
   12 call fluxst(id,igo,igo,ta,cosi,0,d(locib+3*ne),d(locqe),  d(loct),
     *     d(loct+nd*nt), d(locco), d(locqte), d(locqae), d(locud),
     *     d(locsd), d(locqt), d(locrsp),con,nmtg,ne,nd,nt,na,nresp)
   20 return
      end
      subroutine rgtomd (mmed, mxreg)
```

Figure F9.C.11 (continued)

```
c
c   routine to determine media number corresponding to region number.
c ***  special version for collision density sample problem
c
      dimension mmed(mxreg)
c
c
      do 5 i = 1, mxreg
      mmed(i) = 1
    5 continue
c
      return
      end
      subroutine endrun (wts)
c     this version is for morse-sgc collision density sample problem * *
c * * * * *                                  * * * *
c * * * * local arrays in endrun may require change of dimension  * * *
c * * * * *                                     * * * *
      common /point/lfp1,lfp2,lfp3,lfp4,lfp5,lfp6,lfp7,lfp8,lfp9, lfp10,
     * lfp11,lfp12,lfp13,lfp14,lfp15,lfp16,lfp17,lfp18,lfp19, lfp20,
     * lfp21,lfp22,lfp23,lfp24,lap1,lap2,lap3,lap4,lap5, lap6,lap7,lsp1,
     * lsp2,lsp3, lfp25,lfp26,lfp27,lfp28,lfp29,lfp30,lfp31,lfp32, lx1,
     * lx2,lx3,lx4,lt1,lt9
      dimension wts(*)
      common /perm/ inn,iout,nlft,lci,n12,n14,n16,n17,n81,n90,n91,n92
     1 ,n95,n96,n97,n98
      common/qdet/locrsp,locxd,locib,locco,loct,locud,locsd, locqe,
     * locqt,locqte,locqae,lmax,efirst,egtop
      common/input/iadjm,nstrt,nmost,nits,nquit,ncoltp,istat,nsplt,
     * nkill,npast,noleak,iebias,nkcalc,normf, media,nmix,medalb,mxreg,
     * mfistp,nnga,ngga,nngtp, ngopt,iggopt,ndsn,ndsg,ncoef,nsct,maxgp,
     * irdsg,istr, ifmu,imom,iprin,ipun,ixtape,jxtape,io6r,igqpt, isour,
     * ngpfs,isbias,nsour, nd,nne,ne,nt,na,nresp,nex,nexnd,iflag(20),
     * tmax,tcut,wtstrt,agstrt,xstrt,ystrt,zstrt, uinp,vinp,winp,nxpm,
     * nhistr,nhismx,nbind(36),ncolls(13), ng,iftg,igg,nnuc,idt,nrp,n1m,
     * n2m,nsgps,title(20),dat(8)
      dimension gnor(25),mmed(25),sum1(25),sum2(25)
      character*24 tmtl,tmt2
      character*4 tirc(6),titl(6),tout(6)
      equivalence (tmtl,tirc(1)),(tmt2,titl(1))
      data pn/1./
      data tmtl/ ’real collision weights  ’/
      data tmt2/ ’track length estimator  ’/
      data ihb,hr1,hr2,hr3,hr4/1h ,4hflue,4hnce ,4hvalu,4he   /
      data gnor /25*1./
      data mmed /25*0/
c * * * * for cases with more than 1 media, set mmed(nreg) equal * * * *
c * * * * to the medium for each region  * * * * * * * * * * * * * * * *
c * * * * note * * a region cannot contain more than 1 medium  * * *
c * * * * lswhr is the number of arrays length nmtg used elsewhere in *
c * * * * the case * * * * * * * * * * * * * * * * * * * * * * * * * * *
      lswhr=0
      nmtg=nnga+ngga
c*********************************************************************
c * * * * nex should be equal to lswhr + mxreg + 2 * * * * * * * * * * *
c * * if it is not, assumes no calculation desired in endrun
      ebotn = wts(nnga+1)
      ebotg = wts(nmtg+2)
      locnsc = lfp19
      if(nex-lswhr-mxreg-2) 2,3,3
    2 return
    3 continue
      if(mxreg.le.25) go to 4
      write(iout,1080)
 1080 format(48h0mxreg is greater than 25,increase dimensions in  ,
     1 19h endrun and rerun.    )
      stop
```

Figure F9.C.11 (continued)

```
  4   continue
      call rgtomd(mmed, mxreg)
      iflg=0
      il= locrsp+nresp*nmtg+lswhr*nmtg-1
      n3= locnsc + nmtg*mxreg-1
  1   continue
      if(iflg.eq.0) go to 11
      do 12 i=1,6
 12   tout(i)=titl(i)
      go to 13
 11   do 14 i=1,6
 14   tout(i)=tirc(i)
 13   continue
      fnt= nstrt* nits
      n=nnga-1
      l=il
c    calculate dele
      m=1
      if(iadjm)10,10,15
 10   fn= 1.
      ebot1 = ebotn
      ebot2=ebotg
      if(nnga.eq.0) ebot1= ebotg
c   * if gamma only problem, set 1st ebot = lowest gamma energy ***
      go to 30
 15   fn=-1.
      ebot1= ebotg
      ebot2= ebotn
      if(nnga*ngga) 25, 25, 20
 20   n= nmtg-nnga-1
 25   if(ngga .eq.0) ebot1= ebotn
c ** if neutron only, set 1st ebot = ebotn
      l= l+1
      wts(l) = wts(l) -ebot1
      if(wts(l) .eq.0.) wts(l) = 1.
 30   continue
      do 35 i=m,n
      l= l+1
 35   wts(l) = fn *(wts(i)-wts(i+1))
      if(iadjm) 40,40,45
 40   l=l+1
      wts(l) = wts(nnga)-ebot1
      if(wts(l) .eq.0.) wts(l) = 1.
 45   if(nnga*ngga) 75,75,50
 50   n=n+3
      m= nmtg
c * * * * * * * * * * * * *
c   in morse-sgc, the bottom neutron energy is stored in the energy arra
c  rather than separately, therefore the 1st gamma energy is in
c   location nnga+2 rather than in nnga+1 as it is in morse-cga.
c   in cga, the above statements would be n=n+2  and m=nmtg-1
c * * * * * * * * * * * *
      if(iadjm) 60,60,55
 55   l=l+1
      wts(l) = wts(n) -ebot2
      if(wts(l) .eq.0.) wts(l) = 1.
 60   do 65 i=n,m
      l= l+1
      wts(l) = fn*(wts(i)-wts(i+1))
c65   wts(l) = fn*(wts(i)-wts(i+1))
 65   continue
      if(iadjm) 70,70,75
 70   l=l+1
      wts(l) = wts(nmtg)-ebot2
      if(wts(l) .eq.0.) wts(l) = 1.
 75   call gtvol(wts,mxreg, gnor)
c   gtvol returns 1/vol  of each region to calculate fluence/cm**2
c     for the sample problem, gtvol calculates 4 pi r**2/vol
      iw=il+nmtg
```

Figure F9.C.11 (continued)

```
      l=iw
      limz=nnga
      if(iadjm.gt.0.and.nnga*ngga.gt.0) limz=nmtg-nnga
      lim=limz+1
      do 110 nreg= 1,mxreg
      n1 = n3+(nreg-1)*nmtg
      sum1(nreg)=0
      imed = mmed(nreg)
      if(limz .le.0) go to 92
      do 90 i=1,limz
      n2= n1+i
       l=l+1
      if(imed.eq.0) go to 86
      if(iflg.eq.0) call nsigta(i,i,mmed(nreg),pn,
     1wts(lfp11),wts(lfp10),sigt,nmtg,media)
      con=wts(n2)*gnor(nreg)/fnt/sigt
      sum1(nreg)=sum1(nreg)+con
      if(iadjm) 85,85,80
 80   wts(l) = con
      go to 90
 85   wts(l)=con/wts(il+i)
      goto 90
   86 wts(l) = 0.
 90   continue
 92   if (nmtg.eq.nnga) go to 110
      sum2(nreg)=0
      do 105 i=lim,nmtg
      l=l+1
      n2=n1+i
      if(imed.eq.0) go to 101
      if(iflg.eq.0)call nsigta(i,i,mmed(nreg),pn,
     1wts(lfp11),wts(lfp10),sigt,nmtg,media)
      con=wts(n2)*gnor(nreg)/fnt/sigt
      sum2(nreg)=sum2(nreg)+con
      if(iadjm) 100,100,95
 95   wts(l) = con
      go to 105
 100  wts(l)=con/wts(il+i)
      goto 105
  101 wts(l) = 0.
 105  continue
 110  continue
      nr1= 1
      nr0= iw+1
      if(iadjm) 120,120,115
  115 write(iout,1010)(tout(i),i=1,6)
1010  format( 26h1importance estimate from ,6a4,20h (importance/source)
     1 /1h0,40x,28h*** beware of statistics *** /1h0/)
      hr1 = hr3
      hr2 = hr4
      go to 125
  120 write(iout,1020)(tout(i),i=1,6)
c     this title is for sample problem only *****
1020  format( 23h1fluence estimate from ,6a4,31h(neutrons or gammas/ev/s
     1ource)  /1h0,40x,28h*** beware of statistics *** /1h0/)
c1020 format(1h1, 'fluence estimate from real collision weights (neutron
c    1s or gammas/cm**2/ev/source) ' /1h0,40x,'***beware of statistics *
c    2** ' /1h0/)
 125  continue
      nr2= nr1+5
      if(nr2.gt.mxreg)  nr2=mxreg
      nr= iw+nr2*nmtg
      write(iout,1060) (ihb,l,l=nr1,nr2)
      write(iout,1070) (ihb,hr1,hr2,l=nr1,nr2)
      do 130 i=1,nmtg
      write(iout,1030) i,(wts(k),k=nr0,nr,nmtg)
 1030 format(i5,6(3x,1pe12.3) )
      nr0= nr0+1
```

Figure F9.C.11 (continued)

```
  130  continue
       write (iout,1040) (sum1(nrr),nrr=nr1,nr2)
 1040  format(10h primary   ,6(1pe10.3,5x))
       if (nmtg.eq.nnga) go to 135
       write (iout,1050) (sum2(nrr),nrr=nr1,nr2)
 1050  format(10h secondary ,6(1pe10.3,5x))
  135  continue
       nr1= nr1+6
       nr0 = nr+1
       if(nr1.le.mxreg)  go to 125
  1060 format(8h0energy ,6(a1,3x,6hregion ,i3,2x))
  1070 format(7h  group,2x,6(a1,2x,a4,a3,5x))
       return
c * *  activate for morse versions calculating track length
c      if(iflg.gt.0) return
c      n3=locnsc+9*nmtg*mxreg
c      sigt=1.
c      iflg=1
c      go to 1
c * * * * * * * * *
       end
       subroutine gtvol(stor,mxreg,gnor)
c * * * version for combinatorial geometry non-ibm computers* * * *
       dimension stor(*)
       dimension gnor(mxreg)
       common/gomloc/ kma,kfpd,klcr,knbd,kior,kriz,krcz,kmiz,kmcz,kkr1,
      1 kkr2,knsr,kvol,nadd,ldata,ltma,lfpd,numr,irtru,numb,nir,kbiz,kbcz
clng
c      n=1
clng
c       activate for ibm double prec
cshr
       n=2
cshr
       do 10 i=1,mxreg
       ins=kvol+n*(i-1)
   10  gnor(i)=1.0/fvol(stor(ins))
       return
       end
       function fvol(v)
c      this routine used only by gtvol
       double precision v
       fvol = v
       return
       end
```

Figure F9.C.11 (continued)

```
=morse
collision density sample problem #7
    235467251643
 1$$ 0 40 100 100     1   0  3r1 5r0
 2$$ 1  1 0   10  0
 3$$ 22  17 5z 6 3 39 0  5z 2 3z
  4$$ 1 3z
 5$$ 14 7 10 0 0 5 12 1 4z 1 3z 1 0 1 f0 t
combinatorial geometry for infinite air problem
 1 0 0 10
  sph        0.0       0.0       0.0     10000.
  sph        0.0       0.0       0.0     15000.
  sph        0.0       0.0       0.0     20000.
  sph        0.0       0.0       0.0     30000.
  sph        0.0       0.0       0.0     40000.
  sph        0.0       0.0       0.0     45000.
  sph        0.0       0.0       0.0     50000.
  sph        0.0       0.0       0.0     70000.
  sph        0.0       0.0       0.0     75000.
  sph        0.0       0.0       0.0     80000.
  sph        0.0       0.0       0.0     90000.
  sph        0.0       0.0       0.0    100000.
  sph        0.0       0.0       0.0    120000.
  sph        0.0       0.0       0.0    140000.
  sph        0.0       0.0       0.0    500000.
  sph        0.0       0.0       0.0    900000.
  end
   a1         +1
   a2         +2  -1
   a3       +3    -2
   a4         +4  -3
   x5         +5  -4
   a6         +6  -5
   a7         +7  -6
   a8         +8  -7
   a9         +9  -8
  a10        +10  -9
  a11        +11  -10
  a12        +12  -11
  a13        +13  -12
  a14        +14  -13
  a15        +15  -14
  a16        +16  -15
  end
    1    2    2    3    3    4    4    5    6    6    7    7    8    9
   10   10
 16*0
    1    1    2    2    1    1    2    2    2    1    1    2    2    2
    2    0
 0
 6** 6.0 0.0 8r0
 8**
  19r10.0 3r2.0 13r5.0 4r2.0  19r10.0 3r2.0 13r5.0 4r2.0
  19r10.0 3r2.0 13r5.0 4r2.0  19r10.0 3r2.0 13r5.0 4r2.0
  19r10.0 3r2.0 13r5.0 4r2.0  19r10.0 3r2.0 13r5.0 4r2.0
  19r10.0 3r2.0 13r5.0 4r2.0  19r10.0 3r2.0 13r5.0 4r2.0
  19r10.0 3r2.0 13r5.0 4r2.0  19r10.0 3r2.0 13r5.0 4r2.0
  390r1.0e-3
 19r0.5 3r0.2 13r1.0 4r0.5  19r0.5 3r0.2 13r1.0 4r0.5
 19r0.5 3r0.2 13r1.0 4r0.5  19r0.5 3r0.2 13r1.0 4r0.5
 19r0.5 3r0.2 13r1.0 4r0.5  19r0.5 3r0.2 13r1.0 4r0.5
 19r0.5 3r0.2 13r1.0 4r0.5  19r0.5 3r0.2 13r1.0 4r0.5
 19r0.5 3r0.2 13r1.0 4r0.5  19r0.5 3r0.2 13r1.0 4r0.5
 390z
 11**  .3      .3    .2         .2        .2        .1        .05
    .10e-1    .10e-1    .10e-1    .10e-1     .5e-2    .10e-2    .50e-3
     5.e-4     5.e-4     1.e-3     1.e-3     1.e-3     1.e-3     1.e-2
     1.e-2
     .3      .3         .2         .2        .2        .1        .05
    .10e-1    .10e-1    .10e-1    .10e-1     .5e-2    .10e-2    .50e-3
     5.e-4     5.e-4     1.e-3     1.e-3     1.e-3     1.e-3     1.e-2
     1.e-2
```

Figure F9.C.12 MORSE-SGC sample problem 7 (CGA8) input data

```
     .3      .3         .2         .2        .2        .1        .05
    .10e-1    .10e-1    .10e-1    .10e-1     .5e-2    .10e-2    .50e-3
     5.e-4     5.e-4     1.e-3     1.e-3     1.e-3     1.e-3     1.e-2
     1.e-2
     .3      .3         .2         .2        .2        .1        .05
    .10e-1    .10e-1    .10e-1    .10e-1     .5e-2    .10e-2    .50e-3
     5.e-4     5.e-4     1.e-3     1.e-3     1.e-3     1.e-3     1.e-2
     1.e-2
     .3      .3         .2         .2        .2        .1        .05
    .10e-1    .10e-1    .10e-1    .10e-1     .5e-2    .10e-2    .50e-3
     5.e-4     5.e-4     1.e-3     1.e-3     1.e-3     1.e-3     1.e-2
     1.e-2
     .3      .3         .2         .2        .2        .1        .05
    .10e-1    .10e-1    .10e-1    .10e-1     .5e-2    .10e-2    .50e-3
     5.e-4     5.e-4     1.e-3     1.e-3     1.e-3     1.e-3     1.e-2
     1.e-2
     .3      .3         .2         .2        .2        .1        .05
    .10e-1    .10e-1    .10e-1    .10e-1     .5e-2    .10e-2    .50e-3
     5.e-4     5.e-4     1.e-3     1.e-3     1.e-3     1.e-3     1.e-2
     1.e-2
     .3      .3         .2         .2        .2        .1        .05
    .10e-1    .10e-1    .10e-1    .10e-1     .5e-2    .10e-2    .50e-3
     5.e-4     5.e-4     1.e-3     1.e-3     1.e-3     1.e-3     1.e-2
     1.e-2
     .3      .3         .2         .2        .2        .1        .05
    .10e-1    .10e-1    .10e-1    .10e-1     .5e-2    .10e-2    .50e-3
     5.e-4     5.e-4     1.e-3     1.e-3     1.e-3     1.e-3     1.e-2
     1.e-2
     .3      .3         .2         .2        .2        .1        .05
    .10e-1    .10e-1    .10e-1    .10e-1     .5e-2    .10e-2    .50e-3
     5.e-4     5.e-4     1.e-3     1.e-3     1.e-3     1.e-3     1.e-2
     1.e-2
  12$$  1  13$$  7014
 14**  1.16
  t
sambo for 10 region air
4 pi r**2 group 1 fluence (neutrons/source)
4 pi r**2 group 10 + 11 fluence (neutrons/source)
4 pi r**2 group 28 fluence ( gammas/source )
4 pi r**2 fast neutron fluence (neutrons/source)
4 pi r**2 gamma ray fluence (gammas/source)
(neutrons/ev/source)
 19**
     0.        0.        1.5e+4
     0.        0.        3.0e+4
     0.        0.        4.5e+4
     0.        0.        7.5e+4
     0.        0.        9.0e+4
     0.        0.        1.0e+4
     0.        0.        2.0e+4
     0.        0.        4.0e+4
     0.        0.        5.0e+4
     0.        0.        7.0e+4
     0.        0.        8.0e+4
     0.        0.       10.0e+4
     0.        0.       12.0e+4
     0.        0.       14.0e+4
 20**
   1.0 38r0.0
  9r0.0 2r1.0 28r0.0
 27r0.0 1.0 11r0.0
  22r1.0 17r0.0
  22r0.0 17r1.0
 21$$ 1    3    4    6    9   11   13   27   28   39
   t
    5
end
```

Figure F9.C.12 (continued)

Table F9.C.8 Results from sample problem 7
(100 batches of 40 particles)

$4\pi r^2$ fast neutron fluence (neutrons/source)

Detector	Uncoll. response	FSD uncoll.	Total response	FSD total
1	2.9455E-01	0.00013	1.6835E+0	0.02388
2	8.6761E-02	0.00012	2.1256E+0	0.05465
3	2.5556E-02	0.00017	1.8546E+0	0.02068
4	2.2172E-03	0.00013	1.0854E+0	0.03122
5	6.5310E-04	0.00004	7.2959E-01	0.04364
6	0	0	1.3744E+0	0.02288
7	0	0	1.7668E+0	0.01937
8	0	0	2.0580E+0	0.01549
9	0	0	1.8887E+0	0.01778
10	0	0	1.5445E+0	0.01641
11	0	0	1.0944E+0	0.02624
12	0	0	6.9138E-01	0.02952
13	0	0	3.2199E-01	0.03716
14	0	0	2.2335E-01	0.06414

$4\pi r^2$ gamma-ray fluence (gammas/source)

Detector	Total response	FSD total
1	1.8671E+0	0.03590
2	2.3374E+0	0.03844
3	2.0659E+0	0.03181
4	1.0634E+0	0.04872
5	7.0830E-01	0.06306
6	1.0339E+0	0.05670
7	1.9054E+0	0.03597
8	2.4037E+0	0.02618
9	2.1083E+0	0.02689
10	1.5378E+0	0.02931
11	1.0625E+0	0.04577
12	7.1153E-01	0.04519
13	3.7193E-01	0.06360
14	4.1743E-01	0.08860

Neutron deaths	No.	Weight
Killed by Russian roulette	4107	0.38345E+01
Escaped	0	0
Reached energy cutoff	7940	0.42919E+04
Reached time cutoff	0	0

Number of scatterings	No.
Medium 1	778625

```
=nitawl
0$$ 85 e  1$$ a2 11 e 1t
2$$ 8016 24000 25055 26000 28000 40000 42000 92235 92238 94239 94240 2t
end
=morse
morse-sgc/s sample problem 8
    13579bdfdb97
1$$ 0 100 800 200 1 0 0 1 1 1 0 0 0 0
2$$ 4 13 88 8 1
3$$ 22 5r0 3 4 2 22 0 1 4r0 4 0 0 1
4$$ 0 14 1 0
5$$ 2 0 0 0 0 1 0 2 4r0 1 6r0 1 8r0
  t
2d dry cask with detail 7 fuel assemblies
 0 0 1 20
 rcc 1 0 0 -1 0 0 229.60 44.55
 rcc 2 0 0 -1 0 0 230.87 45.82
 rcc 3 0 0 -1 0 0 233.41 48.36
 rcc 4 0 0 -1 0 0 235.95 50.90
 rcc 5 0 0 -1 0 0 238.49 53.44
 rcc 6 0 0 -1 0 0 242.30 57.25
 rcc 7 0 0 -2 0 0 2.0000 60.00
 rcc 8 0 0 -1 0 0 240.39 55.34
 rcc 9 0 0 -1 0 0 229.50 58.00
 rcc 10 0 0 -6 0 0 260    70
 rcc 11 0 0 0.01 0 0 182.87 .399
 rcc 12 0 0 0.01 0 0 182.87 .427
 rcc 13 0 0 0.01 0 0 182.87 .454
 rpp 14 -0.63 0.63 -0.63 0.63 .01 182.88
 rpp 15 -10.71 10.71 -10.71 10.71 .01 182.88
 rpp 16 -12.35 12.35 -12.35 12.35 .01 182.88
 rpp 17 -24.70 24.70 -37.05 -12.35 .01 182.88
 rpp 18 -37.05 37.05 -12.35 12.35 .01 182.88
 rpp 19 -24.70 24.70  12.35 37.05 .01 182.88
   end
 inv 1 -17 -18 -19 -7
 ss1 2 -1 -7 -9
 du1 3 -2 -7 -9
 du2 4 -3 -7 -9
 du3 5 -4 -7 -9
 ss2 6 -8 -7 -9
 alb 7 6
 exv 10  -6
 ss3 8 -5 -7 -9
 ss4 2 9 -1 -7
 du4 5 9 -2 -7
 ss5 6 9 -5 -7
 fue  11
 gap  12 -11
 cla  13 -12
 fcl  14 -13 -7
 uv1  10 -14
 ar1  15
 unt  16 -15
 uv2  10 -16
 ar2  17
 ar3  18
 ar4  19
   end
  1 2 3 4 5 6 1 1 7 8 8 8  11r1
 12r0  5r1  3r2  3r0
 1000 2 3 3 3 2 88 0 2 2 3 2 1 1000 4 1000 -1000 -1 1000 -1000 -2 -3 -4
 17 17 1  2 1 1  3 1 1  2 1 1  0  0
 289*1  7*2
 2r0
6** 10.0 3r0.0 -37.05 37.05 -37.05 37.05 0.0 182.88
```

Figure F9.C.13 MORSE-SGC sample problem 8 input data

```
8** 22r5.0 22r3.55 22r2.25 22r1.2 22r.65 22r.39 22r.27 22r10.
     22r.2 22r.142 22r.09 22r.048 22r.026 22r.0156 22r.0108 22r.4
    22r1. 22r.71 22r.45 22r.24 22r.13 22r.078 22r.054 22r2.
    11r.5 11r0. 11r.5 11r0. 11r.5 11r0. 11r.5 11r0. 44r0.
    11r.5 11r0. 11r.5 11r0.
10** 1. .71 .45 .24 .13 .078 .054 2.
 4.9338-4 1.9473-3 5.8223-3 1.9193-2 3.9752-2 5.1325-2 1.0834-1
 8.7299-2 2.1106-2 1.1541-1 2.0886-1 1.9255-1 1.3361-1 1.4261-2
 2.1043-5 1.5248-6 1.0053-7 5r0.0
 22r0.0
 3.3607-4 1.4566-3 4.7086-3 1.6604-2 3.6095-2 4.8189-2 1.0487-1
 8.6328-2 2.0664-2 1.1546-1 2.1259-1 1.9902-1 1.3943-1 1.4248-2
 8r0.0
22r0.0
12$$  5r1  4  5r2  2r3
13$$ 92235 92238 94239 94240 8016 40000 24000 25055 26000 28000
 42000 92235 92238
14** 1.6736-4 1.9602-2 1.0409-4 4.7534-5 3.9877-2 1.9350-2
 1.662-2 1.2-3 5.775-2 7.52-3 1.1-4 1.07-4 4.77-2
17** 8.611-5 7.295-4 2.007-3 1.009-2 2.516-2 5.07-2 .2123 .1581
  .03157 .1331 .1737 .1386 .0636 6.131-6
18** 107. 97.0 93.2 78.6 64.0 58.6 56.0 53.9 53.4 48.6 38.6 33.1
     25.3 8.89
  t
 neutron leakage
 neutron leakage
19** f0.0
20** f1.0
  t
end
```

Figure F9.C.13 (continued)

F9.D MORSE-SGC THEORY SECTION FROM MORSE-CG DOCUMENT, ORNL-4972

F9.D.1 INTEGRAL FORMS OF THE BOLTZMANN TRANSPORT EQUATION AND ITS ADJOINT

The purpose here is to derive a complete set of forward and adjoint integral transport equations in energy-group notation and to relate these equations to the Monte Carlo procedures used in the MORSE code.

The Boltzmann Transport Equation

The derivation begins with the general time-dependent integro-differential form of the Boltzmann transport equation, the derivation of which can be regarded as a bookkeeping process that sets the net storage of particles within a differential element of phase space $(d\bar{r}dEd\bar{\Omega})$ equal to the particle gains minus particle losses in $((d\bar{r}dEd\bar{\Omega}))$ and leads to the following familiar and useful form:

$$\frac{1}{v}\frac{\partial}{\partial t}\phi(\bar{r},E,\bar{\Omega},t) + \bar{\Omega}\cdot\nabla\phi(\bar{r},E,\bar{\Omega},t) + \Sigma_t(\bar{r},E)\,\phi(\bar{r},E,\bar{\Omega},t)$$

$$= S(\bar{r},E,\bar{\Omega},t) + \int\int dE'd\bar{\Omega}'\,\Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\,\phi(\bar{r},E',\bar{\Omega}',t)\,, \qquad \text{(F9.D.1)}$$

where

$(\bar{r},E,\bar{\Omega},t)$ denotes the general seven-dimensional phase space,

$\bar{r} =$ position variable,

$E =$ the particle's kinetic energy,

$v =$ the particle's speed corresponding to its kinetic energy E,

$\bar{\Omega} =$ a unit vector which describes the particle's direction of motion,

$t =$ time variable,

$\phi(\bar{r},E,\bar{\Omega},t) =$ the time-dependent angular flux,

$\phi(\bar{r},E,\bar{\Omega},t)\,dEd\bar{\Omega} =$ the number of particles that cross a unit area normal to the $\bar{\Omega}$ direction per unit time at time at the space point $\bar{r}$ and time t with energies in dE about E and with directions that lie within the differential solid angle $d\bar{\Omega}$ about the unit vector $\bar{\Omega}$,

$\frac{1}{v}\frac{\partial}{\partial t}\phi(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega} =$ net storage (gains minus losses) per unit volume and time at the space point $\bar{r}$ and time t of particles with energies in dE about E and with directions that lie in $d\bar{\Omega}$ about $\bar{\Omega}$,

$\bar{\Omega}\cdot\nabla\phi(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega} =$ net convective loss per unit volume and time at the space point $\bar{r}$ and time t of particles with energies in dE about E and directions that lie in $d\bar{\Omega}$ about $\bar{\Omega}$,

$\Sigma_t(\bar{r},E) =$ the total cross section at the space point $\bar{r}$ for particles of energy E,

$\Sigma_t(\bar{r},E)\,\phi(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega}$ = collision loss per unit volume and time at the space point $\bar{r}$ and time t of particles with energies in dE about E and directions that lie in $d\bar{\Omega}$ about $\bar{\Omega}$,

$\Sigma_s(\bar{r},E'\rightarrow E,\bar{\Omega}'\rightarrow\bar{\Omega})dEd\bar{\Omega}$ = the differential scattering cross section which describes the probability per unit path that a particle with an initial energy E′ and an initial direction $\bar{\Omega}'$ undergoes a scattering collision at $\bar{r}$ which places it into a direction that lies in $d\bar{\Omega}$ about $\bar{\Omega}$ with a new energy in dE about E,

$(\int\int\Sigma_s(\bar{r},E'\rightarrow E,\bar{\Omega}'\rightarrow\bar{\Omega})\,\phi(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}')\,dEd\bar{\Omega}$ = inscattering gain per unit volume and time at the space point $\bar{r}$ and time t of particles with energies in dE about E and directions which lie in $d\bar{\Omega}$ about $\bar{\Omega}$,

$S(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega}$ = source particles emitted per unit volume and time at the space point $\bar{r}$ and time t with energies in dE about E and directions which lie in $d\bar{\Omega}$ about $\bar{\Omega}$.

An effect of interest such as biological dose, energy deposition, or particle flux (denoted by λ) for a given problem can be expressed in terms of the flux field $\Phi(\bar{r},E,\bar{\Omega},t)$ and an appropriate response function $P^{\Phi}(\bar{r},E,\bar{\Omega},t)$ due to a unit angular flux and is given by

$$\lambda = \int\int\int\int P^{\Phi}(\bar{r},E,\bar{\Omega},t)\,\phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \ . \qquad \text{(F9.D.2)}$$

Consistent with the MORSE code, the energy dependence of Eq. (F9.D.1) will be represented in terms of energy groups which are defined such that

ΔE_g = energy width of the *gth* group,
g = 1 corresponds to the highest energy group,
g = G corresponds to the lowest energy group,

with the obvious constraint that

$$\sum_{g=1}^{G} \Delta E_g = \int_0^{E_o} dE = E_o, \text{ the maximum particle energy.}$$

A "group" form of Eq. (F9.D.1) is obtained by integrating each term with respect to the energy variable over the energy interval ΔE_g:

$$\frac{\partial}{\partial t}\int_{\Delta E_g}\frac{1}{v}\phi(\bar{r},E,\bar{\Omega},t)dE + \bar{\Omega}\cdot\nabla\int_{\Delta E_g}\phi(\bar{r},E,\bar{\Omega},t)dE + \int_{\Delta E_g}\Sigma_t(\bar{r},E)\,\phi(\bar{r},E,\bar{\Omega},t)dE$$

$$= \int_{\Delta E_g} S(\bar{r},E,\bar{\Omega},t)dE + \sum_{g'=g}^{1}\int_{\Delta E_{g'}}\int_{4\pi} dE'd\bar{\Omega}'\int_{\Delta E_g}\Sigma_s(\bar{r},E'\rightarrow E,\bar{\Omega}'\rightarrow\bar{\Omega})\,\phi(\bar{r},E',\bar{\Omega}',t)dE \ . \qquad \text{(F9.D.3)}$$

Equation (F9.D.3) provides the formal basis for the following group parameters:*

$\phi_g(\bar{r},\bar{\Omega},t)$ = time-dependent group angular flux,

$$= \int_{\Delta E_g} \phi(\bar{r},E,\bar{\Omega},t)dE \ , \tag{F9.D.4}$$

$\Sigma_t^g(\bar{r})$ = energy-averaged total cross section for the *gth* group,

$$\equiv \frac{\int_{\Delta E_g} \Sigma_t(\bar{r},E)\ \phi(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g} \phi(\bar{r},E,\bar{\Omega},t)dE} \tag{F9.D.5}$$

v_g = energy-averaged particle speed for the *gth* group,

$$\equiv \frac{\int_{\Delta E_g} \phi(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g} \frac{1}{v}\ \phi(\bar{r},E,\bar{\Omega},t)dE} \tag{F9.D.6}$$

$\Sigma_s^{g'\rightarrow g}(\bar{r},\bar{\Omega}'\rightarrow\bar{\Omega})$ = group g′ to group g scattering cross section,

$$\equiv \frac{\int_{\Delta E_{g'}} \int_{\Delta E_g} \Sigma_s(\bar{r},E'\rightarrow E,\bar{\Omega}'\rightarrow\bar{\Omega})\ \phi(\bar{r},E',\bar{\Omega}',t)dE'dE}{\int_{\Delta E_{g'}} \phi(\bar{r},E',\bar{\Omega}',t)dE'} \tag{F9.D.7}$$

$S_g(\bar{r},\bar{\Omega},t)$ = distribution of source particles for the *gth* group,

$$\equiv \int_{\Delta E_g} S(\bar{r},E,\bar{\Omega},t)dE \ . \tag{F9.D.8}$$

The group form of the Boltzmann equation expressed in terms of the aforedefined group parameters is given by

*These parameters will be referred to as forward-weighted group parameters.

$$\begin{aligned}&\frac{1}{v_g}\frac{\partial}{\partial t}\phi_g(\bar{r},\bar{\Omega},t) + \bar{\Omega}\cdot\nabla\phi_g(\bar{r},\bar{\Omega},t) + \Sigma_t^g(\bar{r})\,\phi_g(\bar{r},\bar{\Omega},t)\\ &= S_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\,\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})\,\phi_{g'}(\bar{r},\bar{\Omega}',t)\ ,\end{aligned} \qquad \text{(F9.D.9)}$$

where the summation over energy groups could be expanded over all g′ to allow for upscattering — not usually considered important in shielding problems.

Integral Flux Density Equation

The transformation of Eq. (F9.D.9) into an integral form is now considered. To accomplish this, the combination of the convection and storage terms are first expressed in terms of the spatial variable R which relates a fixed point in space ($\bar{r}$) to an arbitrary point ($\bar{r}'$), as shown below in Fig. F9.D.1

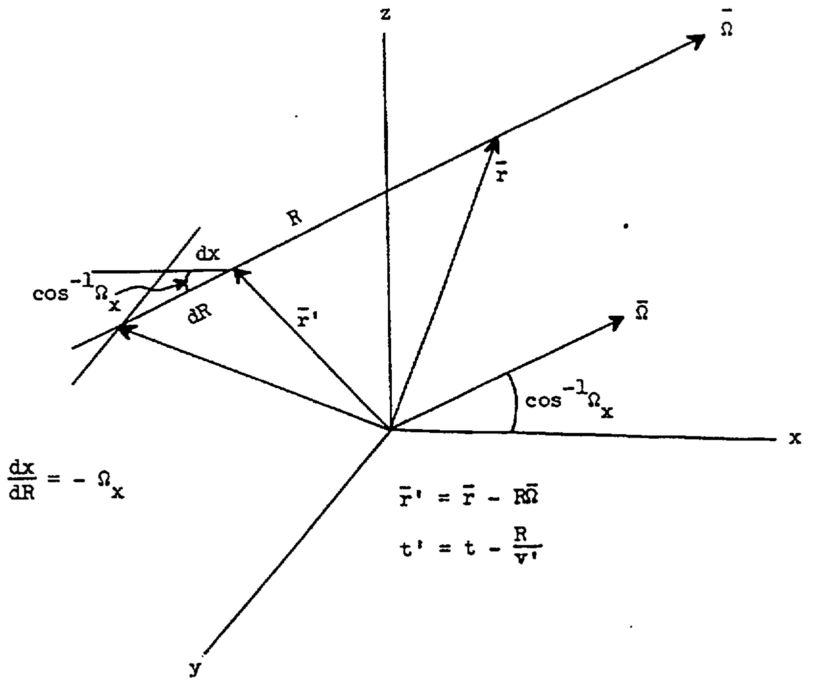


Figure F9.D.1 Relation of fixed point in space to arbitrary point in forward case.

The total derivative of the angular flux with respect to R is given by

$$\frac{d}{dR}\phi(\bar{r}',E,\bar{\Omega},t') = \frac{\partial x}{\partial R}\frac{\partial \phi}{\partial x} + \frac{\partial y}{\partial R}\frac{\partial \phi}{\partial y} + \frac{\partial z}{\partial R}\frac{\partial \phi}{\partial z} + \frac{\partial t}{\partial R}\frac{\partial \phi}{\partial t} ,$$

which, according to Fig. A.1 and noting that the particle's speed (v) is equal to (– dR/dt), can be rewritten as

$$\frac{d}{dR}\phi(\bar{r}',E,\bar{\Omega},t') = - \Omega_x\frac{\partial \phi}{\partial x} - \Omega_y\frac{\partial \phi}{\partial y} - \Omega_z\frac{\partial \phi}{\partial z} - \frac{1}{v}\frac{\partial \phi}{\partial t}$$

$$= - \bar{\Omega}\cdot\nabla\, \phi(\bar{r}',E,\bar{\Omega},t') - \frac{1}{v}\frac{\partial \phi}{\partial t} . \qquad \text{(F9.D.10)}$$

Equation (F9.D.10) can be expressed in group notation as

$$- \frac{d}{dR}\phi_g(\bar{r}',\bar{\Omega},t') = \frac{1}{v_g}\frac{\partial}{\partial t}\phi_g(\bar{r}',\bar{\Omega},t') + \bar{\Omega}\cdot\nabla\, \phi_g(\bar{r}',\bar{\Omega},t') . \qquad \text{(F9.D.11)}$$

Substitution of Eq. (F9.D.11) into Eq. (F9.D.9) with $\bar{r} \equiv \bar{r}'$ and $t \equiv t'$ yields

$$- \frac{d}{dR}\phi_g(\bar{r}',\bar{\Omega},t') + \Sigma_t^g(\bar{r}')\, \phi_g(\bar{r}',\bar{\Omega},t') = S_g(\bar{r}',\bar{\Omega},t')$$

$$+ \sum_{g'=g}^{I} \int_{4\pi} d\bar{\Omega}'\; \Sigma_s^{g'\to g}(\bar{r}',\bar{\Omega}'\to\bar{\Omega})\, \phi_{g'}(\bar{r}',\bar{\Omega}',t') . \qquad \text{(F9.D.12)}$$

The integrating factor,

$$e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'} ,$$

is introduced in the following manner:

$$\frac{d}{dR}\left(\phi_g(\bar{r}',\bar{\Omega},t')\, e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'} \right) = - e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}$$

$$\times \left(- \frac{d\phi_g}{dR} + \Sigma_t^g(\bar{r}')\, \phi_g(\bar{r}',\bar{\Omega},t') \right) . \qquad \text{(F9.D.13)}$$

Using Eq. (F9.D.13), Eq. (F9.D.12) can be rewritten as

$$-\frac{d}{dR}\left(\phi_g(\bar{r}',\bar{\Omega},t')\; e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}\right) = e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}$$

$$\times\left(S_g(\bar{r}',\bar{\Omega},t') + \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\; \Sigma_s^{g'\to g}(\bar{r}',\bar{\Omega}'\to\bar{\Omega})\; \phi_{g'}(\bar{r}',\bar{\Omega}',t')\right) . \qquad \text{(F9.D.14)}$$

Multiply Eq. (F9.D.14) by dR and integrate (R = 0 to R = ∞), then

$$\phi_g(\bar{r},\bar{\Omega},t) - \phi_g(\infty,\bar{\Omega},t_\infty)e^{-\int_0^\infty \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'} = \int_0^\infty dR\; e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}$$

$$\left(S_g(\bar{r} - R\bar{\Omega},\bar{\Omega},t - \frac{R}{v}) + \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\; \Sigma_s^{g'\to g}(\bar{r} - R\bar{\Omega},\bar{\Omega}'\to\bar{\Omega})\; \phi_{g'}(\bar{r}',\bar{\Omega}',t')\right) . \qquad \text{(F9.D.15)}$$

Require that

$$\left(\phi_g(\infty,\bar{\Omega},t_\infty)e^{-\int_0^\infty \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}\right) = 0 \qquad \text{(F9.D.16)}$$

and introduce the "optical thickness"

$$\beta_g(\bar{r},R,\bar{\Omega}) \equiv \int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR' , \qquad \text{(F9.D.17)}$$

and Eq. (F9.D.15) becomes

$$\phi_g(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\; e^{-\beta_g(\bar{r},R,\bar{\Omega})}\Bigg(S_g(\bar{r} - R\bar{\Omega},\bar{\Omega},t - R/v)$$

$$+ \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\; \Sigma_s^{g'\to g}(\bar{r} - R\bar{\Omega},\bar{\Omega}'\to\bar{\Omega})\; \phi_{g'}(\bar{r}',\bar{\Omega}',t')\Bigg) . \qquad \text{(F9.D.18)}$$

Equation (F9.D.18) will be referred to as the "Integral Flux Density Equation."

An effect of interest λ in group notation can be expressed as

$$\lambda_g = \int\int\int P_g^{\phi}(\bar{r},\bar{\Omega},t)\ \phi_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt\ , \tag{F9.D.19}$$

where

$P_g^{\phi}(\bar{r},\bar{\Omega},t)$ = the response function of the effect of interest due to a unit angular group flux (group g, $\bar{r}$, $\bar{\Omega}$, time t),

$$= \frac{\int_{\Delta E_g} P^{\Phi}(\bar{r},E,\bar{\Omega},t)\,\phi(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g} \Phi(\bar{r},E,\bar{\Omega},t)dE}$$

λ_g = that portion of the effect of interest associated with the *gth* energy group.

The λ_g are so defined that the total effect of interest λ is given by the summation

$$\lambda = \sum_{g=1}^{G} \lambda_g\ . \tag{F9.D.20}$$

Integral Event Density Equation

The "event density" $\psi_g(\bar{r},\bar{\Omega},t)$ describes the density of particles going into a collision and is related to the group angular flux in the following manner:

$$\psi_g(\bar{r},\bar{\Omega},t) \equiv \Sigma_t^g(\bar{r})\ \phi_g(\bar{r},\bar{\Omega},t)\ , \tag{F9.D.21}$$

where

$\psi_g(\bar{r},\bar{\Omega},t)d\bar{\Omega}$ = the number of collision events per unit volume and time at the space point $\bar{r}$ and time t experienced by particles having energies within the *gth* energy group and directions in $d\bar{\Omega}$ about $\bar{\Omega}$.

The defining equation for the event density is obtained by multiplying both sides of Eq. (F9.D.18) by the group total cross section $\Sigma_t^g(\bar{r})$ and identifying the product $\Sigma_t^g(\bar{r})\,\phi_g(\bar{r},\bar{\Omega},t)$ as the event density $\psi_g(\bar{r},\bar{\Omega},t)$:

$$\psi_g(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, \Sigma_t^g(\bar{r})\, e^{-\beta_g(\bar{r},R,\bar{\Omega})}\; S_g(\bar{r} - R\bar{\Omega},\bar{\Omega},t - R/v)$$

$$+ \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}' \frac{\Sigma_s^{g'\to g}(\bar{r} - R\bar{\Omega},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r}')} \psi_{g'}(\bar{r}',\bar{\Omega}',t') \quad . \qquad \text{(F9.D.22)}$$

Equation (F9.D.22) will be referred to as the "Integral Event Density Equation."

The effect of interest λ_g can be expressed in terms of the event density; consider Eq. (F9.D.19) rewritten as

$$\lambda_g = \int\int\int \frac{P_g^{\phi}(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} \Sigma_t^g(\bar{r})\, \phi_g(\bar{r},\bar{\Omega},t) d\bar{r} d\bar{\Omega} dt$$

$$= \int\int\int P_g^{\psi}(\bar{r},\bar{\Omega},t)\, \psi_g(\bar{r},\bar{\Omega},t) d\bar{r} d\bar{\Omega} dt \ , \qquad \text{(F9.D.23)}$$

where

$P_g^{\psi}(\bar{r},\bar{\Omega},t)$ = the response function of the effect of interest due to a particle that experiences an event at (group g, $\bar{r}$, $\bar{\Omega}$, time t),

$P_g^{\psi}(\bar{r},\bar{\Omega},t) = P_g^{\phi}(\bar{r},\bar{\Omega},t)/\Sigma_t^g(\bar{r})$

or

$$P_g^{\phi}(\bar{r},\bar{\Omega},t) = \Sigma_t^g(\bar{r})\, P_g^{\psi}(\bar{r},\bar{\Omega},t) \ . \qquad \text{(F9.D.24)}$$

Integral Emergent Particle Density Equation

Define the emergent particle density $\chi_g(\bar{r},\bar{\Omega},t)$ as the density of particles leaving a source or emerging from a real collision with phase space coordinates (group g, $\bar{r}$, $\bar{\Omega}$, t),

$$\chi_g(\bar{r},\bar{\Omega},t) = S_g(\bar{r},\bar{\Omega},t) + \sum_{g'} \int_{4\pi} d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})\, \phi_{g'}(\bar{r},\bar{\Omega}',t) \ . \qquad \text{(F9.D.25)}$$

Then Eq. (F9.D.18) can be written as

$$\phi_g(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, e^{-\beta_g(\bar{r},R,\bar{\Omega})} \chi_g(\bar{r}',\bar{\Omega},t') \ . \qquad \text{(F9.D.26)}$$

The "Integral Emergent Particle Density Equation" is obtained by substituting Eq. (F9.D.26) into Eq. (F9.D.25):

$$\begin{aligned}\chi_g(\bar{r},\bar{\Omega},t) &\\ &= S_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega}) \int_0^\infty dR\, e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')} \chi_{g'}(\bar{r}',\bar{\Omega}',t') \\ &= S_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\, \frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})} \int_0^\infty dR\, \Sigma_t^{g'}(\bar{r})\, e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')} \chi_{g'}(\bar{r}',\bar{\Omega}',t') \ . \qquad \text{(F9.D.27)}\end{aligned}$$

The effect of interest λ_g can also be expressed in terms of the emergent particle density

$$\lambda_g = \int\int\int P_g^{\chi}(\bar{r},\bar{\Omega},t)\ \chi_g(\bar{r},\bar{\Omega},t) d\bar{r} d\bar{\Omega} dt \ . \qquad \text{(F9.D.28)}$$

The response function $P_g^{\chi}(\bar{r},\bar{\Omega},t)$ is obtained by considering a particle that emerges from a collision at $\bar{r}$ with phase space coordinates (group g, $\bar{\Omega}$, time t). This particle will experience an event in dR about $\bar{r}' = \bar{r} + R\bar{\Omega}$ at time t′ = t + R/v with the probability

$$\Sigma_t^g(\bar{r}')\, e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'} dR \ ,$$

and the contribution of this event is the response function $P_g^{\psi}(\bar{r}',\bar{\Omega},t')$. The sum of all such contributions to the effect of interest is given by

$$\int_0^\infty dR\, \Sigma_t^g(\bar{r}')\, e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'} P_g^{\psi}(\bar{r}',\bar{\Omega},t') \ ,$$

and should be the same as a response function $P_g^{\chi}(\bar{r},\bar{\Omega},t)$ which is based on emergent particle density. This leads to the following relationship:

$$P_g^{\chi}(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, \Sigma_t^g(\bar{r}')\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} P_g^{\psi}(\bar{r}',\bar{\Omega},t') \ , \qquad \text{(F9.D.29)}$$

where

$P_g^{\chi}(\bar{r},\bar{\Omega},t)$ ≡ the response function (of the effect of interest due to a particle that emerges from a collision having the phase space coordinates (group g, $\bar{r}$, $\bar{\Omega}$, time t)

$$\beta_g^*(\bar{r},R,\bar{\Omega}) \equiv \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR' \ . \tag{F9.D.30}$$

Note that $\beta_t^*(\bar{r},R,\bar{\Omega})$ differs from the optical thickness $\beta_g(\bar{r},R,\bar{\Omega})$ as defined by Eq. (F9.D.17) in that the integration is performed in the positive $\bar{\Omega}$ direction and as such $\beta_t^*(\bar{r},R,\bar{\Omega})$ is the adjoint of $\beta_g(\bar{r},R,\bar{\Omega})$. $P_g^\chi(\bar{r},\bar{\Omega},t)$ can also be expressed in terms of $P_g^\phi(\bar{r},\bar{\Omega},t)$ by substituting Eq. (F9.D.24) into Eq. (F9.D.29), yielding

$$P_g^\chi(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} P_g^\phi(\bar{r}',\bar{\Omega},t') \ . \tag{F9.D.31}$$

Operator Notation and Summary of the Forward Equations

Define the transport integral operator

$$T_g(\bar{r}'\to\bar{r},\bar{\Omega}) \equiv \int_0^\infty dR\, \Sigma_t^g(\bar{r})\, e^{-\beta_g(\bar{r},R,\bar{\Omega})} \ , \tag{F9.D.32}$$

and the collision integral operator

$$C_{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega}) = \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}' \frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})} \ , \tag{F9.D.33}$$

which can be rewritten as

$$C_{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega}) = \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}' \left(\frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_s^{g'}(\bar{r})} \right) \left(\frac{\Sigma_s^{g'}(\bar{r})}{\Sigma_t^{g'}(\bar{r})} \right) \ , \tag{F9.D.34}$$

where

$$\Sigma_s^{g'}(\bar{r}) = \sum_g \int_{4\pi} d\bar{\Omega}\, \Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega}) \ . \tag{F9.D.35}$$

In Eq. (F9.D.34), $[\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})/\Sigma_s^{g'}(\bar{r})]$ is a normalized probability density function from which the selection of a new energy group and direction can be accomplished and $[\Sigma_s^{g'}(\bar{r})/\Sigma_t^{g'}(\bar{r})]$ is the nonabsorption probability.

Using the transport and collision integral operators, Eq. (F9.D.22) can be rewritten as

$$\psi_g(\bar{r},\hat{\Omega},t) = T_g(\bar{r}'\rightarrow\bar{r},\hat{\Omega})(S_g(\bar{r}',\hat{\Omega},t') + C_{g'\rightarrow g}(\bar{r}',\hat{\Omega}'\rightarrow\hat{\Omega})\,\psi_g(\bar{r}',\hat{\Omega}',t')) \ . \qquad \text{(F9.D.36)}$$

The term $T_g(\bar{r}',\bar{r},\hat{\Omega})S_g(\bar{r}',\hat{\Omega},t')$ can be identified as the "first collision source" and denoted by

$$S_c^{\,g}(\bar{r},\hat{\Omega},t) \equiv T_g(\bar{r}'\rightarrow\bar{r},\hat{\Omega})\,S_g(\bar{r}',\hat{\Omega},t') \ , \qquad \text{(F9.D.37)}$$

and the "Integral Event Density Equation" becomes'

$$\psi_g(\bar{r},\hat{\Omega},t) = S_c^{\,g}(\bar{r},\hat{\Omega},t) + T_g(\bar{r}'\rightarrow\bar{r},\hat{\Omega})\,C_{g'\rightarrow g}(\bar{r}',\hat{\Omega}'\rightarrow\hat{\Omega})\,\psi_g(\bar{r}',\hat{\Omega}',t') \ . \qquad \text{(F9.D.38)}$$

Using the relationship $\psi_g(\bar{r},\hat{\Omega},t) = \Sigma_t^g(\bar{r})\,\phi_g(\bar{r},\hat{\Omega},t)$, Eq. (F9.D.38) can be transformed into the "Integral Flux Density Equation:"

$$\phi_g(\bar{r},\hat{\Omega},t) = \frac{S_c^{\,g}(\bar{r},\hat{\Omega},t)}{\Sigma_t^g(\bar{r})} + T_g(\bar{r}'\rightarrow\bar{r},\hat{\Omega})\,C_{g'\rightarrow g}(\bar{r}',\hat{\Omega}'\rightarrow\hat{\Omega})\,\frac{\Sigma_t^{g'}(\bar{r}')}{\Sigma_t^g(\bar{r})}\,\phi_{g'}(\bar{r}',\hat{\Omega}',t') \ . \qquad \text{(F9.D.39)}$$

Finally, the integral operators are introduced into Eq. (F9.D.27) and the following form for the "Integral Emergent Particle Density Equation" is obtained:

$$\chi_g(\bar{r},\hat{\Omega},t) = S_g(\bar{r},\hat{\Omega},t) + C_{g'\rightarrow g}(\bar{r},\hat{\Omega}'\rightarrow\hat{\Omega})\,T_{g'}(\bar{r}'\rightarrow\bar{r},\hat{\Omega}')\,\chi_{g'}(\bar{r}',\hat{\Omega}',t') \ . \qquad \text{(F9.D.40)}$$

An examination of Eqs. (F9.D.38), (F9.D.39), and (F9.D.40) would reveal that either the "Integral Event Density Equation" or the "Integral Emergent Particle Density Equation" would provide a reasonable basis for a Monte Carlo random walk. Equation (F9.D.40) was selected for the MORSE code since the source particles would be introduced according to the natural distribution rather than the distribution of first collisions. However, note that after the introduction of the source particle, the subsequent random walk can be regarded in terms of either Eq. (F9.D.38) or Eq. (F9.D.40) with the particle's weight at a collision site being the weight before collision (WTBC) or the weight after collision (WATE), respectively.

The random walk based on the "Integral Emergent Particle Density Equation" would introduce a particle into the system according to the source function. The particle travels to the site of its first collision as determined by the transport kernel. Its weight is modified by the nonabsorption probability and a new energy group and flight direction are selected from the collision kernel. The transport and collision kernels are applied successively determining the particle's emergent phase space coordinates corresponding to the second, third, etc., collision sites until the random walk is terminated. Termination is caused by the reduction of the particle's weight below some cutoff value or because the particle escapes from that portion of phase space associated with a particular problem (for example, escape from the system, slowing down below an energy cutoff, or exceeding some arbitrarily specified age cutoff).

Random Walk Procedure

The actual implementation of the random walk procedure is accomplished by approximating the integrals implied in the collision and transport integral operators by the sum

$$\chi_g(\bar{r},\hat{\Omega},t) = \sum_{n=0}^{\infty} \chi_g^n(\bar{r},\hat{\Omega},t) \ , \qquad \text{(F9.D.41)}$$

where

$\chi_g^n(\bar{r},\hat{\Omega},t)d\hat{\Omega}$ = the emergent particle density of particles emerging from its nth collision and having phase space coordinates (group g, $\bar{r}$, $d\hat{\Omega}$ about $\hat{\Omega}$, time t),

$\chi_g^o(\bar{r},\hat{\Omega},t) = S_g(\bar{r},\hat{\Omega},t),$

$\chi_g^n(\bar{r},\hat{\Omega},t) = C_{g'\rightarrow g}(\bar{r},\hat{\Omega}'\rightarrow\hat{\Omega})\, T_{g'}(\bar{r}'\rightarrow\bar{r},\hat{\Omega}')\, \chi_g^{n-1}(\bar{r}',\hat{\Omega}',t')$.

Thus, the source coordinates (group g_o, $\bar{r}_o$, $\hat{\Omega}_o$, time t_o) are selected from $S_g(\bar{r},\hat{\Omega},t)$ and a flight distance R is picked $\Sigma_t^{g_o}(\bar{r})\, e^{-\beta_{g_o}(\bar{r},R,\hat{\Omega}_o)}$ to determine the site for the first collision $\bar{r}_1$ and the particle's age $t_1 = t_o + R/v_{g_o}$.

The probability of scattering is $\Sigma_s^{g_o}(\bar{r}_1)/\Sigma_t^{g_o}(\bar{r}_1)$. All particles are forced to scatter, and their weight is modified with the probability. A new group g_1 is selected according to the distribution

$$\frac{\int_{4\pi} d\hat{\Omega}\, \Sigma_s^{g_o\rightarrow g}(\bar{r}_1,\hat{\Omega}_o\rightarrow\hat{\Omega})}{\Sigma_s^{g_o}(\bar{r}_1)} \ ,$$

and then a new direction $\hat{\Omega}$ is determined from

$$\frac{\Sigma_s^{g_o\rightarrow g_1}(\bar{r}_1,\hat{\Omega}_o\rightarrow\hat{\Omega})}{\Sigma_s^{g_o\rightarrow g_1}(\bar{r}_1)} \ .$$

The process is repeated until the particle history is terminated. Contributions to the quantity of interest are estimated at appropriate points in the random walk (boundary crossings, before or after real collisions, etc.) using the particle's WATE and the estimator $P_g^{\chi}(\bar{r},\hat{\Omega},t)$.

Derivation of the Adjoint Integro-Differential Boltzmann Transport Equation

Consider a (as yet unspecified) function $\phi^*(\bar{r},E,\hat{\Omega},t)$ which exists over the same phase space and satisfies the same kind of boundary conditions satisfied by the forward angular flux $\phi(\bar{r},E,\hat{\Omega},t)$. Further, let an operator O^* be defined such that the following integral relationship is satisfied:

$$\int\!\!\int\!\!\int\!\!\int \phi^*(\bar{r},E,\hat{\Omega},t)\ 0\ \phi(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt$$

$$= \int\!\!\int\!\!\int\!\!\int \phi(\bar{r},E,\hat{\Omega},t)\ 0^*\ \phi^*(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt \ + \text{(Boundary Terms)} \ .$$

The 0^* operator will be referred to as the adjoint operator to the corresponding forward operator 0.

Multiply each term of the Boltzmann transport equation, Eq. (1), by the function $\phi^*(\bar{r},E,\bar{\Omega},t)$ and integrate the resultant equation (term by term) over all phase space:

$$\begin{aligned}&\iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\,\frac{1}{v}\frac{\partial}{\partial t}\,\phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \;+\; \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\\ &\quad\times\ \nabla\cdot\bar{\Omega}\ \phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \;+\; \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ \Sigma_t(\bar{r},E)\\ &\quad\times\ \phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \;=\; \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ S(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt\\ &\quad+\ \iiiint \phi^*(\bar{r},E,\bar{\Omega},t) \iint \Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\ \phi(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt\ . \qquad \text{(F9.D.42)}\end{aligned}$$

It can be shown that the following adjoint relationships are true for the conditions associated with a particle transport problem:

$$\begin{aligned}&\iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\,\frac{1}{v}\frac{\partial}{\partial t}\,\phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \;=\; -\iiiint \phi(\bar{r},E,\bar{\Omega},t)\,\frac{1}{v}\frac{\partial}{\partial t}\\ &\quad\times\ \phi^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \;+\; \left(\iiiint \frac{1}{v}\frac{\partial}{\partial t}\,\Phi\ \Phi^*d\bar{r}dEd\bar{\Omega}dt\right)_{\text{Boundary Term}} \qquad \text{(F9.D.43)}\end{aligned}$$

$$\begin{aligned}&\iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ \Sigma_t(\bar{r},E)\ \phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \;=\; \iiiint \phi(\bar{r},E,\bar{\Omega},t)\\ &\quad\times\ \Sigma_t(\bar{r},E)\ \phi^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt\ , \qquad \text{(F9.D.44)}\end{aligned}$$

$$\begin{aligned}&\iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ \nabla\cdot\bar{\Omega}\ \phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \;=\; -\iiiint \phi(\bar{r},E,\bar{\Omega},t)\\ &\quad\times\ \nabla\cdot\bar{\Omega}\ \phi^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \;+\; \left(\iiiint \Phi\ \Phi^*\bar{\Omega}\cdot\ d\bar{r}dEd\bar{\Omega}dt\right)_{\text{Boundary Term}}\end{aligned}$$

$$\begin{aligned}&\iiiint \phi^*(\bar{r},E,\bar{\Omega},t) \iint \Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\ \phi(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt\\ &\quad=\ \iiiint \phi(\bar{r},E,\bar{\Omega},t) \iint \Sigma_s(\bar{r},E\to E',\bar{\Omega}\to\bar{\Omega}')\ \phi^*(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt\ . \qquad \text{(F9.D.46)}\end{aligned}$$

The boundary terms that occur in Eqs. (F9.D.43) and (F9.D.45) may be made to vanish while conforming to the natural characteristics of the system under analysis. For example, the extent of the time domain can be defined such that initial and final values of Φ and/or Φ^* are zero [and the boundary term of Eq. (F9.D.43) vanishes]. Also, the surface within which the spatial domain of phase space is contained can be so located that the combination [$\Phi\ \Phi^*$] is zero everywhere on that surface [and the boundary term of Eq. (F9.D.45) vanishes]. For most Monte Carlo analyses, the elimination of the boundary terms in no way restricts the generality of the solution obtained.

Using the adjoint relationships given by Eqs. (F9.D.43) through (F9.D.46), and presuming that the boundary terms vanish, Eq. (F9.D.42) can be rewritten as

$$\begin{aligned}
& -\int\!\!\int\!\!\int\!\!\int \phi(\bar{r},E,\hat{\Omega},t)\frac{1}{v}\frac{\partial}{\partial t}\phi^*(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt - \int\!\!\int\!\!\int\!\!\int \phi(\bar{r},E,\hat{\Omega},t) \\
& \times \nabla\cdot\hat{\Omega}\,\phi^*(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt + \int\!\!\int\!\!\int\!\!\int \phi(\bar{r},E,\hat{\Omega},t)\,\Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt \\
& = \int\!\!\int\!\!\int\!\!\int \phi(\bar{r},E,\hat{\Omega},t)\,S^*(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt + \int\!\!\int\!\!\int\!\!\int \phi(\bar{r},E,\hat{\Omega},t) \\
& \times \int\!\!\int \Sigma_s(\bar{r},E\to E',\hat{\Omega}\to\hat{\Omega}')\,\phi^*(\bar{r},E',\hat{\Omega}',t)dE'd\hat{\Omega}'d\bar{r}dEd\hat{\Omega}dt\ ,
\end{aligned} \tag{F9.D.47}$$

where the adjoint source term $S^*(\bar{r},E,\hat{\Omega},t)$ is defined such that

$$\begin{aligned}
& \int\!\!\int\!\!\int\!\!\int \phi(\bar{r},E,\hat{\Omega},t)\,S^*(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt \\
& \quad = \int\!\!\int\!\!\int\!\!\int \phi^*(\bar{r},E,\hat{\Omega},t)\,S(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt\ .
\end{aligned} \tag{F9.D.48}$$

Noting that the forward flux $\phi(\bar{r},E,\hat{\Omega},t)$ can be factored from each term, Eq. (F9.D.47) can be rearranged as follows:

$$\begin{aligned}
& \int\!\!\int\!\!\int\!\!\int \phi(\bar{r},E,\hat{\Omega},t) \quad -\frac{1}{v}\frac{\partial}{\partial t}\phi^*(\bar{r},E,\hat{\Omega},t)\ -\ \nabla\cdot\hat{\Omega}\,\phi^*(\bar{r},E,\hat{\Omega},t) \\
& \quad + \Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\hat{\Omega},t)\ -\ S^*(\bar{r},E,\hat{\Omega},t)\ -\ \int\!\!\int \Sigma_s(\bar{r},E\to E',\hat{\Omega}\to\hat{\Omega}') \\
& \quad \times\ \phi^*(\bar{r},E',\hat{\Omega}',t)dE'd\hat{\Omega}' \quad d\bar{r}dEd\hat{\Omega}dt\ =\ 0\ .
\end{aligned} \tag{F9.D.49}$$

It is required that the forward angular flux $\phi(\bar{r},E,\hat{\Omega},t)$ correspond to nontrivial physical situations [i.e., $\phi(\bar{r},E,\hat{\Omega},t) > 0$] over at least some portion of phase space. The observation is made that $\phi^*(\bar{r},E,\hat{\Omega},t)$ is still essentially undefined and that many functions $\phi^*(\bar{r},E,\hat{\Omega},t)$ probably satisfy Eq. (F9.D.49). At this point, $\phi^*(\bar{r},E,\hat{\Omega},t)$ is defined to be that function that satisfies the following equation:

$$\begin{aligned}
& -\frac{1}{v}\frac{\partial}{\partial t}\phi^*(\bar{r},E,\hat{\Omega},t)\ -\ \nabla\cdot\hat{\Omega}\,\Phi^*(\bar{r},E,\hat{\Omega},t)\ +\ \Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\hat{\Omega},t)\ -\ S^*(\bar{r},E,\hat{\Omega},t) \\
& \quad -\int\!\!\int \Sigma_s(\bar{r},E\to E',\hat{\Omega}\to\hat{\Omega}')\,\phi^*(\bar{r},E',\hat{\Omega}',t)dE'd\hat{\Omega}'\ =\ 0\ .
\end{aligned}$$

This condition also satisfies Eq. (F9.D.49) exactly and provides the following $\phi^*(\bar{r},E,\hat{\Omega},t)$-defining integro-differential equation:

$$-\frac{1}{v}\frac{\partial}{\partial t}\phi^*(\bar{r},E,\bar{\Omega},t)\ -\ \nabla\cdot\bar{\Omega}\,\phi^*(\bar{r},E,\bar{\Omega},t)\ +\ \Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\bar{\Omega},t)$$

$$= S^*(\bar{r},E,\bar{\Omega},t)\ +\ \int\!\!\int \Sigma_s(\bar{r},E\rightarrow E',\bar{\Omega}\rightarrow\bar{\Omega}')\,\phi^*(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'\ , \qquad \text{(F9.D.50)}$$

which is commonly called the "Adjoint Integro-Differential Boltzmann Equation." However, it will not be the practice here to refer to the function $\phi^*(\bar{r},E,\bar{\Omega},t)$ as the adjoint flux; consistent terminology will be introduced later in this section.

At this point, two procedures for defining and calculating group adjoint fluxes are considered. One method involves integrating each term of Eq. (F9.D.50) over the energy interval ΔE_g, which leads to the following group equations:

$$-\frac{1}{\hat{v}_g}\frac{\partial}{\partial t}\hat{\phi}_g^*(\bar{r},\bar{\Omega},t)\ -\ \nabla\cdot\bar{\Omega}\,\hat{\phi}_g^*(\bar{r},\bar{\Omega},t)\ +\ \hat{\Sigma}_t^g(\bar{r})\,\hat{\phi}_g^*(\bar{r},\bar{\Omega},t)$$

$$= S_g^*(\bar{r},\bar{\Omega},t)\ +\ \sum_{g'=g}^{G}\int d\bar{\Omega}'\,\hat{\Sigma}_s^{g\rightarrow g'}(\bar{r},\bar{\Omega}\rightarrow\bar{\Omega}')\,\hat{\phi}_{g'}^*(\bar{r},\bar{\Omega}',t)\ , \qquad \text{(F9.D.51)}$$

$g = 1,2,...G$

where

$$\hat{\phi}_g^*(\bar{r},\bar{\Omega},t)\ =\ \frac{1}{\Delta E_g}\int_{\Delta E_g}\phi^*(\bar{r},E,\bar{\Omega},t)dE\ , \qquad \text{(F9.D.52)}$$

$$\hat{v}_g\ =\ \frac{\int_{\Delta E_g}\phi^*(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g}\frac{1}{v}\,\phi^*(\bar{r},E,\bar{\Omega},t)dE}\ , \qquad \text{(F9.D.53)}$$

$$\hat{\Sigma}_t^g(\bar{r})\ \equiv\ \frac{\int_{\Delta E_g}\Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g}\phi^*(\bar{r},E,\bar{\Omega},t)dE}\ , \qquad \text{(F9.D.54)}$$

$$\hat{\Sigma}_s^{g\to g'}(\bar{r},\hat{\Omega}\to\hat{\Omega}') \equiv \frac{\int\limits_{\Delta E_g}\int\limits_{\Delta E_{g'}} \Sigma_s(\bar{r},E\to E',\hat{\Omega}\to\hat{\Omega}')\,\phi^*(\bar{r},E',\hat{\Omega}',t)dE'dE}{\int\limits_{\Delta E_{g'}} \phi^*(\bar{r},E',\hat{\Omega}',t)dE'} \ , \qquad \text{(F9.D.55)}$$

$$\hat{S}_g^*(\bar{r},\hat{\Omega},t) \equiv \frac{1}{\Delta E_g}\int\limits_{\Delta E_g} S^*(\bar{r},E,\hat{\Omega},t)dE \ . \qquad \text{(F9.D.56)}$$

The $\hat{\Sigma}_t^g(\bar{r})$, $\hat{\Sigma}_s^{g\to g'}(\bar{r},\hat{\Omega}\to\hat{\Omega}')$, and $\hat{v}_g$ are adjoint-weighted group parameters and their use in the solution of Eq. (F9.D.51) provides group adjoint fluxes defined by Eq. (F9.D.52) where $\Phi^*(\bar{r},E,\hat{\Omega},t)$ represents the solution of Eq. (F9.D.50).

Another approach for defining group adjoint fluxes is to directly devise the equation that is adjoint to the group form of the Boltzmann equation [Eq. (9)]. The group adjoint equation so obtained* is given by

$$-\frac{1}{v_g}\frac{\partial}{\partial t}\phi_g^*(\bar{r},\hat{\Omega},t) - \nabla\cdot\hat{\Omega}\phi_g^*(\bar{r},\hat{\Omega},t) + \Sigma_t^g(\bar{r})\,\phi_g^*(\bar{r},\hat{\Omega},t)$$

$$= S_g^*(\bar{r},\hat{\Omega},t) + \sum_{g'=g}^{G}\int d\hat{\Omega}'\,\Sigma_s^{g\to g'}(\bar{r},\hat{\Omega}\to\hat{\Omega}')\,\phi_{g'}^*(\bar{r},\hat{\Omega}',t) \ , \qquad \text{(F9.D.57)}$$

$$g = 1,2,...G$$

where v_g, $\Sigma_t^g(\bar{r})$ are forward-weighted group parameters identical to those which occur in Eq. (F9.D.9) and the matrix $\Sigma_s^{g\to g'}(\bar{r},\hat{\Omega}\to\hat{\Omega}')$ is simply the transposition of the forward-weighted group-to-group differential scattering cross-section matrix.

The group adjoint fluxes $\Phi_g^*(\bar{r},\hat{\Omega},t)$ which represent the solution of Eq. (F9.D.57) are adjoint to the group fluxes Φ_g and do not necessarily assume the same values as the group adjoint fluxes $\hat{\Phi}_g^*(\bar{r},\hat{\Omega},t)$, i.e.,

$$\Phi_g^*(\bar{r},\hat{\Omega},t) \neq \frac{1}{\Delta E_g}\int\limits_{\Delta E_g} \Phi^*(\bar{r},E,\hat{\Omega},t)dE \ .$$

This follows since $\Sigma_t^g(\bar{r})$, $\Sigma_s^{g\to g'}(\bar{r},\hat{\Omega}\to\hat{\Omega}')$, and v_g are, in general, different from the adjoint weighted values. Usually forward-weighted group parameters, as implied by Eq. (F9.D.57), are used in MORSE. However, other weighting schemes, such as adjoint or adjoint and forward, deserve consideration when cross-section

*The derivation of Eq. (F9.D.57) is not presented here because of its similarity with the previous derivation of Eq. (F9.D.50); the integrals over energy are simply replaced by appropriate group summations.

weighting is a problem. When a sufficiently fine group structure is employed, the group parameters become less sensitive to the weighting scheme and the corresponding group adjoint fluxes are also nearly the same.

Integral Point-Value Equation

Equation (F9.D.57) is now transformed into an integral form following essentially the same procedures used with the forward equations. As shown below in Fig. F9.D.2, let $\bar{r}' = \bar{r} + R\bar{\Omega}$ rather than $\bar{r}' = \bar{r} - R\bar{\Omega}$ as was the convention with the forward equations. The total derivative of $\phi_g^*(\bar{r}',\bar{\Omega},t)$ with respect to R is given by

$$\frac{d}{dR}\,\phi_g^*(\bar{r}',\bar{\Omega},t') = \frac{1}{v_g}\frac{\partial}{\partial t}\,\phi_g^*(\bar{r}',\bar{\Omega},t') + \nabla\cdot\bar{\Omega}\,\phi_g^*(\bar{r}',\bar{\Omega},t')\ . \qquad \text{(F9.D.58)}$$

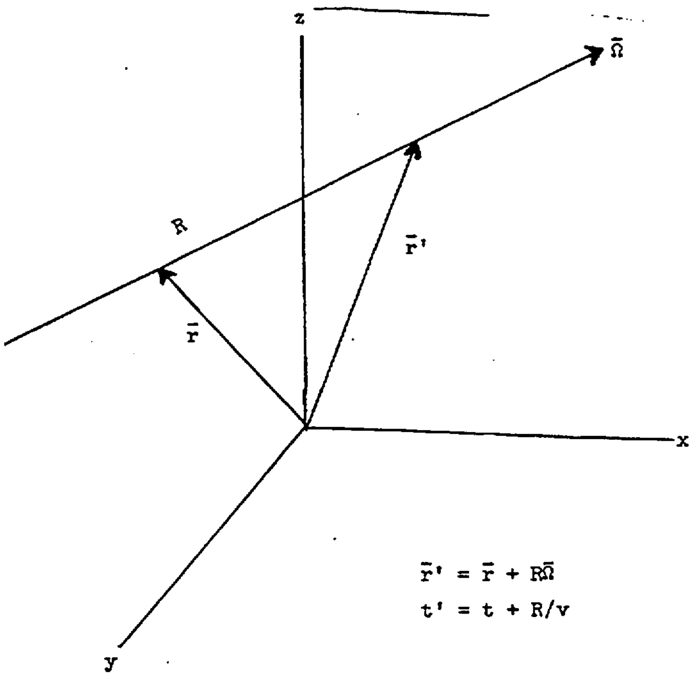


Figure F9.D.2 Relation of fixed point in space to arbitrary point for adjoint case.

Use of the integrating factor $e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'}$ provides the following relationship:

$$\frac{d}{dR}\left\{\phi_g^*(\bar{r}',\bar{\Omega},t')\,e^{-\int_0^R \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'}\right\}$$

$$= \frac{d\phi_g^*}{dR}(\bar{r}',\bar{\Omega},t')\,e^{-\int_0^R \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'} \quad - \Sigma_t^g(\bar{r}')\,\phi_g^*(\bar{r}',\bar{\Omega},t')$$

$$\times\, e^{-\int_0^R \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'} = e^{-\int_0^R \Sigma_t^g(\bar{r}'+R'\bar{\Omega})dR'}$$

$$\times\left(\frac{d}{dR}\,\phi_g^*(\bar{r}',\bar{\Omega},t') - \Sigma_t^g(\bar{r}')\,\phi_g^*(\bar{r}',\bar{\Omega},t')\right) . \qquad \text{(F9.D.59)}$$

Equation (F9.D.59), together with Eq. (F9.D.58), can be arranged to give

$$\left(-\frac{1}{v_g}\frac{\partial}{\partial t}\phi_g^*(\bar{r}',\bar{\Omega},t') - \nabla\cdot\bar{\Omega}\,\phi_g^*(\bar{r}',\bar{\Omega},t') + \Sigma_t^g(\bar{r}')\,\phi_g^*(\bar{r}',\bar{\Omega},t')\right)$$

$$= e^{+\int_0^R \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'}\,\frac{d}{dR}\left(\phi_g^*(\bar{r}',\bar{\Omega},t')\,e^{-\int_0^R \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'}\right) . \qquad \text{(F9.D.60)}$$

Note that Eq. (F9.D.60) is identically the left-hand side of Eq. (F9.D.57), which can now be rewritten as

$$-\frac{d}{dR}\left(\phi_g^*(\bar{r}',\bar{\Omega},t')\,e^{-\int_0^R \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'}\right) = e^{-\int_0^R \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'}$$

$$\times\left\{S_g^*(\bar{r}',\bar{\Omega},t') + \sum_{g'=g}^{G}\int d\bar{\Omega}'\,\Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')\,\phi_g^*(\bar{r}',\bar{\Omega}',t')\right\} . \qquad \text{(F9.D.61)}$$

Integrate Eq. (F9.D.61) from R = 0 to R = ∞ and assume that

$$\left\{\phi_g^*(\infty,\bar{\Omega},t_\infty)\,e^{-\int_0^\infty \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'}\right\} \equiv 0 \; ; \qquad \text{(F9.D.62)}$$

then the following integral expression for $\phi_g^*(\bar{r},\bar{\Omega},t)$ is obtained:

$$\phi_g^*(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})}\{S_g^*(\bar{r} + R\bar{\Omega},\bar{\Omega},t + R/v_g) + \sum_{g'} \int d\bar{\Omega}'\, \Sigma_s^{g\to g'}(\bar{r} + R\bar{\Omega},\bar{\Omega}\to\bar{\Omega}')\, \phi_{g'}^*(\bar{r}',\bar{\Omega}',t')\} \ . \tag{F9.D.63}$$

Equation (F9.D.63) contains the adjoint optical thickness $\beta_g^*(\bar{r},R,\bar{\Omega})$, which was defined earlier by Eq. (F9.D.30) as

$$\beta_g^*(\bar{r},R,\bar{\Omega}) \equiv \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR' \ .$$

Redefine the source term as

$$S_{Tg}^*(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})}\, S_g^*(\bar{r} + R\bar{\Omega},\bar{\Omega},t + R/v_g) \ , \tag{F9.D.64}$$

and Eq. (F9.D.63) can be rewritten as

$$\begin{aligned}\phi_g^*(\bar{r},\bar{\Omega},t) &= S_{Tg}^*(\bar{r},\bar{\Omega},t) + \int_0^\infty dR\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} \sum_{g'} \int d\bar{\Omega}'\, \Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}') \\ &\times \phi_{g'}^*(\bar{r}',\bar{\Omega}',t') = S_{Tg}^*(\bar{r},\bar{\Omega},t) + \int_0^\infty dR\, \Sigma_t^g(\bar{r} + R\bar{\Omega})\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} \\ &\times \sum_{g'} \int d\bar{\Omega}' \frac{\Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')}{\Sigma_t^g(\bar{r}')}\, \phi_{g'}^*(\bar{r}',\bar{\Omega}',t') \ ,\end{aligned} \tag{F9.D.65}$$

and in terms of the transport and collision operators, Eq. (F9.D.65) becomes

$$\phi_g^*(\bar{r},\bar{\Omega},t) = S_{Tg}^*(\bar{r},\bar{\Omega},t) + T_g(\bar{r}\to\bar{r}',\bar{\Omega})\, C_{g\to g'}(\bar{\Omega}\to\bar{\Omega}',\bar{r}')\, \phi_{g'}^*(\bar{r}',\bar{\Omega}',t') \ . \tag{F9.D.66}$$

A comparison of Eq. (F9.D.66) with Eqs. (F9.D.38), (F9.D.39), and (F9.D.40) reveals that the function $\phi_g^*(\bar{r},\bar{\Omega},t)$ as defined by Eq. (F9.D.66) is adjoint to the emergent particle density $\chi_g(\bar{r},\bar{\Omega},t)$ as defined by Eq. (F9.D.40). Therefore, let $\phi_g^*(\bar{r},\bar{\Omega},t)$ be denoted by $\chi_g^*(\bar{r},\bar{\Omega},t)$, and Eq. (F9.D.66) becomes

$$\chi_g^*(\bar{r},\bar{\Omega},t) = S_{Tg}^*(\bar{r},\bar{\Omega},t) + T_g(\bar{r}\to\bar{r}',\bar{\Omega})\, C_{g\to g'}(\bar{\Omega}\to\bar{\Omega}',\bar{r}')\, \chi_{g'}^*(\bar{r}',\bar{\Omega}',t') \ . \tag{F9.D.67}$$

The nature of $\chi^*_g(\bar{r},\hat{\Omega},t)$ will depend on $S^*_{Tg}(\bar{r},\hat{\Omega},t)$ — how or on what basis should $S^*_{Tg}(\bar{r},\hat{\Omega},t)$ be specified? If $S^*(\bar{r},E,\hat{\Omega},t)$ is set equal to $P^{\Phi}(\bar{r},E,\hat{\Omega},t)$ (the response function of the effect of interest λ due to a unit angular flux), then

$$\int\!\!\int\!\!\int\!\!\int \phi(\bar{r},E,\hat{\Omega},t)\, S^*(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt = \int\!\!\int\!\!\int\!\!\int \phi(\bar{r},E,\hat{\Omega},t) \times P^{\Phi}(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt = \lambda \ . \tag{F9.D.68}$$

According to Eq. (F9.D.48), the effect of interest λ would also be given by

$$\lambda = \int\!\!\int\!\!\int\!\!\int \phi^*(\bar{r},E,\hat{\Omega},t)\, S(\bar{r},E,\hat{\Omega},t)d\bar{r}dEd\hat{\Omega}dt \ . \tag{F9.D.69}$$

The effect of interest as given by Eq. (F9.D.69) can also be expressed in group notation

$$\begin{aligned}\lambda &= \sum_g \int\!\!\int\!\!\int \hat{\Phi}^*_g(\bar{r},\hat{\Omega},t)\, \hat{S}_g(\bar{r},\hat{\Omega},t)d\bar{r}d\hat{\Omega}dt \\ &= \sum_g \int\!\!\int\!\!\int \Phi_g(\bar{r},\hat{\Omega},t)\, \hat{S}^*_g(\bar{r},\hat{\Omega},t)d\bar{r}d\hat{\Omega}dt \ ,\end{aligned}$$

where

$\hat{\Phi}^*_g(\bar{r},\hat{\Omega},t)$ is the group adjoint flux corresponding to the adjoint weighted group parameters,

$$\hat{S}_g(\bar{r},\hat{\Omega},t) = \frac{\int_{\Delta E_g} \Phi^*(\bar{r},E,\hat{\Omega},t)S(\bar{r},E,\hat{\Omega},t)dE}{\hat{\Phi}^*_g(\bar{r},\hat{\Omega},t)}$$

$$= \frac{\int_{\Delta E_g} P^{\Phi}(\bar{r},E,\hat{\Omega},t)\ \Phi(\dot{r},E,\hat{\Omega},t)dE}{\Phi_g(\bar{r},\hat{\Omega},t)} = P^{\Phi}_g(\bar{r},\hat{\Omega},t) \ .$$

However, as noted earlier, usually forward-weighted group parameters are input to MORSE, and the group adjoint fluxes $\Phi^*_g(\bar{r},\hat{\Omega},t)$ are calculated. As a direct consequence of the derivation of the $\Phi^*_g(\bar{r},\hat{\Omega},t)$ defining equation, Eq. (F9.D.57), the effect of interest for the g^th^ group is also given by

$$\begin{aligned}\lambda^*_g &= \int\!\!\int\!\!\int \Phi^*_g(\bar{r},\hat{\Omega},t)\, S_g(\bar{r},\hat{\Omega},t)d\bar{r}d\hat{\Omega}dt & \lambda &= \sum_g \lambda^*_g \\ \lambda_g &= \int\!\!\int\!\!\int \Phi_g(\bar{r},\hat{\Omega},t)\, S^*_g(\bar{r},\hat{\Omega},t)d\bar{r}\,d\hat{\Omega}dt \ , & &= \sum_g \lambda_g \ ,\end{aligned} \tag{F9.D.70}$$

where

$\Phi^*_g(\bar{r},\hat{\Omega},t)$ is the group adjoint flux corresponding to the forward-weighted group parameters,

$$S_g(\bar{r},\bar{\Omega},t) = \int\limits_{\Delta E_g} S(\bar{r},E,\bar{\Omega},t)dE \ ,$$

$$S_g^*(\bar{r},\bar{\Omega},t) = P_g^{\Phi}(\bar{r},\bar{\Omega},t) \ . \qquad \text{(F9.D.71)}$$

The derivation from this point on will implicitly assume forward-weighted group parameters. However, the results can, with slight modification, be made to correspond to the adjoint-weighted group parameters. Substitution of Eq. (F9.D.71) into Eq. (F9.D.64) yields

$$S_{Tg}^*(\bar{r},\bar{\Omega},t) = \int dR\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} P_g^{\phi}(\bar{r}',\bar{\Omega},t') \ , \qquad \text{(F9.D.72)}$$

and according to Eqs. (F9.D.24) and (F9.D.29), Eq. (F9.D.72) can be rewritten as Eqs. (F9.D.73) and (F9.D.74), respectively:

$$S_{Tg}^*(\bar{r},\bar{\Omega},t) = \int dR\, \Sigma_t^g(\bar{r}')\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} P_g^{\psi}(\bar{r}',\bar{\Omega},t') \qquad \text{(F9.D.73)}$$

and

$$S_{Tg}^*(\bar{r},\bar{\Omega},t) = P_g^{\chi}(\bar{r},\bar{\Omega},t) \ . \qquad \text{(F9.D.74)}$$

Substitution of Eq. (F9.D.73) into Eq. (F9.D.67) and Eq. (F9.D.74) into Eq. (F9.D.67) yields the following forms for the "Integral Point-Value Equation:"

$$\chi_g^*(\bar{r},\bar{\Omega},t) = T_g(\bar{r}\rightarrow\bar{r}',\bar{\Omega})\,\{P_g^{\psi}(\bar{r}',\bar{\Omega},t') + C_{g\rightarrow g'}(\bar{r}',\bar{\Omega}\rightarrow\bar{\Omega}')\,\chi_{g'}^*(\bar{r}',\bar{\Omega}',t')\} \qquad \text{(F9.D.75)}$$

and

$$\chi_g^*(\bar{r},\bar{\Omega},t) = P_g^{\chi}(\bar{r},\bar{\Omega},t) + T_g(\bar{r}\rightarrow\bar{r}',\bar{\Omega})\,C_{g\rightarrow g'}(\bar{r}',\bar{\Omega}\rightarrow\bar{\Omega}')\,\chi_{g'}^*(\bar{r}',\bar{\Omega}',t') \ . \qquad \text{(F9.D.76)}$$

Integral Event-Value Equation

At this point, let us introduce a value function based on the event density and relate this quantity to the point-value function by considering a particle leaving a collision at $\bar{r}$ with phase-space coordinates (group g, $\bar{\Omega}$, time t). The value of this particle to the effect of interest is the point-value function $\chi_g^*(\bar{r},\bar{\Omega},t)$. This particle will experience an event in dR about $\bar{r}' = \bar{r} + R\bar{\Omega}$, and the probability $[\Sigma_t^g(\bar{r}')\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} dR]$ and the value of this event (to the effect of interest) will be referred to as the "event-value" and be denoted by $W_g(\bar{r}',\bar{\Omega},t')$. The "event-value" $W_g(\bar{r}',\bar{\Omega},t')$ is defined as the value (to the effect of interest) of having an event at $\bar{r}'$ with an incoming particle which has phase-space coordinates (group g, $\bar{\Omega}$, time t'). The sum of all such contributions to the effect of interest is given by

$$\int_0^\infty dR\, \Sigma_t^g(\bar{r}')\, e^{-\beta_g^*(\bar{r},R,\hat{\Omega})} W_g(\bar{r}',\hat{\Omega},t')\ ,$$

and, if the event-value function is properly defined, should equal the point-value function; that is,

$$\chi_g^*(\bar{r},\hat{\Omega},t) = \int_0^\infty dR\, \Sigma_t^g(\bar{r}')\, e^{-\beta_g^*(\bar{r},R,\hat{\Omega})} W_g(\bar{r}',\hat{\Omega},t') \qquad \text{(F9.D.77)}$$

or

$$\chi_g^*(\bar{r},\hat{\Omega},t) = T_g(\bar{r}\rightarrow\bar{r}',\hat{\Omega})\, W_g(\bar{r}',\hat{\Omega},t')\ . \qquad \text{(F9.D.78)}$$

A comparison of Eq. (F9.D.78) with Eq. (F9.D.75) would show that $W_g(\bar{r},\hat{\Omega},t)$ can be identified as

$$W_g(\bar{r},\hat{\Omega},t) = P_g^{\psi}(\bar{r},\hat{\Omega},t) + C_{g\rightarrow g'}(\bar{r},\hat{\Omega}\rightarrow\hat{\Omega}')\, \chi_{g'}^*(\bar{r},\hat{\Omega}',t)\ , \qquad \text{(F9.D.79)}$$

and substitution of Eq. (F9.D.78) into Eq. (F9.D.79) yields the defining equation for the "Event-Value Function"

$$W_g(\bar{r},\hat{\Omega},t) = P_g^{\psi}(\bar{r},\hat{\Omega},t) + C_{g\rightarrow g'}(\bar{r},\hat{\Omega}\rightarrow\hat{\Omega}')\, T_{g'}(\bar{r}\rightarrow\bar{r}',\hat{\Omega}')\, W_{g'}(\bar{r}',\hat{\Omega}',t')\ . \qquad \text{(F9.D.80)}$$

Equation (F9.D.80) will be referred to as the "Integral Event-Value Equation." A comparison of Eq. (F9.D.80) with Eq. (F9.D.38) would show that the event-value function $W_g(\bar{r},\hat{\Omega},t)$ is adjoint to the event density $\psi_g(\bar{r},\hat{\Omega},t)$. Therefore, the effect of interest is given by

$$\lambda = \sum_g \int\!\!\int\!\!\int S_c^g(\bar{r},\hat{\Omega},t)\, W_g(\bar{r},\hat{\Omega},t)\, d\bar{r}\, d\hat{\Omega}\, dt\ . \qquad \text{(F9.D.81)}$$

Integral Emergent Adjuncton Density Equation

The solution of either the point-value equation, Eq. (F9.D.76), or the event-value equation, Eq. (F9.D.80), could be accomplished by Monte Carlo procedures; however, the random walk would not be the same as that implied by Eq. (F9.D.40).[*] Consider the following altered form of Eq. (F9.D.76),

[*] The desire in MORSE is to use the same random-walk logic for both forward and adjoint calculations.

$$\chi_g^*(\bar{r},\bar{\Omega},t) = P_g^{\chi}(\bar{r},\bar{\Omega},t) + \int dR\, \Sigma_t^g(\bar{r})\, e^{-\beta_g^*(\bar{r},R,\Omega)} \left(\frac{\Sigma_t^g(\bar{r}')}{\Sigma_t^g(\bar{r})} \right)$$

$$\times \sum_{g'} \int d\bar{\Omega}' \frac{\Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')}{\sum_g \Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')} \left(\frac{\sum_g \Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')}{\Sigma_t^g(\bar{r}')} \right) \chi_{g'}^*(\bar{r}',\bar{\Omega}',t') \ . \qquad \text{(F9.D.82)}$$

The additional weight factor $[\Sigma_t^g(\bar{r}')/\Sigma_t^g(\bar{r})]$ arises since Eq. (F9.D.76) and its altered form, Eq. (F9.D.82), are actually **flux-like** equations, even though $\chi_g^*(\bar{r},\bar{\Omega},t)$ is adjoint to the emergent particle density $\chi_g(\bar{r},\bar{\Omega},t)$.

In a fashion analogous to the forward problem, the following new quantities are defined:

$$H_g(\bar{r},\bar{\Omega},t) \equiv \Sigma_t^g(\bar{r})\, \chi_g^*(\bar{r},\bar{\Omega},t) \qquad \text{(F9.D.83)}$$

and

$$H_g(\bar{r},\bar{\Omega},t) = T_g(\bar{r}\to\bar{r}',\bar{\Omega})\, G_g(\bar{r}',\bar{\Omega},t') \ . \qquad \text{(F9.D.84)}$$

Since $\chi_g^*(\bar{r},\bar{\Omega},t)$ is a flux-like variable, the new variable $H_g(\bar{r},\bar{\Omega},t)$ can be regarded as an event density and $G_g(\bar{r},\bar{\Omega},t)$ like an emergent particle density. The defining integral equation for $G_g(\bar{r},\bar{\Omega},t)$ should be the proper basis for an adjoint random walk.

The defining equation for the adjoint event density function $H_g(\bar{r},\bar{\Omega},t)$ is obtained by considering the following altered form of Eq. (F9.D.75):

$$\chi_g^*(\bar{r},\bar{\Omega},t) = \int dR\, \Sigma_t^g(\bar{r}')\, e^{-\beta_g^*(\bar{r},R,\Omega)} [P_g^{\psi}(\bar{r}',\bar{\Omega},t') + C_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')\, \chi_{g'}^*(\bar{r}',\bar{\Omega}',t')] \ . \qquad \text{(F9.D.85)}$$

Multiply Eq. (F9.D.85) by $\Sigma_t^g(\bar{r})$ and rearrange as follows:

$$\Sigma_t^g(\bar{r})\, \chi_g^*(\bar{r},\bar{\Omega},t) = \int dR\, \Sigma_t^g(\bar{r})\, e^{-\beta_g^*(\bar{r},R,\Omega)} [\Sigma_t^g(\bar{r}')\, P_g^{\psi}(\bar{r}',\bar{\Omega},t')$$

$$+ \check{C}_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')\, \Sigma_t^{g'}(\bar{r}')\, \chi_{g'}^*(\bar{r}',\bar{\Omega}',t')] , \qquad \text{(F9.D.86)}$$

where

$$\check{C}_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}') \equiv \frac{\Sigma_t^g(\bar{r}')}{\Sigma_t^{g'}(\bar{r}')} C_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}') \ . \qquad \text{(F9.D.87)}$$

Noting that

$$H_g(\bar{r},\bar{\Omega},t) = \Sigma_t^g(\bar{r})\,\chi_g^*(\bar{r},\bar{\Omega},t) ,$$

$$\int dR\, \Sigma_t^g(\bar{r})\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} = T_g(\bar{r}\rightarrow\bar{r}',\bar{\Omega}) ,$$

and

$$\Sigma_t^g(\bar{r})\, P_g^{\psi}(\bar{r},\bar{\Omega},t) = P_g^{\phi}(\bar{r},\bar{\Omega},t) ,$$

Eq. (F9.D.86) becomes

$$H_g(\bar{r},\bar{\Omega},t) = T_g(\bar{r}\rightarrow\bar{r}',\bar{\Omega})[P_g^{\phi}(\bar{r}',\bar{\Omega},t') + \check{C}_{g\rightarrow g'}(\bar{r}',\bar{\Omega}\rightarrow\bar{\Omega}')H_{g'}(\bar{r}',\bar{\Omega}',t')] . \qquad \text{(F9.D.88)}$$

A comparison of Eq. (F9.D.88) with Eq. (F9.D.84) reveals that

$$G_g(\bar{r},\bar{\Omega},t) = P_g^{\phi}(\bar{r},\bar{\Omega},t) + \check{C}_{g\rightarrow g'}(\bar{r},\bar{\Omega}\rightarrow\bar{\Omega}')\, H_{g'}(\bar{r},\bar{\Omega}',t) , \qquad \text{(F9.D.89)}$$

and the subsequent substitution of Eq. (F9.D.84) into Eq. (F9.D.89) yields the following defining equation for the adjoint emergent particle density:

$$G_g(\bar{r},\bar{\Omega},t) = P_g^{\phi}(\bar{r},\bar{\Omega},t) + \check{C}_{g\rightarrow g'}(\bar{r},\bar{\Omega}\rightarrow\bar{\Omega}')\, T_{g'}(\bar{r}\rightarrow\bar{r}',\bar{\Omega}')\, G_{g'}(\bar{r}',\bar{\Omega}',t') . \qquad \text{(F9.D.90)}$$

Equation (F9.D.90) is almost identical with Eq. (F9.D.40), which defines the forward emergent particle density $\chi_g(\bar{r},\bar{\Omega},t)$ and also serves as the formal basis for the forward random walk. At this point, let us interpret Eq. (F9.D.90) in terms of the transport of pseudo-particles called "adjunctons" in the (P′→P) direction of phase space. This presents two immediate problems:

1. The transport of the adjunctons from $\bar{r}' = \bar{r} + R\bar{\Omega}$ to $\bar{r}$ would be in a direction opposite to the direction vector $\bar{\Omega}$ — therefore, the direction vector for the adjuncton should be $\hat{\Omega} \equiv -\bar{\Omega}$, and $\bar{r}' = \bar{r} - R\hat{\Omega}$.
2. The collision kernel should be interpreted as describing the (E′→E) change in phase space experienced by the adjuncton during its random walk; therefore, let

$$C_{g'\rightarrow g}(\bar{r},\hat{\Omega}'\rightarrow\hat{\Omega}) \equiv \check{C}_{g\rightarrow g'}(\bar{r},\bar{\Omega}\rightarrow\bar{\Omega}') = \sum_{g'} \int d\bar{\Omega}' \frac{\Sigma_s^{g\rightarrow g'}(\bar{r},\bar{\Omega}\rightarrow\bar{\Omega}')}{\Sigma_t^{g'}(\bar{r})} . \qquad \text{(F9.D.91)}$$

Equation (F9.D.91) may be rewritten in terms of a normalized collision kernel and a weight factor:

$$C_{g'\rightarrow g}(\bar{r},\hat{\Omega}'\rightarrow\hat{\Omega}) = \sum_{g'} \int d\bar{\Omega}' \frac{\Sigma_s^{g\rightarrow g'}(\bar{r},\bar{\Omega}\rightarrow\bar{\Omega}')}{\sum_{g} \int d\bar{\Omega}\, \Sigma_s^{g\rightarrow g'}(\bar{r},\bar{\Omega}\rightarrow\bar{\Omega}')} \left(\frac{\sum_{g} \int d\bar{\Omega}\, \Sigma_s^{g\rightarrow g'}(\bar{r},\bar{\Omega}\rightarrow\bar{\Omega}')}{\Sigma_t^{g'}(\bar{r})} \right)^{*} . \qquad \text{(F9.D.92)}$$

The selection of new phase-space coordinates (group g, $\hat{\Omega} = -\bar{\Omega}$) is made from the normalized kernel, and the weight of the adjuncton is modified by the weight factor $[\ ]^*$ which is no longer a simple nonabsorption probability and may assume values in excess of unity. Therefore, there is no "analogue" scattering for adjunctons, and the adjuncton's weight may increase at some collisions.

Equation (F9.D.90) can be rewritten as

$$G_g(\bar{r},\hat{\Omega},t) = P_g^{\phi}(\bar{r},\hat{\Omega},t) + C_{g'\to g}(\bar{r},\hat{\Omega}'\to\hat{\Omega})\, T_{g'}(\bar{r}'\to\bar{r},\hat{\Omega}')\, G_{g'}(\bar{r}',\hat{\Omega}',t') , \qquad \text{(F9.D.93)}$$

which now corresponds to the transport of adjunctons and provides the desired basis for the adjoint random walk in the MORSE code. Note that the source of adjunctons is provided by $P_g^{\phi}(\bar{r},\hat{\Omega},t)$, which is related to $P_g^{\phi}(\bar{r},\bar{\Omega},t)$ as follows:

$$P_g^{\phi}(\bar{r},\hat{\Omega},t) = P_g^{\phi}(\bar{r},-\bar{\Omega},t) , \qquad \text{(F9.D.94)}$$

which must be taken into consideration if the response function $P_g^{\phi}(\bar{r},\bar{\Omega},t)$ has angular dependence; however, many physical situations permit an isotropic assumption for the $\bar{\Omega}$ dependence.

A Monte Carlo solution of Eq. (F9.D.93), the "integral emergent adjuncton density equation, will generate data from which the adjuncton flux $\chi_g^*(\bar{r},\hat{\Omega})$ and other quantities of interest can be determined. The general use of $\chi_g^*(\bar{r},\hat{\Omega})$ must take into account the reversal of direction between adjunctons and real particles (i.e., $\hat{\Omega} = -\bar{\Omega}$). For example, consider the various ways of calculating the answer of interest:

$$\begin{aligned}\lambda &= \sum_g \iiint P_g^{\phi}(\bar{r},\bar{\Omega},t)\, \phi_g(\bar{r},\bar{\Omega},t)\, d\bar{r}\, d\bar{\Omega}\, dt \\ &= \sum_g \iiint \frac{P_g^{\phi}(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})}\, T_g(\bar{r}'\to\bar{r},\bar{\Omega})\, \chi_g(\bar{r}',\bar{\Omega},t')\, d\bar{r}\, d\bar{\Omega}\, dt \end{aligned} \qquad \text{(F9.D.95)}$$

$$\lambda = \sum_g \iiint S_g(\bar{r},\bar{\Omega},t)\, \chi_g^*(\bar{r},\bar{\Omega},t)\, d\bar{r}\, d\bar{\Omega}\, dt = \iiint S_g(\bar{r},\bar{\Omega},t)\, \chi_g^*(\bar{r},-\hat{\Omega},t)\, d\bar{r}\, d\bar{\Omega}\, dt \qquad \text{(F9.D.96)}$$

$$\lambda = \sum_g \iiint S_c^g(\bar{r},\bar{\Omega},t)\, W_g(\bar{r},\bar{\Omega},t)\, d\bar{r}\, d\bar{\Omega}\, dt = \iiint S_c^g(\bar{r},\bar{\Omega},t)\, W_g(\bar{r},-\hat{\Omega},t)\, d\bar{r}\, d\bar{\Omega}\, dt \qquad \text{(F9.D.97)}$$

$$\lambda = \sum_g \iiint \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})}\, H_g(\bar{r},\bar{\Omega},t)\, d\bar{r}\, d\bar{\Omega}\, dt = \iiint \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})}\, H_g(\bar{r},-\hat{\Omega},t)\, d\bar{r}\, d\bar{\Omega}\, dt \qquad \text{(F9.D.98)}$$

$$\lambda = \sum_g \int\int\int \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} T_g(\bar{r}'\rightarrow\bar{r},\bar{\Omega})\, G_g(\bar{r}',\bar{\Omega},t')d\bar{r}d\bar{\Omega}dt$$

$$= \sum_g \int\int\int \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} T_g(\bar{r}'\rightarrow\bar{r},-\hat{\Omega})\, G_g(\bar{r}',-\hat{\Omega},t')d\bar{r}d\bar{\Omega}dt \ . \qquad \text{(F9.D.99)}$$

Further, if outward boundary crossings would be scored in the forward problem, the corresponding source adjunctons would be introduced in the inward direction. Likewise, adjunctons would be scored for entering a volume from which the source particles in the forward problem would be emitted. Note that many sources and response functions are isotropic, thus the problem of direction reversal need not be considered.

Multiplying Systems[*]

The group notation form of the general integral equations as provided in the previous section are modified in this section to address the specific problem of multiplying systems. In a fissioning system, it will be presumed that the source of neutrons for the *nth* generation comes from fissions that occur during the previous generation, the (n−1)*st* generation. In group notation and seven-dimensional phase space, the source term for the nth generation, $S_g^n(\bar{r},\bar{\Omega},t)$, is given by

$$S_g^n(\bar{r},\bar{\Omega},t) = \int_{\Delta E_g} S^n(\bar{r},E,\bar{\Omega},t)dE \, , \qquad \text{(F9.D.100)}$$

where

$S^n(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega}$ = source particles emitted for the nth generation per unit volume and time at the space point $\bar{r}$ and time t with energies in dE about E and directions that lie in $d\bar{\Omega}$ about $\bar{\Omega}$,

$$S^n(\bar{r},E,\bar{\Omega},t) = \frac{f(E)}{4\pi} \int\int_{4\pi} dE'd\bar{\Omega}' \ \nu\Sigma_f(\bar{r},E')\phi^{n-1}(\bar{r},E',\bar{\Omega}',t) \qquad \text{(F9.D.101)}$$

$f(E)dE$ = fraction of fission neutrons emitted having energies in dE about E,

$\phi^{n-1}(\bar{r},E,\bar{\Omega},t)$ = angular neutron flux for the (n-1)st generation,

$\nu\Sigma_f(\bar{r},E')$ = fission neutron yield × macroscopic fission cross section.

[*] The terms *generation* and *batch* will be used interchangeably in this section and will refer to the batches of neutrons processed in the MORSE Monte Carlo calculation.

Substitution of Eq. (F9.D.101) into Eq. (F9.D.100) and expressing the energy integration as a summation over energy groups yields

$$S_g^n(\bar{r},\bar{\Omega},t) = \frac{f_g}{4\pi}\sum_{g'=G}^{1}\int_{4\pi} d\bar{\Omega}'\, \nu\Sigma_f^{g'}(\bar{r})\,\phi_{g'}^{n-1}(\bar{r},\bar{\Omega}',t) \ , \qquad \text{(F9.D.102)}$$

where

$$f_g = \int_{\Delta E_g} f(E)dE \qquad \text{(F9.D.103)}$$

$$\nu\Sigma_f^g(\bar{r}) = \frac{\displaystyle\int_{\Delta E_g} \nu\Sigma_f(\bar{r},E)\,\phi(\bar{r},E,\bar{\Omega},t)dE}{\displaystyle\int_{\Delta E_g} \phi(\bar{r},E,\bar{\Omega},t)dE} \ . \qquad \text{(F9.D.104)}$$

Equation (F9.D.102) can also be expressed in terms of the emergent particle density:

$$S_g^n(\bar{r},\bar{\Omega},t) = \frac{f_g}{4\pi}\sum_{g'=G}^{1}\int_{4\pi} d\bar{\Omega}'\, \frac{\nu\Sigma_f^{g'}(\bar{r})}{\Sigma_t^{g'}(\bar{r})}\int_0^\infty dR\, \Sigma_t^{g'}(\bar{r})e^{-\beta_g(\bar{r},R,\bar{\Omega}')}\chi_{g'}^{n-1}(\bar{r}',\bar{\Omega}',t') , \qquad \text{(F9.D.105)}$$

where

$$\phi_{g'}(\bar{r},\bar{\Omega}',t) = \int_0^\infty dR\, e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')}\chi_{g'}(\bar{r}',\bar{\Omega}',t') \ , \qquad \text{(F9.D.106)}$$

so that for a given $S_g^n(\bar{r},\bar{\Omega},t)$, the emergent particle density distribution for the nth generation can be calculated using the following modified form of Eq. (F9.D.27):

$$\begin{aligned}
\chi_g^n(\bar{r},\bar{\Omega},t) &= S_g^n(\bar{r},\bar{\Omega},t) \\
&+ \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\, \frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})}\int_0^\infty dR\, \Sigma_t^{g'}(\bar{r})\, e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')}\chi_{g'}^n(\bar{r}',\bar{\Omega}',t') \\
&= S_g^n(\bar{r},\bar{\Omega},t) + C_{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})\,T_{g'}(\bar{r}'\to\bar{r},\bar{\Omega}')\,\chi_{g'}^n(\bar{r}',\bar{\Omega}',t') \ . \qquad \text{(F9.D.107)}
\end{aligned}$$

Equations (F9.D.105) and (F9.D.107) can be combined and written as an eigenvalue equation in seven-dimensional phase space

$$\chi_g(\bar{r},\bar{\Omega},t) = \frac{1}{k}\frac{f_g}{4\pi}\sum_{g'=G}^{1}\int_{4\pi} d\bar{\Omega}'\,\frac{\nu\Sigma_f^{g'}(\bar{r})}{\Sigma_t^{g'}(\bar{r})}\,dR\,\Sigma_t^{g'}(\bar{r})\,e^{-\beta_g(\bar{r},R,\Omega)}\chi_{g'}(\bar{r}',\bar{\Omega},t')$$

$$+\sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\,\frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})}\int_0^\infty dR\,\Sigma_t^{g'}(\bar{r})\,e^{-\beta_{g'}(\bar{r},R,\Omega')}\chi_{g'}(\bar{r}',\bar{\Omega}',t')\ . \qquad \text{(F9.D.108)}$$

The usual objective in a reactor calculation is to find the eigenfunctions $\chi_g(\bar{r},\bar{\Omega},t)$ and the eigenvalue k. In MORSE, this is accomplished iteratively, each batch being one iteration. The source for the first batch is unknown and must be assumed. From this source an estimate of the resulting emergent particle densities, χ_g^1, are calculated from Eq. (F9.D.107). The source for the next batch, S_g^2, is obtained from Eq. (F9.D.105) and then estimates of the χ_g^2 are obtained from Eq. (F9.D.107). After the source has converged (usually after a few batches), the χ_g^n are presumed to be a valid estimate of the eigenfunction χ_g in Eq. (F9.D.108) and an estimate of the multiplication factor can be obtained for each of the succeeding batches.

The multiplication factor corresponding to the nth generation (or batch) is defined as the ratio of the total production of fission neutrons during the nth generation to the total number of source neutrons introduced into the nth generation

$$k_n = \frac{\displaystyle\sum_{g'=G}^{1}\iiint d\bar{r}d\bar{\Omega}'dt\,\frac{\nu\Sigma_f^{g'}(\bar{r})}{\Sigma_t^{g'}(\bar{r})}\int_0^\infty dR\,\Sigma_t^{g'}(\bar{r})\,e^{-\beta_{g'}(\bar{r},R,\Omega')}\chi_{g'}^n(\bar{r}',\bar{\Omega}',t')}{\displaystyle\sum_{g'=G}^{1}\iiint S_{g'}^n\,d\bar{r}d\bar{\Omega}dt}\ , \qquad \text{(F9.D.109)}$$

which can also be expressed as the ratio of successive sources

$$k_n = \frac{\displaystyle\sum_{g'=G}^{1}\iiint S_{g'}^{n+1}(\bar{r},\bar{\Omega},t)d\bar{r}\,d\bar{\Omega}dt}{\displaystyle\sum_{g'=G}^{1}\iiint S_{g'}^{n}(\bar{r},\bar{\Omega},t)d\bar{r}\,d\bar{\Omega}dt}\ . \qquad \text{(F9.D.110)}$$

The multiplication factor is calculated at the end of each batch and the eigenvalue, k, is taken as the mean value of the k_n averaged over all the batches calculated after convergence of the eigenfunctions was achieved.

Equation (F9.D.107) is solved by MORSE in the same manner as it would be for nonfissioning systems. The fission event is treated as an absorption, and the neutron's weight is modified accordingly (i.e., fissions that occur do not introduce new neutrons into the present generation). The multiplication factor, k_n, is estimated by summing the contribution $\nu\Sigma_f^g(\bar{r})/\Sigma_t^g(\bar{r})$. W_b, the neutron's weight before collision, at every collision is an estimate of the collision density. At the end of the batch k_n is divided by N, the total starting weight of the batch.

The source for the next batch is not obtained directly from the individual contributions $(\nu\Sigma_f^g \cdot W_b)$. Rather, Russian roulette and splitting are used to discretize these contributions into ones of equal value. The splitting and Russian roulette parameters used are determined by the input parameter, FWLOW, the desired value of a single contribution. To keep the number of neutrons from multiplying or decreasing indefinitely, FWLO is modified from batch to batch such that the number of source neutrons for each batch remains nearly constant. The value of FWLOW for the (n + 1)*st* batch is calculated at the completion of the n*th* batch as follows:

$$\mathrm{FWLO}_{n+1} = \mathrm{FWLO}_n \cdot \bar{k}_n \cdot \frac{\text{(fisson neutrons produced during the nth batch)}}{\text{(source neutrons introduced into the first batch)}} \;, \qquad \text{(F9.D.111)}$$

where $\bar{k}_n$ is an accumulative estimate of k through n batches. The $\bar{k}_n$ modifying factor is required since the FWLO calculated after the *nth* batch affects the number of source neutrons in the (n + 2)*nd* batch.

The adjoint problem for the fissioning system is solved by MORSE in terms of the random walk of "adjunctons" as described by the integral emergent adjuncton density equation, Eq. (F9.D.93), that can be rewritten in batch notation as

$$G_g^n(\bar{r},\hat{\Omega},t) = [P_g^{\phi}(\bar{r},\hat{\Omega},t)]^n + C_{g'\rightarrow g}(\bar{r},\hat{\Omega}'\rightarrow\hat{\Omega})\, T_{g'}(\bar{r}'\rightarrow\bar{r},\hat{\Omega}')\, G_{g'}^n(\bar{r}',\hat{\Omega}',t') \;, \qquad \text{(F9.D.112)}$$

with the source of adjunctons being provided by the response function based on flux density, P_g^{ϕ}. The effect of interest for the nth generation, λ_g^n, is the production of fission neutrons due to fissions in group g that appear at the fission site in the next generation according to the group fission spectrum, f_g, and is given by

$$\begin{aligned}\lambda_g^{n-1} &= \iiint [P_g^{\phi}(\bar{r},\bar{\Omega},t)]^n\, \phi_g^{n-1}(\bar{r},\bar{\Omega},t)\,d\bar{r}\,d\bar{\Omega}\,dt \\ &= \iiint \left(\nu\Sigma_f^g(\bar{r}) \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}'\, \frac{f_{g'}}{4\pi}\, \chi_{g'}^{*n-1}(\bar{r},\bar{\Omega}',t) \right) \phi_g^{n-1}(\bar{r},\bar{\Omega},t)\,d\bar{r}\,d\bar{\Omega}\,dt \,, \end{aligned} \qquad \text{(F9.D.113)}$$

where

$\chi_g^{*n}(\bar{r},\bar{\Omega},t)$ = the value to the effect of interest in the n*th* generation of an emergent neutron with phase-space coordinates (group g, $\bar{r}$, $\bar{\Omega}$, t).

From Eq. (F9.D.113), the source of adjunctons for the n*th* generation is identified as

$$[P_g^{\phi}(\bar{r},\bar{\Omega},t)]^n = \nu\Sigma_f^g(\bar{r}) \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}'\, \frac{f_{g'}}{4\pi}\, \chi_{g'}^{*n-1}(\bar{r},\bar{\Omega}',t) \;. \qquad \text{(F9.D.114)}$$

Noting that according to Eqs. (F9.D.83) and (F9.D.84)

$$\chi_g^{*n}(\bar{r},\bar{\Omega},t) = \frac{1}{\Sigma_t^g(\bar{r})}\, T_g(\bar{r}'\rightarrow\bar{r},\bar{\Omega})\, G_g^n(\bar{r}',\bar{\Omega},t') \;, \qquad \text{(F9.D.115)}$$

Eq. (F9.D.114) can be rewritten as

$$[P_g^{\phi}(\bar{r},\bar{\Omega},t)]^n = \nu\Sigma_f^g(\bar{r}) \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}' \frac{f_{g'}}{4\pi\, \Sigma_t^{g'}(\bar{r})} T_{g'}(\bar{r}'\rightarrow\bar{r},\bar{\Omega}')\, G_{g'}^{n-1}(\bar{r}',\bar{\Omega}',t')$$

$$= \frac{\nu\Sigma_f^g(\bar{r})}{4\pi} \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}'\, f_{g'} \int_0^{\infty} dR\, e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')}\, G_{g'}^{n-1}(\bar{r}',\bar{\Omega}',t') \ . \qquad \text{(F9.D.116)}$$

Noting that the fission process is independent of the incident neutron's direction and that the fission neutrons are emitted isotropically

$$[P_g^{\phi}(\bar{r},\bar{\Omega},t)]^n \equiv [P_g^{\phi}(\bar{r},\hat{\Omega},t)]^n \ , \qquad \text{(F9.D.117)}$$

and Eq. (F9.D.116) can be used in conjunction with Eq. (F9.D.112) (i.e., $\bar{\Omega}$ replaced by $\hat{\Omega}$).

Equation (F9.D.116) can be rewritten as

$$[P_g^{\phi}(\bar{r},\hat{\Omega},t)]^n = \frac{1}{4\pi} \frac{\nu\Sigma_f^g(\bar{r})}{\nu\Sigma_f(\bar{r})} \sum_{g'=G}^{1} \int_{4\pi} d\hat{\Omega}'\, [f_{g'}\, \nu\Sigma_f(\bar{r})] \int_0^{\infty} dR\, e^{-\beta_{g'}(\bar{r},R,\hat{\Omega}')}$$

$$\times\, G_{g'}^{n-1}(\bar{r}',\hat{\Omega}',t') \ , \qquad \text{(F9.D.118)}$$

where

$$\nu\Sigma_f(\bar{r}) = \sum_{g=G}^{1} \nu\Sigma_f^g(\bar{r}) \ , \qquad \text{(F9.D.119)}$$

$\dfrac{\nu\Sigma_f^g(\bar{r})}{\nu\Sigma_f(\bar{r})}$ = energy distribution of adjunctons emerging from an adjoint fission,

$[f_{g'}\, \nu\Sigma_f(\bar{r})]$ = the g*th* group cross section for adjoint fission.

Note that the integral emergent particle density equation, Eq. (F9.D.107), is identical in form with the integral emergent adjuncton density equation, Eq. (F9.D.112), so that essentially* the same random walk procedures can apply to the solution of the forward and adjoint fissioning systems. The adjoint source, Eq. (F9.D.118), differs from the forward source, Eq. (F9.D.105), only in that the fission cross section and the group fission spectrum have changed their roles. The adjoint-fission group cross section is $[f_g \cdot \nu\Sigma_f(\bar{r})]$ and the energy distribution of the adjunctons emerging from an adjoint fission is $\nu\Sigma_f^g(\bar{r})/\nu\Sigma_f(\bar{r})$.

*The same differences will exist between the forward and adjoint collision kernels here as was the case for nonfissioning systems.

The adjoint solution is started by assuming some arbitrary initial source, $[P_g^{\phi}(\bar{r},\hat{\Omega},t)]^1$, and calculating the $G_g^1(\bar{r},\hat{\Omega},t)$ using Eq. (F9.D.112). A new source term, $[P_g^{\phi}(\bar{r},\hat{\Omega})]^2$, is then calculated from Eq. (F9.D.118) and the next estimate, $G_g^2(\bar{r},\hat{\Omega})$, is calculated using Eq. (F9.D.112). This procedure continues until, as in the forward case, the source has converged and the G_g^n's are presumed to be an estimate of the eigenfunctions G_g. Then for each succeeding batch, the following estimate is made for the eigenvalue k:

$$k_n = \frac{\sum\limits_{g'=G}^{1} \iiint d\bar{r}d\hat{\Omega}'dt\,[f_{g'}\cdot \nu\Sigma_f^{g'}(\bar{r})] \int\limits_0 dR\, e^{-\beta_{g'}(\bar{r},R,\hat{\Omega}')} G_{g'}^n(\bar{r}',\hat{\Omega}',t')}{\sum\limits_{g'=G}^{1} \iiint [P_{g'}^{\phi}(\bar{r},\hat{\Omega},t)]^n d\bar{r}d\hat{\Omega}dt} , \qquad \text{(F9.D.120)}$$

and the eigenvalue, k, is taken as the mean value of the k_n averaged over all the batches calculated after convergence of the eigenfunctions was achieved — exactly the same procedure used in the forward calculation.

F9.D.2 GENERALIZED GAUSSIAN QUADRATURE

General Statement of the Problem and Its Solution

Given $\omega(x)$, $a \le x \le b$, such that $\omega(x) \ge 0$ (Restriction I).

Problem: find $\{x_i, \omega_i\}$ for i = 1,n so that

$$\int_a^b f(x)\,\omega(x)\,dx = \sum_{i=1}^{n} f(x_i)\cdot\omega_i \qquad \text{(Restriction II)}$$

holds for all f(x) where f(x) is a polynomial of degree 2n − 1 or less.

Solution: Determine a set of polynomials $Q_i(x)$ (i = 1,n) orthogonal with respect to $\omega(x)$. That is,

$$\int_a^b Q_i(x)\,Q_j(x)\,\omega(x)\,dx = \delta_{ij}\,N_i ,$$

where δ_{ij} is the Kronecker delta and N_i is a normalization constant. Then $\{x_i\}_{i=1}^{n}$ are given by the roots of $Q_n(x)$, $Q_n(x_i) = 0$, and

$$\omega_i = \left(\sum_{i=1}^{n-1} Q_i^2(x_i)/N_i \right)^{-1} .$$

Note: Since the functions $1,x,x^2,....x^{2n-1}$ are independent and form a basis for the space of all polynomials of degree 2n − 1 or less, it is equivalent to Restriction II to require that

$$M_\nu = \int_a^b x^\nu \omega(x)\,dx = \sum_{i=1}^n x_i^\nu \cdot \omega_i \text{ for } \nu = 0,2n-1 .$$

In other words, the problem is that of finding a discrete distribution,

$$\omega^*(x) = \sum_{i=1}^n \omega_i\, \delta(x - x_i) ,$$

having its first 2n moments, $\{M_\nu\}_{\nu=0}^{2n-1}$ identical to those of the original distribution $\omega(x)$.

It is then possible to relax the nonnegativity restriction, $\omega(x) \geq 0$, and in fact to state that $\omega(x)$ need not be completely specified but only its first 2n moments be given. Restriction I then becomes two restrictions on the moments:

I_a: $|C_i| \geq 0$ $i = 1, n-1$

where $|C_i|$ is the Gram determinant

$$|C_1| = \begin{vmatrix} M_0 M_1 \\ M_1 M_2 \end{vmatrix}, \ |C_2| = \begin{vmatrix} M_0 M_1 M_2 \\ M_1 M_2 M_3 \\ M_2 M_3 M_4 \end{vmatrix}, \ ..., \ |C_{n-1}| = \begin{vmatrix} M_0 M_1 M_2 \ldots M_{n-1} \\ M_1 M_2 \quad \ldots M_n \\ \cdot \\ \cdot \\ \cdot \\ M_{n-1} M_n \text{ --- } M_{2n-2} \end{vmatrix}$$

and

I_b: the roots of $Q_n(x)$ lie inside the interval [a,b], i.e., $a \leq x_i \leq b$ whenever $Q_n(x_i) = 0$.

Equivalence of Moments and Legendre Coefficients

We shall use the following form for the Legendre expansion of an angular distribution:

$$f(\mu) = \sum_{\ell=0}^{\infty} \frac{2\ell + 1}{2} f_\ell P_\ell(\mu) . \qquad \text{(F9.D.121)}$$

From this it follows that

$$f_\ell = \int_{-1}^{1} f(\mu) P_\ell(\mu)\, d\mu \text{ and } f_0 \equiv 1 . \qquad \text{(F9.D.122)}$$

The moments of the distribution are defined by

$$M_n = \int_{-1}^{1} \mu^n f(\mu)\, d\mu \ . \qquad \text{(F9.D.123)}$$

If the Legendre polynomials are written

$$P_\ell(\mu) = \sum_{n=0}^{\ell} p_{\ell n}\, \mu^n \qquad \text{(F9.D.124)}$$

[the $p_{\ell n}$'s may be derived easily from the recurrence relation for $P_\ell(\mu)$]. Then it follows simply from Eq. (F9.D.122) that

$$f_\ell = \sum_{n=0}^{\ell} p_{\ell n} \int_{-1}^{1} f(\mu)\, \mu^n\, d\mu = \sum_{n=0}^{\ell} p_{\ell n} M_n \ . \qquad \text{(F9.D.125)}$$

Likewise,

$$M_n = \int_{-1}^{1} \mu^n f(\mu)\, d\mu = \sum_{\ell=0}^{\infty} \frac{2\ell + 1}{2} f_\ell \int_{-1}^{1} \mu^n P_\ell(\mu)\, d\mu \ .$$

From the orthogonality property we know that $P_\ell(\mu)$ is orthogonal to any polynomial of degree less than ℓ. Hence,

$$\int_{-1}^{1} \mu^n P_\ell(\mu)\, d\mu = 0 \quad \text{for } \ell > n \ .$$

Then,

$$M_n = \sum_{\ell=0}^{n} \frac{2\ell + 1}{2} f_\ell\, p_{n\ell}^{-1} , \qquad \text{(F9.D.126)}$$

where

$$p_{n\ell}^{-1} = \int_{-1}^{1} \mu^n P_\ell(\mu) d\mu$$

are the coefficients for a Legendre expansion of μ^n; that is,

$$\mu^n = \sum_{\ell=0}^{n} \frac{2\ell + 1}{2} p_{n\ell}^{-1} P_\ell(\mu) \ .$$

[The Legendre polynomial recurrence relation may also be used to derive recurrence relations for the $p_{n\ell}^{-1}$.]

Equations (F9.D.125) and (F9.D.126) show that the first n moments of an angular distribution may be derived from the first n Legendre coefficients and vice versa.

Generation of Polynomials Orthogonal With Respect to ω(x)

Let us now presume that we are given the first 2n moments, $M_0, M_1, \ldots, M_{2n-1}$, of an arbitrary function ω(x) and are given no additional information about ω(x). We shall attempt to derive a set of polynomials which are orthogonal with respect to ω(x). If we define the notation

$$E[I(x)] = \int_a^b I(x)\,\omega(x)\,dx \ ,$$

then what we wish is to determine $Q_0, Q_1, \ldots, Q_n$ such that

$$Q_i(x) = \sum_{k=0}^{i} a_{ik}\,x^k \ , \qquad \text{(F9.D.127)}$$

with the normalization condition $a_{ii} = 1$, and that

$$E[Q_i(x)\,Q_j(x)] = \delta_{ij}\,N_i \ . \qquad \text{(F9.D.128)}$$

Note that

$$N_i = E[Q_i^2(x)] = \int_a^b Q_i^2(x)\,\omega(x)\,dx \ .$$

Since $\omega(x) \geq 0$, then it follows that*

$$N_i > 0 \,. \tag{F9.D.129}$$

From the properties of orthogonal polynomials we know that an arbitrary polynomial of order i, $S_i(x)$, may be expanded in terms of the Q polynomials,

$$S_i(x) = \sum_{k=0}^{i} s_{ik} Q_k(x) \,.$$

It follows that

$$E[S_i(x)\, Q_j(x)] = 0 \quad \text{for } i < j \,.$$

Let us presume that we have obtained the first i polynomials and are attempting to derive $Q_{i+1}(x)$. Due to our normalization condition ($a_{ii} = 1$), we have

$$Q_{i+1}(x) = x^{i+1} + R_i(x) \,, \tag{F9.D.130}$$

where

$$R_i(x) = \sum_{k=0}^{i} a_{i+1,k} x^{k} \,.$$

$$\begin{aligned} Q_{i+1}(x) &= x \cdot x^{i} + R_i(x) \\ &= x \cdot [Q_i(x) - R_{i-1}(x)] + R_i(x) \\ &= x\, Q_i(x) + [R_i(x) - x\, R_{i-1}(x)] \,. \end{aligned}$$

The term $R_i(x) - x\, R_{i-1}(x)$ is a polynomial of order i and may be expanded in terms of the Q's. Thus,

$$Q_{i+1}(x) = x\, Q_i(x) + \sum_{k=0}^{i} d_{ik}\, Q_k(x) \,. \tag{F9.D.131}$$

For $j \leq i-2$ we can use the orthogonality relation

*Since we wish to relax the nonnegativity restriction slightly but not completely, we will retain Eq. (F9.D.129) as a reasonable requirement for a "well-behaved" $\omega(x)$. This requirement is essential to allow full use of the properties of orthogonal polynomials. It is also essential to the eventual use of this development as a Monte Carlo selection technique since it is needed to ensure that the "probabilities," ω_i, be positive.

$$
\begin{aligned}
E[Q_{i+1}(x)\,Q_j(x)] \;=\; 0 \;=\; E[x\,Q_i(x)\,Q_j(x)] &+ \sum_{k=0}^{i} d_{ik}\,E[Q_k(x)\,Q_j(x)] \\
&= E[Q_i(x)\,(x\,Q_j(x))] \;+\; d_{ij}\,N_j \\
&= d_{ij}\,N_j \ ,
\end{aligned}
$$

since $x\ Q_j(x)$ is a polynomial of order $\leq i - 1$ and is orthogonal to $Q_i(x)$. Since $N_j > 0$, we must have $d_{ij} = 0$.

If we write

$$\mu_{i+1} = -\,d_{i,1}$$

and

$$\sigma_i^2 = -d_{i,i-1} \ ,$$

then Eq. (F9.D.131) reduces to

$$Q_{i+1}(x) \;=\; (x \;-\; \mu_{i+1})\,Q_i(x) \;-\; \sigma_i^2 Q_{i-1}(x) \ . \qquad \text{(F9.D.132)}$$

This equation is the basic recurrence relation for our polynomials. We have

$$
\begin{aligned}
&E[Q_{i+1}(x)\ Q_{i-1}(x)] = 0 \\
&\qquad = E[x\ Q_i(x)\ Q_{i-1}(x)] - \mu_{i+1}\,E[Q_i(x)\ Q_{i-1}(x)] - \sigma_i^2\ E[\,Q_{i-1}^2(x)] \\
&\qquad = E[Q_i(x)\ (x\ Q_{i-1}(x))] - \sigma_i^2\ N_{i-1} \\
&\qquad = E[Q_i(x)\ \{Q_i(x) - \sum_{k=0}^{i-1}\ d_{i-1,k}\ Q_k(x)\}] - \sigma_i^2\ N_{i-1} \\
&\qquad = E[\,Q_i^2(x)] - \sigma_i^2\ N_{i-1} \\
&\qquad = N_i - \sigma_i^2\ N_{i-1} \ .
\end{aligned}
$$

This equation is easily solved for

$$\sigma_i^2 \;=\; N_i/N_{i-1} \qquad \text{(F9.D.133)}$$

If we return to Eq. (F9.D.130), it is easy to see that

$$N_i = E[Q_i(x)\, Q_i(x)] = E[Q_i(x)\, x^{i}] + E[Q_i(x)\, R_{i-1}(x)]$$

$$= E[Q_i(x)\, x^{i}] = \sum_{k=0}^{i} a_{ik} \int_a^b x^{k} x^{i} \omega(x)\, dx = \sum_{k=0}^{i} a_{ik} M_{k+i} . \qquad \text{(F9.D.134)}$$

Likewise, we will define

$$L_{i+1} = E[Q_i(x)\, x^{i+1}]$$

$$= \sum_{k=0}^{i} a_{i,k} M_{k+i+1} . \qquad \text{(F9.D.135)}$$

Then the final orthogonality relation used in defining $Q_{i+1}(x)$ gives us

$$E[Q_{i+1}(x)\, Q_i(x)] = 0$$

$$= E[Q_{i+1}(x)\, x^{i}] + E[Q_{i+1}(x)\, R_{i-1}(x)]$$

$$= E[x\, Q_i(x)\, x^{i}] - \mu_{i+1} E[Q_i(x)\, x^{i}] - \sigma_i^2\, E[Q_{i-1}(x)\, x^{i}]$$

$$= L_{i+1} - \mu_{i+1} N_i - \sigma_i^2\, L_i$$

or

$$\mu_{i+1} = \frac{L_{i+1}}{N_i} - \sigma_i^2 \frac{L_i}{N_i}$$

$$= \frac{L_{i+1}}{N_i} - \frac{L_i}{N_{i-1}} . \qquad \text{(F9.D.136)}$$

The coefficients a_{ik} may be obtained from the recurrence relation, Eq. (F9.D.132), by taking the coefficient of x^k on both sides of the equation. This gives

$$a_{i+1,k} = a_{i,k-1} - \mu_{i+1} a_{i,k} - \sigma_i^2 a_{i-1,k} . \qquad \text{(F9.D.137)}$$

To recapitulate, one uses moments through M_{2i} and the values of a_{ik} from $Q_i(x)$ to calculate N_i [Eq. (F9.D.134)]. N_i, along with the previously determined N_{i-1}, allows one to calculate σ_i^2 [Eq. (F9.D.133)]. The moments through M_{2i+1} and $Q_i(x)$ determine L_{i+1} [Eq. (F9.D.135)]. This value in turn allows the calculation of μ_{i+1} [Eq. (F9.D.136)]. With σ_i^2 and μ_{i+1} the recurrence relation [Eq. (F9.D.132)] determines $Q_{i+1}(x)$. In sum, the moments M_0, M_1,..., M_{2n-1} of $\omega(x)$ allow the determination of the orthogonal polynomials $Q_0(x)$, $Q_1(x)$, ..., $Q_n(x)$. This determination is subject only to the restriction $N_i > 0$, $i = 0,n$. Although it is far from obvious, this restriction may be written in simple closed form as

$$\left| \begin{array}{l} M_0M_1M_2 \ldots M_i \\ M_1M_2M_3 \ldots M_{i+1} \\ . \\ . \\ . \\ . \\ M_iM_{i+1} \ldots M_{2i} \end{array} \right| > 0 \; .$$

Properties of the Roots of the Orthogonal Polynomials

The roots of the orthogonal polynomials have two useful properties which we shall prove:

Lemma I: $Q_n(x)$ has n distinct, real roots which "interleave" with the roots of $Q_{n-1}(x)$; that is, between any two adjacent roots of $Q_{n-1}(x)$ there is one and only one root of $Q_n(x)$, and furthermore there is one root of $Q_n(x)$ greater than the largest root of $Q_{n-1}(x)$ and one smaller than the least root of $Q_{n-1}(x)$. Likewise, there is one and only one root of $Q_{n-1}(x)$ between any two adjacent roots of $Q_n(x)$.

Proof: We assume the Lemma to be true for Q_{n-1} and Q_{n-2}. Let $x_1 > x_2 > \ldots > x_{n-1}$ be the roots of Q_{n-1}. Then it follows that the sequence $Q_{n-2}(x_1)$, $Q_{n-2}(x_2)$, ..., $Q_{n-2}(x_{n-1})$ alternates in sign. Since

$$Q_n(x_i) = (x_i - \mu_n) Q_{n-1}(x_i) - \sigma^2_{n-1} Q_{n-2}(x_i)$$

$$= - \sigma^2_{n-1} Q_{n-2}(x_i) \; .$$

The sequence $Q_n(x_i)$, $Q_n(x_2)$, ..., $Q_n(x_{n-1})$ also alternates in sign. This establishes that there is at least one root of Q_n between any two roots of Q_{n-1}. Because the Q_i's are normalized to $a_{ii} = 1$, they are all positive at $+\infty$ and alternate in sign at $-\infty$. Q_{n-2} has no root between x_1 and $+\infty$; hence, $Q_{n-2}(x_1) > 0$. But $\sigma^2_{n-1} > 0$ (because $N_{n-1} > 0$ and $N_{n-2} > 0$); therefore, $Q_n(x_1) < 0$ and Q_n must have at least one root greater than x_1. Similar reasoning leads to the conclusion that $Q_{n-2}(x_{n-1})$, $Q_{n-2}(x \rightarrow -\infty)$, and $Q_n(x \rightarrow -\infty)$ have the same sign while $Q_n(x_{n-1})$ is of the opposite sign. Thus, Q_n must have at least one root between x_{n-1} and $-\infty$. Since this gives us n intervals where Q_n must have "at least one" root, it is clear that Q_n has n distinct roots which interleave with the roots of Q_{n-1}.

The proof by induction may be completed by using similar arguments to show that one of the two roots of $Q_2(x)$ lies above the single root of $Q_1(x)$ and one below it.

Lemma II: The n roots of $Q_n(x)$ lie in the interval (a,b).

Proof: Assume that $Q_n(x)$ has only s changes of sign in the interval (a,b) at the points x_1, x_2, ..., x_s. Let

$$\theta(x) = (x - x_1)(x - x_2)(x - x_3) \ldots . (x - x_s) \; ,$$

then $\theta(x)\, Q_n(x)$ does not change sign in the interval (a,b). It follows that*

$$E[\theta(x)\, Q_n(x)] = \int_a^b \theta(x)\, Q_n(x)\, \omega(x)\, dx \neq 0 \ .$$

However, $\theta(x)$ is a polynomial of order $s \leq n$. Since $Q_n(x)$ is orthogonal to all polynomials of order less than n, we must have s = n, thus proving the assertion.

The Meaning of the Two Restrictions Which Replace the Nonnegativity Requirement, $\omega(x) \geq 0$

In the foregoing development, knowledge of the entire function $\omega(x)$ is never required. Instead, all that is needed are the moments, $M_0, M_1, ..., M_{2n-1}$, of $\omega(x)$. The generalized quadrature thus developed is thereby valid for the whole class of functions having those moments. Since the moments are equivalent to the Legendre coefficients, $f_0, f_1, ..., f_{2n-1}$, this class is comprised of all functions having the same truncated Legendre expansion; that is,

$$\omega(x) \approx \omega^*(x) = \sum_{\ell=0}^{2n-1} \frac{2\ell+1}{2} f_\ell P_\ell(x) \ .$$

In particular, the discrete distribution derived by this technique is itself one function from this class.

It is not required that all functions of this class be nonnegative; in fact, there are infinitely many which are not. It is not even required that the truncated Legendre expansion $\omega^*(x)$ be nonnegative. However, it is essential that at least one function in this class be nonnegative. The restrictions

(1) $N_i > 0$, i = 1, ..., n and
(2) $Q_n(x)$ has n roots in the interval (−1, +1)

express exactly this requirement. Then it follows that $\omega^*(x)$ is the truncated expansion of some unspecified nonnegative function. The failure of either of those two conditions expresses the fact that the given moments (or Legendre coefficients) are not those of any everywhere positive function.

Generation of the Generalized Gaussian Quadrature

We are given $\omega(x)$, $a \leq x \leq b$ [or, rather, we are given the moments of $\omega(x)$], and we are attempting to find a set of points, x_i, and associated weights, ω_i, so that for any arbitrary polynomial, f(x), of order 2n − 1 or less,

$$E[f(x)] = \int_a^b f(x)\, \omega(x)\, dx = \sum_{i=1}^{n} f(x_i) \cdot \omega_i \ .$$

*This step relies on the requirement that $\omega(x)$ be nonnegative. We wish to relax this restriction somewhat, but not completely. Since Lemma II expresses a property that will be essential to the use of this development as a Monte Carlo selection technique, we will use this property as one of the requirements for a "well-behaved" $\omega(x)$ with which we shall replace the nonnegativity restriction.

By simple division of polynomials,

$$f(x) = q_{n-1}(x)\, Q_n(x) + r_{n-1}(x) \; ,$$

where $q_{n-1}(x)$ and $r_{n-1}(x)$ are polynomials of order n − 1 or less.

$$\begin{aligned} E[f(x)] &= E[q_{n-1}(x)\, Q_n(x)] \; + \; E[r_{n-1}(x)] \\ &= E[r_{n-1}(x)] \text{ from the orthogonality property of } Q_n \; . \end{aligned} \tag{F9.D.138}$$

However, we want

$$\begin{aligned} E[f(x)] &= \sum_{i=1}^{n} f(x_i) \cdot \omega_i = \sum_{i=1}^{n} q_{n-1}(x_i)\, Q_n(x_i) \cdot \omega_i \; + \; \sum_{i=1}^{n} r_{n-1}(x_i) \cdot \omega_i \\ &= \sum_{i=1}^{n} q_{n-1}(x_i)\, Q_n(x_i) \cdot \omega_i \; + \; E[r_{n-1}(x)] \; . \end{aligned} \tag{F9.D.139}$$

By subtracting Eq. (F9.D.138) from Eq. (F9.D.139), we find that we must require, for all polynomials, $q_{n-1}(x)$, that

$$\sum_{i=1}^{n} q_{n-1}(x_i)\, Q_n(x_i) \cdot \omega_i = 0 \; . \tag{F9.D.140}$$

This condition can only be met if

$$Q_n(x_i) = 0 \; , \tag{F9.D.141}$$

that is, the desired points, x_i, are the roots of $Q_n(x)$.

Now we still must pick the weights, ω_i, so that

$$E[r_{n-1}(x)] = \sum_{i=1}^{n} r_{n-1}(x_i) \cdot \omega_i \; ,$$

where $r_{n-1}(x)$ is an arbitrary polynomial of order n − 1 or less. Since r_{n-1} may be expanded as a linear sum of the orthogonal polynomials, Q_0, Q_1, ..., Q_{n-1}, it is sufficient to require

$$E[Q_k(x)] = \sum_{i=1}^{n} Q_k(x_i) \cdot \omega_i \quad \text{for } k = 0,1,...,n-1 \; . \tag{F9.D.142}$$

However,

$$E[Q_k(x)] = E[Q_k(x)\, Q_o(x)] = N_o\, \delta_{ko} \; .$$

Thus we must have

$$\sum_{i=1}^{n} Q_k(x_i) \cdot \omega_i = N_o\,\delta_{ko} \quad \text{for } k = 0,1,\ldots,n-1 \,. \tag{F9.D.143}$$

Multiplying Eq. (F9.D.143) by $[Q_k(x_j)/N_j]$ and summing over k, we find

$$\begin{aligned} \sum_{k=0}^{n-1} \frac{Q_k(x_j)}{N_j} \sum_{i=1}^{n} Q_k(x_i) \cdot \omega_i &= \sum_{i=1}^{n} \omega_i m \left\{ \sum_{k=0}^{n-1} \frac{Q_k(x_j)\, Q_k(x_i)}{N_k} \right\} \\ &= \sum_{k=0}^{n-1} \frac{Q_k(x_j)}{N_j} N_o\,\delta_{ko} = \frac{Q_o(x_j)}{N_o} N_o = 1 \,. \end{aligned} \tag{F9.D.144}$$

Introducing the function

$$D_{n-1}(x,y) = \sum_{k=0}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} \,,$$

we can write Eq. (F9.D.144) as

$$\sum_{i=1}^{n} \omega_i\, D_{n-1}(x_j, x_i) = 1 \,. \tag{F9.D.145}$$

To proceed further, we must establish the Christoffel-Darboux identity,

$$\begin{aligned} &\frac{Q_n(x)\, Q_{n-1}(y) - Q_{n-1}(x)\, Q_n(y)}{N_{n-1}(x - y)} \\ &= \frac{[(x - \mu_n)Q_{n-1}(x) - \sigma^2_{n-1}\, Q_{n-2}(x)]Q_{n-1}(y) - Q_{n-1}(x)[(y - \mu_n)Q_{n-1}(y) - \sigma^2_{n-1}Q_{n-2}(y)]}{N_{n-1}(x - y)} \\ &= \frac{(x - y)\, Q_{n-1}(x)\, Q_{n-1}(y) + \sigma^2_{n-1}\,[Q_{n-1}(x)\, Q_{n-2}(y) - Q_{n-2}(x)\, Q_{n-1}(y)]}{N_{n-1}(x - y)} \\ &= \sum_{k=1}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} + \frac{Q_1(x)\, Q_0(y) - Q_0(x)\, Q_1(y)}{N_0(x - y)} \\ &= \sum_{k=1}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} + \frac{(x - \mu_1) - (y - \mu_1)}{N_0(x - y)} \end{aligned} \tag{F9.D.146}$$

$$= \sum_{k=0}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} = D_{n-1}(x,y) \ .$$

Therefore,

$$D_{n-1}(x_j,x_i) = \frac{Q_n(x_j)\, Q_{n-1}(x_i) - Q_{n-1}(x_j)\, Q_n(x_i)}{N_{n-1}(x_j - x_i)} \ . \qquad \text{(F9.D.147)}$$

For $i \neq j$ and $Q_n(x_j) = Q_n(x_i) = 0$,

$$D_{n-1}(x_j,x_i) = 0 \ .$$

Therefore, returning to Eq. (F9.D.145),

$$\sum_{i=1}^{n} \omega_i\, D_{n-1}(x_j,x_i) = \omega_j\, D_{n-1}(x_j,x_j) = 1$$

or

$$\omega_j = [D_{n-1}(x_j,x_j)]^{-1} = \left(\sum_{k=0}^{n-1} \frac{Q_k^2(x_j)}{N_k} \right)^{-1} \ . \qquad \text{(F9.D.148)}$$

Limits of μ_i and σ_i^2

In the calculations leading to the generalized Gaussian quadrature, we obtained two restrictions that had to be satisfied in order to have a positive distribution located on the interval (−1,+1). These restrictions were

1. $N_i > 0$.
2. All the roots of $Q_i(x)$ lie in the interval (−1,+1).

Let us determine first what limitations these two restrictions place on the quantities μ_i, σ_i^2. Consider first the effect of adding an infinitesimal amount $\Delta\mu$ to μ_i. We have

$$Q_i(x) = (x - \mu_i)\, Q_{i-1}(x) - \sigma_{i-1}^2\, Q_{i-2}(x)$$

and

$$Q_i^*(x) = (x - \mu_i - \Delta\mu)\, Q_{i-1}(x) - \sigma_{i-1}^2\, Q_{i-2}(x) = Q_i(x) - \Delta\mu\, Q_{i-1}(x) \ .$$

If Q_i has a root at x_0, then Q_i^* will have a root at $x_0 + \Delta x_0$

$$Q_i^*(x_0 + \Delta x_0) = 0 = Q_i(x_0 + \Delta x_0) - \Delta\mu \, Q_{i-1}(x_0 + \Delta x_0) \; .$$

If we expand the right-hand side and keep only first-order terms,

$$0 = Q_i(x_0) + \Delta x_0 \, Q'_i(x_0) - \Delta\mu \, Q_{i-1}(x_0) = \Delta x_0 \, Q'_i(x_0) - \Delta\mu \, Q_{i-1}(x_0)$$

or

$$\Delta x_0 = \frac{Q_{i-1}(x_0)}{Q'_i(x_0)} \Delta\mu \; . \qquad \text{(F9.D.149)}$$

Since $Q_i(x)$ is positive as x approaches $+\infty$, then $Q'_i(x_0) > 0$ at x_0 equal to the largest root of Q_i. At successively smaller roots of Q_i, the sign of $Q'_i(x)$ alternates from positive to negative. $Q_{i-1}(x)$ is similarly positive at $+\infty$. Also, it has no roots greater than the largest root of Q_i. Therefore, $Q_{i-1}(x) > 0$ at the largest root of Q_i. Because the roots of Q_{i-1} "interleave" with the roots of Q_i, the sign of $Q_{i-1}(x)$ must alternate at successive roots of $Q_i(x)$. Therefore, at all roots of $Q_i(x)$ we must have

$$\frac{Q_{i-1}(x)}{Q'_i(x)} > 0 \qquad \text{(F9.D.150)}$$

or, going back to Eq. (F9.D.149),

$$\frac{dx_0}{d\mu_i} > 0 \; .$$

Therefore, as μ_i is increased, the roots of $Q_i(x)$ shift to the right, and, as μ_i is decreased, the roots shift downward. If μ_i is steadily increased, the largest root of Q_i will eventually equal 1. This point is determined by

$$Q_i(1) = 0 = (1 - \mu_i) Q_{i-1}(1) - \sigma_{i-1}^2 Q_{i-2}(1)$$

or

$$\mu_i = 1 - \sigma_{i-1}^2 \frac{Q_{i-2}(1)}{Q_{i-1}(1)} \; .$$

This value is clearly the maximum value of μ_i, which will generate positivity in the interval (−1,+1). Likewise, there is a minimum value at which the lowest root of Q_i occurs at $x = -1$.

$$Q_i(-1) = 0 = (-1 - \mu_i) Q_{i-1}(-1) - \sigma_{i-1}^2 Q_{i-2}(-1)$$

or

$$\mu_i^{\min} = -1 - \sigma_{i-1}^2 \frac{Q_{i-2}(-1)}{Q_{i-1}(-1)} .$$

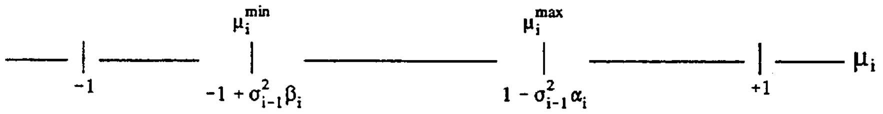
Note that

$$\alpha_i = \frac{Q_{i-2}(1)}{Q_{i-1}(1)} > 0 ,$$

due to the positivity of the functions as they approach +∞ and that

$$\beta_i = - \frac{Q_{i-2}(-1)}{Q_{i-1}(-1)} > 0 ,$$

due to their alternation in sign at −∞. Since $\sigma_{i-1}^2 > 0$, we have the following picture on a μ_i axis,



Now that we have upper and lower limits for μ_i, what can we say about σ_i^2? Since $\sigma_i^2 = N_i/N_{i-1}$, restriction I implies that $\sigma_i^2 > 0$. We can obtain an upper limit to σ_i^2 by setting $\mu_{i+1}^{\min} = \mu_{i+1}^{\max}$. For larger values of σ_i^2, $\mu_{i+1}^{\min} > \mu_{i+1}^{\max}$, which means that there is no value of μ_{i+1} that will allow all the roots of $Q_{i+1}(x)$ to lie inside (−1,+1). Thus

$$1 - (\sigma_i^2)_{\max} \frac{Q_{i-1}(+1)}{Q_i(+1)} = -1 - (\sigma_i^2)_{\max} \frac{Q_{i-1}(-1)}{Q_i(-1)}$$

$$2 = (\sigma_i^2)_{\max} \left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)} \right]$$

$$(\sigma_i^2)_{\max} = 2 \ / \ \left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)} \right].$$

We can work back from the limits on μ_i and σ_i^2 to obtain limits on the moments.

$$\sigma_i^2 = N_i/N_{i-1}$$

$$N_i = \sum_{k=0}^{i} a_{ik} M_{k+i} = M_{2i} + \sum_{k=0}^{i-1} a_{ik} M_{k+i} \quad \text{since } a_{ii} = 1 \ .$$

Therefore,

$$0 < \sigma_i^2 < 2 \qquad \left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)} \right]$$

implies

$$- \sum_{k=0}^{i-1} a_{ik} M_{k+i} < M_{2i} < \frac{2N_{i-1}}{\left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)} \right]} - \sum_{k=0}^{i-1} a_{ik} M_{k+i}$$

$$\mu_{i+1} = \frac{L_{i+1}}{N_i} - \frac{L_i}{N_{i-1}}$$

$$L_{i+1} = \sum_{k=0}^{i} a_{ik} M_{k+i+1} = M_{2i+1} + \sum_{k=0}^{i-1} a_{ik} M_{k+i+1}$$

$$\mu_{i+1}^{max} = 1 - \sigma_i^2 \frac{Q_{i-1}(1)}{Q_i(1)} \ ;$$

therefore,

$$L_{i+1}^{max} = N_i - N_i \, \sigma_i^2 \frac{Q_{i-1}(1)}{Q_i(1)} + \frac{N_i L_i}{N_{i-1}} = N_i \left(1 - \sigma_i^2 \frac{Q_{i-1}(1)}{Q_i(1)} \right) + L_i \, \sigma_i^2$$

$$M_{2i+1} < N_i \left(1 - \sigma_i^2 \frac{Q_{i-1}(1)}{Q_i(1)} \right) + L_i \, \sigma_i^2 - \sum_{k=0}^{i-1} a_{ik} M_{k+i+1} \ ;$$

also,

$$M_{2i+1} > N_i \left(- 1 - \sigma_i^2 \frac{Q_{i-1}(-1)}{Q_i(-1)} \right) + L_i \, \sigma_i^2 - \sum_{k=0}^{i-1} a_{ik} M_{k+i+1} \ .$$

To obtain the limits on the Legendre coefficients, take the set of moments already determined M_1, M_2, ..., M_{2i-1} combined with M_{2i}^{max} and convert from moments to Legendre coefficients. This gives f_{2i}^{max}. When M_1, M_2, ..., M_{2i-1} are combined with M_{2i}^{min} and converted, one obtains f_{2i}^{min}.

F9.E DEFINITION OF STORAGE LOCATIONS AND VARIABLE NAMES

The following tables describe the main variables, their storage locations, and the pointers to their first location in core storage.

Table F9.E.1 Storage locations

Pointer[a]	Variable Name	Length
1[b]	ENER	NNGA+NGGA+2
LFP1	VELTH	NNGA+NGGA
LFP2	Initial weight standards	3*MAXGP*MXREG
LFP3	Path parameters	MAXGP*MXREG
LFP4	EPROB	(NNGA+NGGA)*MXREG
LFP5	FWLO	MXREG
LFP6	FSE	(NNGA+NGGA)*MEDIA
LFP7	GWL	NNGA*MXREG
LFP8	FS	NGPFS
LFP9	BFS	NGPFS
LFP10	SIGT	(NNGA+NGGA)*MEDIA
LFP11	SIGS	(NNGA+NGGA)*MEDIA
LFP12	SIGGP	(NNGA+NGGA)*MEDIA
LFP13	SIGNU	(NNGA+NGGA)*MEDIA
LFP14	NRD	(3+4*NSCT+2*NPL)*MEDIA*NSGPS
LFP15	MWA	2*(NNGA+NGGA)*MEDIA
LFP16	Current weight standards	3*MAXGP*MXREG
LFP17	Split and RR counters	8*MAXGP*MXREG
LFP18	Current FWLO	MXREG
LFP19	Scattering counters	8*(NNGA+NGGA)*MXREG
LFP20	NSCA	MEDIA
LFP21	Geometry data	NADD
LFP22	Neutron bank	12*NMOST
LFP23	Fission bank	7*NMOST
LAP1	Analysis data	
LSP1	NSGN	NNGA+NGGA
LSP2	NSGL	2*NSGPS
LSP3	NSGP	NSGPS
LFP24	Supergrouped cross-section data	

[a] In COMMON POINT.

[b] First array location.

Table F9.E.2 Definitions of variables in main storage

Variable Name	Definition
ENER(IG)	Upper energy boundary of group IG (in eV).
VEL(IG)	Velocity corresponding to the mean energy for neutron groups and the speed of light for gamma-ray groups (in cm/s).
FS(IG)	Unbiased source spectrum—unnormalized fraction of source particles in each energy group—transformed to cumulative distribution function (c.d.f.) by SORIN.
BFS(IG)	Biased source spectrum—relative importance of each energy group—transformed to biased c.d.f. by SORIN.
WTHI(IG,NREG)	Weight above which splitting is performed (vs group and region).
WTLO(IG,NREG)	Weight below which Russian roulette is performed (vs group and region).
WTAV(IG,NREG)	Weight to be assigned Russian roulette survivors (vs group and region).
PATH(IG,NREG)	Exponential transform parameters (vs group and region).
NSPL(IG,NREG)	Splitting counter (vs group and region).
WSPL(IG,NREG)	Weight associated with NSPL [i.e., the sum of weights which have split (vs group and region)].
NOSP(IG,NREG)	Counter for full bank when splitting was requested (vs group and region).
WNOS(IG,NREG)	Weight associated with NOSP.
RRKL(IG,NREG)	Russian roulette death counter (vs group and region).
WRKL(IG,NREG)	Weight associated with RRKL.
RRSU(IG,NREG)	Russian roulette survival counter (vs group and region).
WRSU(IG,NREG)	Weight associated with RRSU.
INIWHI(IG,NREG)	Initial values of WTHI array.

Table F9.E.2 (continued)

Variable name	Definition
INIWLO(IG,NREG)	Initial values of WTLO array.
INIWAV(IG,NREG)	Initial values of WTAV array.
FWLO(NREG)	Weights to be assigned to fission daughters (vs region).
INIFLO(NREG)	Initial values of FWLO.
GWLO(IG,NREG)	Weights to be assigned to secondary particles (vs group and region).
EPROB(IG,NREG)	Relative importance of energy groups after scattering (vs group and region). Present only if IEBIAS > 0.
NSCT(IG,NREG)	Number of real scatterings (vs group and region).
WSCT(IG,NREG)	Weight associated with NSCT.
NALB(IG,NREG)	Number of albedo scatterings (vs group and region).
WALB(IG,NREG)	Weight associated with NALB.
NFIZ(IG,NREG)	Number of fissions (vs group and region).
WFIZ(IG,NREG)	Weight associated with NFIZ.
NGAM(IG,NREG)	Number of secondary productions (vs group and region).
WGAM(IG,NREG)	Weight associated with NGAM.
NSCA(IMED)	Scattering counter (vs cross-section medium).
FSE(IG,IMED)	Source spectrum for fission-induced neutrons for each group—input as frequency of group IG.

Table F9.E.3 Neutron bank

Pointer[a]	Variable name
LPN1	US
LPN2	VS
LPN3	WS
LPN4	XS
LPN5	YS
LPN6	ZS
LPN7	WATES
LPN8	AGES
LPN9	BLZNTS
LPN10	IGS
LPN11	NAMES
LPN12	NAMEXS
LPN13	NMEDS
LPN14	NREGS

[a] In COMMON NUTRON.

Table F9.E.4 Layout of analysis data storage area

Location labels	Variable name	Length
LAP1	LABELS	20 * NRESP
LOCRSP	RESP	NRESP * NMTG
	EXTR	NEX * NMTG
LOCXD	XD YD ZD RAD TO FACT	6 * ND
	EXTR	NEXND * ND
LOCIB	IB EP DELE	3 * NE omitted if NE=0
	IARR	NMTG
LOCCO	COS	NA
LOCT	T DELT	2 * ND * NT
LOCUD	UD SUD SUD2	3 * ND * NRESP
LOCSD	SD SSD SSD2	3 * ND * NRESP
	SUD & SSD Units	20
LOCQE	QE SQE SQE2	3 * NE * ND
	SQE Units	20
LOCQT	QT SQT SQT2	3 * NT * ND * NRESP
	SQT Units	20
LOCQTE	QTE SQTE SQTE2	3 * NT * NE * ND
	SQTE Units	20
LOCQAE	QAE SQAE SQAE2	3 * NA * NE * ND
LMAX	SQAE Units	20

Table F9.E.5 Analysis arrays

Mnemonic variable name	Purpose
LABEL (J,NR)	80-character Hollerith label for each response function.
RESP (IG,NR)	Value of response function.
EXTR (IG,NX)	Extra arrays of length NMTG.
XD (ID)	x-coordinate of detector ID.
YD (ID)	y-coordinate of detector ID.
ZD (ID)	z-coordinate of detector ID.
RAD (ID)	Distance from source (assumed to be at XSTRT, YSTRT, ZSTRT) to detector.
TO (ID)	Minimum flight time to detector.
FACT (ID)	Detector-dependent normalization.
EXTR (ID,NXN)	Extra arrays of length ND.
IB (IE)	Energy bin group number.
EP (IE)	Lower energy of bin.
DELE (IE)	Widths of bin.
IARR (IE)	Key to correspondence between analysis group numbers and random walk group numbers.
COS (IA)	Cosine of angle bin limits
T (IT,ID)	Time bin limits
DELT (IT,ID)	Widths of time bins.
UD (NR,ID)	Uncollided response.
SD (NR,ID)	Total response.
QE (IE,ID)	Energy-dependent fluence.
QT (NR,IT,ID)	Time-dependent response.
QTE (IT,IE,ID)	Time- and energy-dependent fluence.
QAE (IA,IE,ID)	Angle- and energy-dependent fluence.

Table F9.E.5 (continued)

Index	Maximum value	Purpose
NR	NRESP	Response function.
IG	NMTG	Energy group.
NX	NEX	Extra array (length NMTG).
ID	ND	Detector.
NXN	NEXND	Extra array (length ND).
IE	NE	Energy bin (one or more groups).
IA	NA	Angle bin.
IT	NT	Time bin.

Table F9.E.6 Definition of variables in Common APOLL

Variable	Definition
DDF	Starting particle weight as determined in SORIN.
DEADWT(5)	The summed weights of the particles at death. The four deaths are: Russian roulette, escape, energy, and age limit. The fifth, DEADWT(5), is unused.
DFF	Normalization for adjoint problems.
ETA	Number of mean-free-path (m.f.p.) between collisions.
ETATH	Distance in cm to the next collision if the particle does not encounter a change in total cross section.
ETAUSD	Flight path in m.f.p. that has been used since the last event.
XTRA(10)	Used for temporarily storing alphanumeric information on "time used" immediately before printing it.
ITERS	Current batch number.
ITIME	Not used.
ITSTR	Switch indicating that secondary fissions are to be the source for the next batch if > 0.
MAXTIM	The elapsed clock time at which the problem is terminated.
MGPREG	Product of number of weight standard groups (MAXGP) and regions (MXREG).
INALB	An index indicating that an albedo scattering has occurred if > 0.
NDEAD(5)	Number of deaths of each type (see DEADWT).
NEWNM	Name of the last particle in the bank.
NGEOM	Not used.
NLAST	The last cell in main storage that was used by the cross-section storage or is set aside for banking.

Table F9.E.6 (continued)

Variable	Definition
NMEM	The location of the next particle in the bank to be processed
NMGP	Not used
NMTG	The total number of energy groups (both primary and secondary) for which there are cross sections
NPSCL(13)	An array of counters of events for each batch: (1) sources generated (2) splittings occurring (3) fissions occurring (4) gamma rays generated (5) real collisions (6) albedo scatterings (7) boundary crossings (8) escapes (9) energy cutoffs (10) time cutoffs (11) Russian roulette kills (12) Russian roulette survivors (13) gamma rays not generated because bank was full
NSIGL	Not used
NXTRA(10)	Not used
NSUP	Pointer to particles in the neutron bank which are in the current supergroup
NINF	Pointer to particles in the neutron bank which are not in the current supergroup

Table F9.E.7 Definitions of variables in Common FISBNK

Variable	Definition
NFISBM	Location of cell zero of the fission bank.
NFISH	Number of fissions produced in the previous batch (generation).
FTOTL	Total fission weight from all collisions in this batch (generation).
FWATE	Total weight of fission neutrons stored in bank in this batch (generation).
WATEF	Weight of fission neutron stored in bank.
AWATE	Average weight of particles in the current batch.
NPART	Number of particles in the current batch.

Table F9.E.8 Definition of variables in NUTRON Common

Variable	Definition
NAME	Particle's first name.
IQ7	(Dummy variable—for alignment of double-precision variables.)
NAMEX	Particle's family name. (Note that particles do not marry.)
IG	Current energy group index.
IGO	Previous energy group index.
NMED	Medium number at current location.
MEDOLD	Medium number at previous location
NREG	Region number at current location.
U,V,W	Current direction cosine.
UOLD,VOLD,WOLD	Previous direction cosines.
X,Y,Z	Current location.
XOLD,YOLD,ZOLD	Previous location.
WATE	Current weight.
OLDWT	Previous weight. (Equal to WTBC if no path-length stretching.)
WTBC	Weight just before current collision.
IBLZN	Current zone number.
IBLZO	Previous zone number.
AGE	Current age.
OLDAGE	Previous age.
LNP1-14	Pointers for neutron bank.
LNP15-21	Pointers for fission bank.

Table F9.E.9 Definitions of variables in Common QDET

LOCRSP	Location of cell zero of response functions.
LOCXD	Location of cell zero of detector positions.
LOCIB	Location of cell zero of energy bins.
LOCCO	Location of cell zero of angle (cosine) bins.
LOCT	Location of cell zero of time bins.
LOCUD	Location of cell zero of array UD.
LOCSD	Location of cell zero of array SD.
LOCQE	Location of cell zero of array QE.
LOCQT	Location of cell zero of array QT.
LOCQTE	Location of cell zero of array QTE.
LOCQAE	Location of cell zero of array QAE.
LMAX	Last cell used in storage.
EFIRST	Upper energy limit of first energy bin.
EGTOP	Upper energy limit of first gamma-ray bin.

F9.F COMBINING MULTIPLE MORSE RUNS

One of the most difficult things to determine is how long a computer run is needed to get acceptable statistics. After completion of a production run, the user may decide that he must have better (or lower) standard deviations. His course of action should be as follows.

The output from the run will contain the next random number in the sequence. This number should be used as the starting random number in a new run. Upon completion of this second run, he can obtain the average of the runs and also the standard deviations. This is done as follows:

Assume a_1 and a_2 are the answers from runs one and two respectively; f_1 and f_2 are the corresponding f.s.d's (or fractional standard deviations) as output; $\sigma_i = a_i f_i$; and c_1 and c_2 are the number of batches per run.

The average answer is then $\bar{x} = \dfrac{a_i * c_1 + a_2 * c_2}{c_1 + c_2}$ and the approximate corresponding

$$\text{f.s.d.} = \frac{\sqrt{\left(\frac{c_1}{c_1 + c_2}\right)^2 \sigma_1^{\,2} + \left(\frac{c_2}{c_1 + c_2}\right)^2 \sigma_2^{\,2}}}{\bar{x}} .$$

Any number of runs may be averaged together by using an expanded form of the above equations: i.e.,

$$\bar{x} = \sum_{i=1}^{N} a_i c_i / \sum_{i=1}^{N} c_i \quad \text{and}$$

$$\text{f.s.d.} = \frac{\sqrt{\sigma^2}}{\bar{x}} = \sqrt{\sum_{i=1}^{N} c_i^{\,2} \sigma_i^{\,2} / \left(\sum_{i=1}^{N} c_i\right)^2 / \bar{x}} ,$$

where N is the number of runs.

NUREG/CR-0200
Revision 6
Volume 2, Section F10
ORNL/NUREG/CSD-2/V2/R6

Computational Physics and Engineering Division

HEATING 7.2 USER'S MANUAL

K. W. Childs

Date Published: March 2000

Prepared for the
Office of Nuclear Material Safety and Safeguards
U.S. Nuclear Regulatory Commission
Washington, DC 20555
under Interagency Agreement DOE 1886-C000-9B
NRC JCN No. B0009

Prepared by the
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37831
managed by
LOCKHEED MARTIN ENERGY RESEARCH CORP.
for the
U.S. DEPARTMENT OF ENERGY
under contract DE-AC04-96OR22464

ABSTRACT

HEATING is a general-purpose conduction heat transfer program written in FORTRAN 77. HEATING can solve steady-state and/or transient heat conduction problems in one-, two-, or three-dimensional Cartesian, cylindrical, or spherical coordinates. A model may include multiple materials, and the thermal conductivity, density, and specific heat of each material may be both time- and temperature-dependent. The thermal conductivity may also be anisotropic. Materials may undergo change of phase. Thermal properties of materials may be input or may be extracted from a material properties library. Heat-generation rates may be dependent on time, temperature, and position, and boundary temperatures may be time- and position-dependent. The boundary conditions, which may be surface-to-environment or surface-to-surface, may be specified temperatures or any combination of prescribed heat flux, forced convection, natural convection, and radiation. The boundary condition parameters may be time- and/or temperature-dependent. General graybody radiation problems may be modeled with user-defined factors for radiant exchange. The mesh spacing may be variable along each axis. HEATING uses a run-time memory allocation scheme to avoid having to recompile to match memory requirements for each specific problem. HEATING utilizes free-form input.

Three steady-state solution techniques are available: point-successive-overrelaxation iterative method with extrapolation, direct-solution (for one-dimensional or two-dimensional problems), and conjugate gradient. Transient problems may be solved using any one of several finite-difference schemes: Crank-Nicolson implicit, Classical Implicit Procedure (CIP), Classical Explicit Procedure (CEP), or Levy explicit method (which for some circumstances allows a time step greater than the CEP stability criterion.) The solution of the system of equations arising from the implicit techniques is accomplished by point-successive-overrelaxation iteration and includes procedures to estimate the optimum acceleration parameter.

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

Page

CONTENTS (continued)

LIST OF FIGURES

LIST OF FIGURES (continued)

LIST OF TABLES

ACKNOWLEDGMENTS

HEATING 7.2 is the latest in a series of codes in which each version was an outgrowth of its immediate predecessor. The current author gratefully acknowledges the contributions of the developers of all antecedents of the current version.

The primary funding source for the development of HEATING 7.2 was obtained from a fee assessed all users of the HEATING code on the CRAY computer at Lockheed Martin Energy Systems, Inc. The largest user, and subsequently the largest funds contributor, was the Engineering Division at Lockheed Martin Energy Systems. Additional funding for porting HEATING to a PC and final documentation was provided by the Engineering Technology, Metals and Ceramics, and Energy Divisions of Oak Ridge National Laboratory (ORNL).

All figures in this report were prepared by R. S. Wallace of the ORNL Computational Physics and Engineering Division (CPED). The manuscript was prepared for publication by A. I. Rose and W. C. Carter of CPED, and edited by C. H. Shappert of Information Resources Division.

F10.1 INTRODUCTION

HEATING is a multidimensional, general-purpose heat transfer code written in FORTRAN 77. The name HEATING is an acronym for Heat Engineering and Transfer In Nine Geometries (although with modifications there are now 12 geometries). Earlier versions of HEATING have been reported previously.[1-9] HEATING solves steady-state and/or transient problems in one-dimensional (1-D), two-dimensional (2-D), or three-dimensional (3-D) Cartesian, cylindrical, or spherical coordinates. The mesh spacing may be variable along each axis. A model may include multiple materials, and the thermal conductivity, density, and specific heat of each material may be both time- and temperature-dependent. Thermal conductivity may be anisotropic. Materials may undergo up to five changes of phase for transient calculations involving either of the explicit procedures. The heat-generation rates may be dependent on time, temperature, and position. Boundary temperatures may be dependent on time and position. Boundary conditions include specified temperatures or any combination of prescribed heat flux, forced convection, natural convection, and radiation. The boundary condition parameters may be time- and/or temperature-dependent. In addition, one may model either 1-D radiative heat transfer directly across gaps embedded in the model or multidirectional radiation within enclosures using externally calculated radiation exchange factors.

The numerical techniques used for the steady-state and transient calculations are discussed in Sect. F10.2. For steady-state problems, there are three solution techniques available: a point-successive-overrelaxation iterative method with a modified Aitken δ^2 extrapolation process, a direct-solution method for 1-D and 2-D problems, and a conjugate gradient method. Several numerical schemes are available in HEATING to solve transient heat-conduction problems. In the implicit technique, which can range from Crank-Nicolson to the Classical Implicit Procedure, the system of equations is solved by point-successive-overrelaxation iteration. The technique includes procedures to estimate the optimum acceleration parameters for both the case where the coefficient matrix remains unchanged during the calculations and the case where it varies during the solution because of dependence on time, temperature, or time-step size. The time-step size for the implicit transient calculations may be controlled so that either the maximum temperature change or the maximum relative temperature change over a time step does not exceed specified values. The time-step size may also be controlled by specifying an initial time step, a multiplication factor to be multiplied by the time-step size after each time level, a maximum allowable time-step size, and/or the time period for which it applies. Transient problems may also be solved either with the Classical Explicit Procedure, whose time step is stability-limited, or with the Levy explicit method, which allows a larger time step for some cases.

HEATING does not have to be recompiled with each execution since computer memory allocation is handled at execution time. Any array whose size is a function of the input data is variably dimensioned. These array dimensions are determined from the model information in the input data file.

HEATING uses free-form reading subroutines[10] to interpret the input data file, which is subdivided into data blocks identified by keywords. HEATING reads the input data file until a keywork specifying a steady-state or transient solution is encountered or until an end-of-case indicator is encountered. The input data are checked for errors and inconsistencies, and messages identifying any problems are written. Some data errors cause processing to be terminated immediately. However, most errors will allow the processing of input data to be completed, but will not allow the case to be executed. If no input data errors are encountered, HEATING will proceed with the specified calculations for the case.

Section F10.3 discusses modeling concepts and gives advice on the use of HEATING. It includes a number of suggestions concerning the use of special features of the code. A comprehensive discussion of the input data is presented in Sect. F10.4. An outline of the input data is included in the section to serve as a guide for the user who is somewhat familiar with the use of the code. A discussion outlining the output features of

HEATING is presented in Sect. F10.5. The appendices include sample problems, instructions for running the code on computers with UNIX operating systems, a discussion of codes available for postprocessing HEATING results, documentation of a material properties library for use with HEATING, and information on some utility codes for use with HEATING.

HEATING is not a "black box" that automatically yields the correct solution to the physical problem. Care must be exercised in order to correctly simulate a physical problem and correctly interpret the results from the code. This version of HEATING has undergone the same extensive verification documented for a previous version[11] of the code. The verification process is discussed further in Appendix F10.B.

F10.2 NUMERICAL TECHNIQUE

F10.2.1 STATEMENT OF THE PROBLEM

The HEATING computer program solves steady-state or transient heat-conduction problems in one, two, or three dimensions. Three coordinate systems are available in HEATING: Cartesian, cylindrical, and spherical. For each problem, only one of the three coordinate systems can be used, and the entire problem geometry must be consistent with the chosen coordinate system. For an arbitrary orthogonal, curvilinear coordinate system with coordinates *u*, *v*, and *w*, the surfaces defined by the equations *u* = *constant*, *v* = *constant*, and *w* = *constant* are referred to as coordinate surfaces. Any number of coordinate surfaces can be defined along each of the axes. In the Cartesian coordinate system the coordinates are *x*, *y*, and *z*; and the coordinate surfaces are planes perpendicular to the respective axis. In the cylindrical coordinate system the coordinates are *r*, θ, and *z*. An *r* coordinate surface is a surface of an infinitely long cylinder. A θ coordinate surface is a semi-infinite plane whose edge coincides with the axis of the cylinder. A *z* coordinate surface is an infinite plane perpendicular to the axis of the cylinder. In the spherical coordinate system the coordinates are *r*, θ, and ϕ. An *r* coordinate surface is the surface of a sphere. A θ coordinate surface is a semi-infinite plane whose edge coincides with the axis of the sphere (the line connecting the poles of the sphere). The θ coordinate surfaces in a spherical geometry are the same as those in a cylindrical geometry. The ϕ coordinate surfaces are the surfaces of cones with their vertices at the center of the sphere and their axes coinciding with the axis of the sphere. The angle ϕ is measured from the equatorial plane of the sphere. The ϕ = 0 coordinate surface is the equatorial plane.

In order to define the nodes, a system of orthogonal coordinate surfaces is superimposed on the problem. The planes may be unequally spaced, but they must extend to the outer boundaries of the problem. An internal node is defined by the intersection of three coordinate surfaces. For illustrative purposes, the equations and the discussion that follow are written for a three-dimensional problem in Cartesian coordinates.

A portion of the grid surrounding an internal node (denoted by *o*) is depicted in Fig. F10.2.1. Heat may flow from this node to each adjacent node along paths that are parallel to each axis. Thus for a 3-D problem, heat may flow from an internal node to each of its six neighboring nodes. The nodes with which node *o* exchanges heat are denoted by the numbers 1 through 6 in Fig. F10.2.1. The system of equations describing the temperature distribution is derived by performing a heat balance about each node. The volume associated with a node is bounded by coordinate surfaces lying midway between the coordinate surfaces defining the node locations. The volume associated with node *o* is shown in Fig. F10.2.2.

The finite-volume heat balance equation for node *o* is

$$C_o \frac{T_o^{n+1} - T_o^n}{\Delta t} = P_o + \sum_{m=1}^{6} {}_oK_m (T_m^n - T_o^n) \ , \qquad \text{(F10.2.1)}$$

where T_m^n is the temperature of the *m*th node adjacent to node *o* at time t_n, ${}_oK_m$ is the conductance between nodes *o* and *m*, C_o is the heat capacitance associated with node *o*, P_o is the heat generation rate, and Δt is the time step $(t_{n+1} - t_n)$. Equation (F10.2.1) approximates the transient heat conduction equation using a first-order accurate, explicit, forward Euler integration over time. For a uniform grid spacing, the spatial approximation is equivalent to a central difference which is second-order accurate.

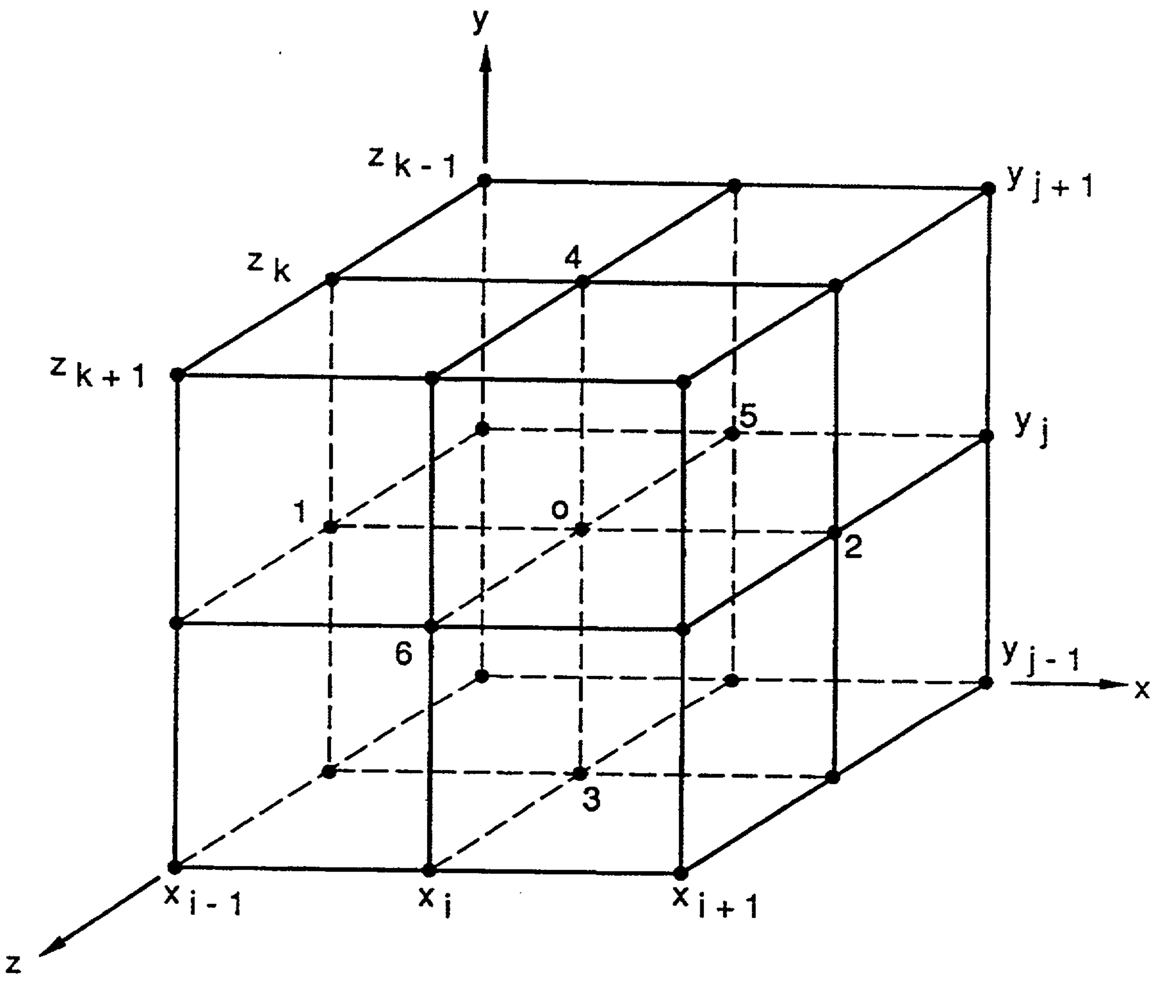


Figure F10.2.1 Grid structure surrounding node of interest

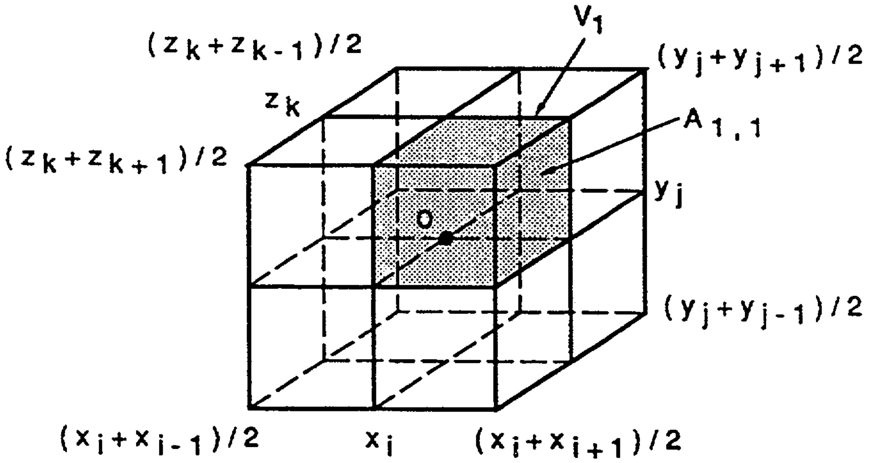


Figure F10.2.2 Volume associated with node of interest (one of octants comprising volume is shaded)

The shaded volume in Fig. F10.2.2 indicates one of the eight subvolumes (or octants) which make up the total nodal volume. Each of the octants may contain a different material. Thus, a node may be composed of as many as eight different materials, and the heat flow path between adjacent nodes may be composed of as many as four different materials positioned in parallel. For a 3-D problem, one *C*, one *P*, and six *K*s will be associated with each internal node. These parameters are calculated as follows for node *o*:

$$C_o = \sum_{\ell=1}^{8} c_{p\ell}\, \rho_\ell\, V_\ell \ , \qquad \text{(F10.2.2)}$$

$$P_o = \sum_{\ell=1}^{8} Q_\ell\, V_\ell \ , \qquad \text{(F10.2.3)}$$

$${}_oK_m = \frac{1}{L_m} \sum_{\gamma=1}^{4} k_{m,\gamma}\, A_{m,\gamma} \ , \qquad \text{(F10.2.4)}$$

where

$c_{p\ell}$ = specific heat of material in the ℓth octant,

ρ_ℓ = density of material in the ℓth octant,

V_ℓ = volume of the ℓth octant of node *o*,

Q_ℓ = heat generation rate per unit volume in ℓth octant,

L_m = distance between node *o* and adjacent node *m*,

$k_{m,\gamma}$ = thermal conductivity of material in the γth of four heat-flow paths between nodes *o* and *m*,

$A_{m,\gamma}$ = cross-sectional area of the γth heat flow path between nodes *o* and *m*.

With reference to Fig. F10.2.2, the V_ℓ's and $A_{m,\gamma}$'s are further defined, by using examples, as follows:

$$V_1 = \left(\frac{x_{i+1} - x_i}{2}\right) \times \left(\frac{y_{j+1} - y_j}{2}\right) \times \left(\frac{z_{k+1} - z_k}{2}\right) \ ; \qquad \text{(F10.2.5)}$$

$$A_{1,1} = \left(\frac{y_{j+1} - y_j}{2}\right) \times \left(\frac{z_{k+1} - z_k}{2}\right) \ . \qquad \text{(F10.2.6)}$$

Since nodes lying on the surface of a model or nodes from 1- or 2-D problems will not have six neighbors, the general heat balance equation for node i having M_i neighbors can be written as

$$C_i \frac{T_i^{n+1} - T_i^n}{\Delta t} = P_i^n + \sum_{m=1}^{M_i} {}_iK_{\alpha_m}(T_{\alpha_m}^n - T_i^n) \ , \qquad \text{(F10.2.7)}$$

where α_m is the mth neighbor of the ith node. By choosing the increments between grid lines small enough, the solution to the system of equations yields an acceptable approximation to the governing differential equation.

F10.2.2 STEADY-STATE HEAT CONDUCTION

For a steady-state heat conduction problem, the heat balance equation reduces to

$$P_i + \sum_{m=1}^{M_i} {}_iK_{\alpha_m}(T_{\alpha_m} - T_i) = 0 \ . \qquad \text{(F10.2.8)}$$

If there are N nodal points, Eq. (F10.2.8) will yield a system of N equations with N unknowns. HEATING contains three techniques for solving this system of equations. The first technique involves point-successive-overrelaxation (SOR) iteration with a modified Aitken δ^2 extrapolation process. This method can be used for any steady-state problem. However, for certain classes of problems convergence may be slow or the convergence criterion may be unreliable since it only requires that the maximum relative change in temperature from one iteration to the next be less than the specified value. Such difficulties can arise in problems with widely varying parameters such as thermal conductivity, grid spacing along an axis, or nonlinear boundary conditions. Thus, direct-solution and conjugate gradient techniques are also available to solve the steady-state system of equations. The direct-solution technique is available only for 1- and 2-D problems, but it is not always more efficient than the successive-overrelaxation method since it can require a large amount of computer memory since the bandwidth is large. A large bandwidth can occur for 2-D calculations with a large number of lattice lines along the x (or r) axis, or with heat transfer across a gap in the y (or θ) or z (or ϕ) direction. The conjugate gradient technique is available for all geometries.

F10.2.2.1 Point-Successive-Overrelaxation Iteration

Solving Eq. (F10.2.8) for T_i yields

$$T_i = \frac{P_i + \sum_{m=1}^{M_i} {}_iK_{\alpha_m} T_{\alpha_m}}{\sum_{m=1}^{M_i} {}_iK_{\alpha_m}} \ . \qquad \text{(F10.2.9)}$$

Since the values of T_α are unknown, the temperature at node i cannot be calculated directly. However, an iterative procedure can be used to estimate the steady-state temperature distribution. If an estimate to the temperature distribution exists, then Eq. (F10.2.9) can be applied at each node, and a better estimate to the temperature distribution may be obtained. This new estimate can be used in Eq. (F10.2.9) to produce a better estimate. This procedure is known as Jacobi iteration, or the method of simultaneous substitutions, and can be written as

$$T_i^{(n+1)} = \frac{P_i + \sum_{m=1}^{M_i} {}_iK_{\alpha_m} T_{\alpha_m}^{(n)}}{\sum_{m=1}^{M_i} {}_iK_{\alpha_m}} \quad , \tag{F10.2.10}$$

where the superscript (n) implies the nth iterate. It can be observed in Eq. (F10.2.10) that some components of $T_i^{(n+1)}$ are known during the iterative sweep but are not used. The Gauss-Seidel method represents a modification of Jacobi iteration by using the most recent values of T_i during the iteration. The term "successive iteration" is commonly employed in conjunction with Gauss-Seidel iteration to denote the fact that new components of the unknown vector T_i are successively used as they are obtained. Gauss-Seidel iteration is defined as

$$T_i^{(n+1)} = \frac{P_i + \sum_{m=1}^{L_i} {}_iK_{\alpha_m} T_{\alpha_m}^{(n+1)} + \sum_{m=L_i+1}^{M_i} {}_iK_{\alpha_m} T_{\alpha_m}^{(n)}}{\sum_{m=1}^{M_i} {}_iK_{\alpha_m}} \quad , \tag{F10.2.11}$$

where L_i is defined so that $\alpha_m < i$ for $m \le L_i$; and $\alpha_m > i$ for $m > L_i$. Instead of using the results of Eq. (F10.2.11) as the $(n + 1)$st iterate, assume that it only yields an intermediate estimate and denote it as $\hat{T}_i^{(n+1)}$. Then define the $(n + 1)$st iterate to be

$$T_i^{(n+1)} = T_i^{(n)} + \beta[\hat{T}_i^{(n+1)} - T_i^{(n)}] \quad . \tag{F10.2.12}$$

The accelerated Gauss-Seidel technique can then be expressed as

$$T_i^{(n+1)} = (1 - \beta)T_i^{(n)} + \beta\left[\frac{P_i + \sum_{m=1}^{L_i} {}_iK_{\alpha_m} T_{\alpha_m}^{(n+1)} + \sum_{m=L_i+1}^{M_i} {}_iK_{\alpha_m} T_{\alpha_m}^{(n)}}{\sum_{m=1}^{M_i} {}_iK_{\alpha_m}}\right] . \tag{F10.2.13}$$

Varga[12] refers to this method as the point-successive-overrelaxation method (SOR). To increase the rate of convergence, an exponential approximation for Eq. (F10.2.13) is made based on the temperature change from one iteration to the next. The algorithm based on this approximation is used instead of Eq. (F10.2.13) to calculate the new temperatures for nodes having relative temperature changes exceeding 10^{-3}. However, to prohibit the technique from diverging because of a bad estimate of the initial temperature distribution, the algorithm is designed to bound the temperature change so that the new temperature cannot be more than twice the old temperature. The exponential approximation reduces to Eq. (F10.2.13) for small temperature changes. Successive iterations are carried out until

$$\left| \frac{T_i^{(n)} - T_i^{(n+1)}}{T_i^{(n+1)}} \right|_{\textit{maximum for all nodes}} \leq \epsilon \ , \tag{F10.2.14}$$

where ϵ is the specified convergence criterion.

Since the coefficient matrix produced by the heat balance equation is symmetric and positive definite, the Ostrowski-Reich theorem assures convergence if the acceleration factor β is limited to $0 < \beta < 2$. An optimum value of β in Eq. (F10.2.13) is difficult to obtain. If an input value is not supplied for β, the default value is 1.9. If the rate of convergence is slow, β is reduced by 0.1. During an extrapolation cycle, the relative temperature change over an iteration is monitored over ten consecutive iterations. If the relative temperature changes do not decrease monotonically over these ten iterations, then the current relative temperature change is compared with the one arising ten iterations earlier. If the current relative temperature change is greater than two-thirds of the old one, then the SOR technique is assumed to be converging slowly. This process may be repeated until $\beta = 1.0$. However, the code will not increase β.

Another extrapolation procedure used to increase the rate of convergence in an iterative solution to a system of equations is the "Aitken δ^2 extrapolation procedure." If $T^{(n-1)}$, $T^{(n)}$, and $T^{(n+1)}$ are the temperatures at a certain point at the (*n*-1)st, n*th* and (*n*+1)st iterations, respectively, and if

$$\left| T^{(n)} - T^{(n-1)} \right| > \left| T^{(n+1)} - T^{(n)} \right| \tag{F10.2.15}$$

and

$$\left[T^{(n+1)} - T^{(n)}\right]\left[T^{(n)} - T^{(n-1)}\right] > 0 \ , \tag{F10.2.16}$$

then a better estimate of the temperature is

$$T_{new} = T^{(n+1)} + \frac{\left[T^{(n+1)} - T^{(n)}\right]^2}{\left[T^{(n)} - T^{(n-1)}\right] - \left[T^{(n+1)} - T^{(n)}\right]} \ . \tag{F10.2.17}$$

HEATING uses a modification of the Aitken δ^2 method by calculating an extrapolation factor, *B*, and approximating Eq. (F10.2.17) at each node with

$$T_i^{(n+1)} = \tilde{T}_i^{(n+1)} + B\left[\tilde{T}_i^{(n+1)} - T_i^{(n)}\right] \ , \tag{F10.2.18}$$

where $\hat{T}_i^{(n+1)}$ represents the $(n+1)$st iterate at node i before extrapolation.

An extrapolation cycle is defined as follows: The code completes 20 iterations and checks to see if the maximum of the absolute values (i.e., the sub-norm) of the relative temperature changes over an iteration has decreased monotonically over the last ten iterations. If not, the cycle starts over. If so, the code will extrapolate only if the relative change in extrapolation factors over two consecutive iterations is less than 5% and the sub-norm of the relative temperature changes decreases monotonically over the same two iterations. The extrapolation factor, B, which is the same for each node, is based on the maximum relative temperature change for two consecutive iterations.

F10.2.2.2 Direct-Solution Technique

If Eq. (F10.2.8) is rewritten with the heat-generation term P_i and the terms associated with the boundary temperatures on the right-hand side of the equal sign, the system of equations can be represented in matrix form as

$$A\vec{t} = \vec{b} \; , \qquad \text{(F10.2.19)}$$

where A is the square coefficient matrix, $\vec{t}$ is the column vector of unknown temperatures, and $\vec{b}$ is the column vector containing the forcing functions, consisting of terms related to heat-generation rates and boundary conditions. Matrix A is banded, symmetric, and positive-definite. When the bandwidth is not large, the system of equations can be solved efficiently using a direct-solution technique. HEATING contains a direct-solution technique to solve 1- and 2-D steady-state problems. The bandwidth on most 3-D problems will be quite large, leading to excessive computer memory and computing time requirements. Thus the direct-solution technique has not been implemented for 3-D problems.

After building the banded coefficient matrix, HEATING determines the bandwidth and calls the appropriate subroutine from a library of simultaneous linear algebraic equation solvers.[13] Most 1-D problems have a tridiagonal coefficient matrix and are solved using a symmetric, positive-definite, tridiagonal matrix subroutine. Otherwise, systems of equations are solved using general symmetric, positive-definite, banded matrix subroutines.

For nonlinear problems, the terms P_i and ${}_iK_{\alpha_m}$ in Eq. (F10.2.8) may be a function of the unknown temperatures of the associated nodes. This system of nonlinear equations may be reduced to a system of linear equations by evaluating P_i and ${}_iK_{\alpha_m}$ at estimates of the nodal temperatures. New estimates for nodal temperatures are then determined using one of the techniques mentioned above. In an iterative process the terms P_i and ${}_iK_{\alpha_m}$ are reevaluated using the newest nodal temperature distribution, and the new system of linear equations is solved. This iterative procedure can be represented in matrix form by denoting the nth iterate of the banded coefficient matrix, the temperature vector, and the vector of forcing functions as $A^{(n)}$, $\vec{t}^{\,(n+1)}$, and $\vec{b}$, respectively. Then,

$$A^{(n)} = A\left(\vec{t}^{\,(n)}\right) \qquad \text{(F10.2.20)}$$

and

$$\vec{b}^{\,(n)} = \vec{b}\left(\vec{t}^{\,(n)}\right) . \qquad \text{(F10.2.21)}$$

The coefficients in matrix A and vector $\vec{b}$ are evaluated using the nth iterate of the temperature vector. The system of equations

$$A^{(n)}\vec{t} = \vec{b}^{(n)} \qquad \text{(F10.2.22)}$$

is solved to yield $\vec{t}^{(n+1)}$. This process, referred to as functional or direct iteration, is repeated until the convergence criterion has been satisfied.

The convergence criterion was developed by Becker.[14] The following discussion was taken from that reference. Define a heat residual vector, $\vec{r}$, at the nth iteration as

$$\vec{r}^{(n)} = A^{(n)}\vec{t}^{(n)} - \vec{b}^{(n)} \; . \qquad \text{(F10.2.23)}$$

This vector contains the heat residual, $\vec{r}^{(n)}$, at each node. From Eqs. (F10.2.8) and (F10.2.23), the ith component of vector $\vec{r}$ is

$$r_i^{(n)} = \sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{(n)}\left(T_{\alpha_m}^{(n)} - T_i^{(n)}\right) + P_i^{(n)} \; . \qquad \text{(F10.2.24)}$$

Making this residual sufficiently small at each node is assumed equivalent to approaching the correct temperature at each node. To scale this nodal residual for problems involving large nodal heat flows, an average heat flow at each node is calculated as the average of the absolute values of all the heat flows into and out of the node, including heat generation. A relative residual is then determined at each node as the residual at that node divided by the average heat flow at that node. Thus at each node two measures of accuracy are calculated: a residual and a relative residual. The error at each node is determined as the lesser of either the residual at that node or the relative residual at that node. Then the maximum error is determined over all nodes in the model. If this maximum error is less than the convergence criterion, the calculation is considered converged.

Experience has shown that a residual on the order of 10^{-10} to 10^{-12} is the minimum error that can be achieved because of machine imprecision in the calculations using eight-byte real numbers. It is recommended that the user run a problem with two or three different convergence criteria (such as 10^{-6}, 10^{-8}, 10^{-10}) to gain confidence in the answers given by the code.

F10.2.2.3 Conjugate Gradient Method

A conjugate gradient technique is implemented in HEATING.[15] For a well-posed problem the linear conjugate gradient technique is "guaranteed" to converge in N iterations, where N is the number of unknowns. For the conjugate gradient technique, the system of equations is written in the same matrix form as used with the direct-solution technique:

$$A\vec{t} = \vec{b} \; . \qquad \text{(F10.2.25)}$$

Initially the residual vector, $\vec{r}$, is calculated based on the temperature vector, $\vec{t}$, and the direction vector, $\vec{p}$, is set equal to the residual vector:

$$\vec{r}^{\,0} = \vec{b} - A\vec{t}^{\,0} , \qquad \text{(F10.2.26)}$$

$$\vec{p}^{\,0} = \vec{r}^{\,0} . \qquad \text{(F10.2.27)}$$

The iteration steps for $k = 0,1,...$ are as follows:

1. determine distance to move along direction vector based on a 1-D minimization and calculate new solution estimate,

$$\alpha_k = -(\vec{r}^{\,k},\vec{r}^{\,k})/(\vec{p}^{\,k},A\vec{p}^{\,k}) , \qquad \text{(F10.2.28)}$$

$$\vec{t}^{\,k+1} = \vec{t}^{\,k} - \alpha_k \vec{p}^{\,k} ; \qquad \text{(F10.2.29)}$$

2. calculate a new residual vector,

$$\vec{r}^{\,k+1} = \vec{r}^{\,k} - \alpha_k A\vec{p}^{\,k} ; \qquad \text{(F10.2.30)}$$

3. check for convergence, if

$$\max\left|r_i^{\,k+1}\right|_{1\le i\le N} \ge \epsilon ; \qquad \text{(F10.2.31)}$$

continue,

4. calculate a new direction vector,

$$\beta_k = \frac{(\vec{r}^{\,k+1},\vec{r}^{\,k+1})}{(\vec{r}^{\,k},\vec{r}^{\,k})} , \qquad \text{(F10.2.32)}$$

$$\vec{p}^{\,k+1} = \vec{r}^{\,k+1} + \beta_k \vec{p}^{\,k} . \qquad \text{(F10.2.33)}$$

F10.2.3 TRANSIENT HEAT CONDUCTION

HEATING is designed to solve a transient problem by any one of several numerical schemes. The first is the Classical Explicit Procedure (CEP), which involves the first forward difference with respect to time and is stable only when the time step is smaller than the stability criterion. Levy's modification to the CEP is the second scheme, and it requires the temperature distribution at two times to calculate the temperatures at the new time level. The technique is stable for a time step of any size. The third procedure, which is written generally, contains several implicit techniques that are stable for a time step of any size. One can use the Crank-Nicolson technique, the Classical Implicit Procedure (CIP) (also referred to as backwards Euler), or a linear combination of the two. The resulting system of equations is solved by point-successive-overrelaxation iteration. Techniques have been included in the code to approximate the optimum acceleration parameter. Change-of-phase problems can be modeled with either of the explicit solution techniques. A change-of-phase modeling procedure has not been implemented in the implicit procedures in HEATING.

Equation (F10.2.7) is the basic heat balance equation for transient problems. However, the right-hand side is modified for all but the CEP. The Crank-Nicolson implicit procedure is the recommended technique for solving transient problems.

F10.2.3.1 Classical Explicit Procedure

For a transient heat-conduction problem, the heat balance equation, Eq. (F10.2.7), can be solved for T^{n+1} to give

$$T_i^{n+1} = T_i^n + \frac{\Delta t}{C_i}\left[P_i^n + \sum_{m=1}^{M_i} {}_iK_{\alpha_m}(T_{\alpha_m}^n - T_i^n) \right] . \qquad \text{(F10.2.34)}$$

Since Eq. (F10.2.34) expresses the temperature of node i at the $(n+1)$st time level in terms of temperatures at the nth time level, it is an explicit technique, and the algorithm is known as the CEP or the forward Euler time integration. The numerical solution obtained by using this technique is stable, provided the time step satisfies the following inequality.[16]

$$\Delta t \le \left[\frac{C_i}{\sum_{m=1}^{M_i} {}_iK_{\alpha_m}} \right]_{\text{minimum for all nodes}} . \qquad \text{(F10.2.35)}$$

The stability criterion is also a function of heat generation and heat flux, but this stability is not accounted for when calculating a limiting time step in HEATING.

F10.2.3.2 Levy's Modification to the Classical Explicit Procedure

The limitation on the size of the time step in Eq. (F10.2.35) results in some computation costs becoming so high that the use of the algorithm defined by Eq. (F10.2.34) becomes impractical. Levy[17] proposed an explicit method that is stable for any time step. The basic equation is

$$T_i^{n+1} = T_i^n + \frac{1}{1+Z_i}\left\{ \frac{\Delta t}{C_i}\left[P_i^n + \sum_{m=1}^{M_i} {}_iK_{\alpha_m}(T_{\alpha_m}^n - T_i^n)\right] + Z_i[T_i^n - T_i^{n-1}]\right\}, \qquad \text{(F10.2.36)}$$

where

$$Z_i = \begin{cases} 0 & , \dfrac{\Delta t}{(\Delta t_{\max})_i} \le 1 \\ 0.5\left[\dfrac{\Delta t}{(\Delta t_{\max})_i} - 1\right] & , \dfrac{\Delta t}{(\Delta t_{\max})_i} > 1 \end{cases} . \qquad \text{(F10.2.37)}$$

$(\Delta t_{\max})_i$ is the maximum time step, from Eq. (F10.2.35), allowed at node i for a stable solution in the CEP method.

According to Levy, the accuracy is good if Z_i is zero for somewhat over half of the nodes (i.e., the time step should be smaller than the CEP stability criterion for somewhat over half the nodes). Although it is not immediately obvious from Eq. (F10.2.36), the Levy technique actually calculates the nodal temperature change over a time step as the weighted average of the nodal temperature change calculated from an energy balance for the current time step and the nodal temperature change over the previous time step. The weighting factors are one for the current time step and Z_i for the previous time step. Thus, the Levy technique does not rigorously enforce conservation of energy at nodes where Z_i is nonzero.

The code prints out a message giving the minimum, median, and maximum nodal stability criteria, as well as a warning if the entered Levy factor results in a time step larger than the median nodal stability criterion. Of course, one must experiment with the size of the time step in order to obtain an acceptable solution even when the time step is less than the median stability criterion.

F10.2.3.3 Implicit Procedure

F10.2.3.3.1 Heat-balance equation

If the right-hand side of Eq. (F10.2.7) is evaluated at t_{n+1} instead of t_n, then the technique is the CIP or backward Euler time integration. If the right-hand side of Eq. (F10.2.7) is evaluated at $t_{n+½}$, then the algorithm is the Crank-Nicolson (CN) procedure or trapezoidal rule. A general algorithm that includes both the CN technique and the CIP is

$$C_i^{n+\Theta}\frac{T_i^{n+1} - T_i^n}{\Delta t} = P_i^{n+\Theta} + \Theta\left[\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}(T_{\alpha_m}^{n+1} - T_i^{n+1})\right] + (1-\Theta)\left[\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}(T_{\alpha_m}^{n} - T_i^{n})\right] \qquad \text{(F10.2.38)}$$

where $0 \le \Theta \le 1$ and where the superscript $n + \Theta$ implies that the parameter is evaluated at time $t_{n+\Theta}$. If $\Theta = 0.5$, then Eq. (F10.2.38) becomes the CN technique and if $\Theta = 1.0$, the algorithm is the CIP. When $0 < \Theta < 0.5$, the technique is no longer stable. Note that Eq. (F10.2.38) reverts to Eq. (F10.2.7) when $\Theta = 0$. This algorithm has been incorporated into HEATING for $0.5 \le \Theta \le 1.0$.

F10.2.3.3.2 Numerical technique

If there are N nodes in the problem, Eq. (F10.2.38) yields N equations and N unknowns, and the resulting system of equations can be solved iteratively. If Eq. (F10.2.38) is rewritten so that the temperatures at t_{n+1} are on the left-hand side, then the equation becomes

$$-\Theta\left[\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}^{n+1}\right] + \left[\frac{C_i^{n+\Theta}}{\Delta t} + \Theta\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\right] T_i^{n+1} = H_i\,, \tag{F10.2.39}$$

where

$$H_i = \frac{C_i^{n+\Theta}}{\Delta t} T_i^n + P_i^{n+\Theta} + (1-\Theta)\left[\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\left(T_{\alpha_m}^n - T_i^n\right)\right]. \tag{F10.2.40}$$

Defining

$$D_i = \frac{C_i^{n+\Theta}}{\Delta t} + \Theta\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\,, \tag{F10.2.41}$$

and omitting the superscript, n+1, on the temperature, T, Eq. (F10.2.39) can be rewritten as

$$-\Theta\left[\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}\right] + D_i T_i = H_i\,, \tag{F10.2.42}$$

where T_i represents the temperature of node i at the new time level. Solving for T_i,

$$T_i = \frac{\Theta\left[\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}\right] + H_i}{D_i}\,. \tag{F10.2.43}$$

Since the values of T_{α_m} are unknown, one cannot solve directly for the temperature at node i. However, if an initial estimate of the temperature distribution at the new time level exists, then Eq. (F10.2.43) can be solved at each node to produce a new estimate for the temperature distribution. The procedure can be repeated using this new estimate. This process can be continued until the estimates have converged to the approximation of the temperature distribution at the new time level. This algorithm can be written as

$$T_i^{(n+1)} = \frac{\Theta\left[\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}^{(n)}\right] + H_i}{D_i}\,, \tag{F10.2.44}$$

where the superscript in parentheses refers to the iteration level.

The iterative method resulting from applying Eq. (F10.2.44) is known as Jacobi iteration, or the method of simultaneous substitutions, as discussed in Sect. F10.2.2.1. Instead of using Eq. (F10.2.44) in the iterative process which exhibits a relatively slow convergence rate, the technique can be refined further. Some components of $T_i^{(n)}$ are known during the Jacobi sweep but are not used. The Gauss-Seidel method represents a modification of Jacobi iteration by using the most recent values of T_i during the iteration. The term "successive iteration" is commonly employed in conjunction with Gauss-Seidel iteration to denote the fact that new components of the unknown vector T_i are successively used as they are obtained. The Gauss-Seidel iteration is depicted in Eq. (F10.2.45).

$$T_i^{(n+1)} = \frac{\Theta\left[\sum_{m=1}^{L_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}^{(n+1)} + \sum_{m=L_{i+1}}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}^{(n)}\right] + H_i}{D_i} , \qquad \text{(F10.2.45)}$$

where L_i is defined so that $\alpha_m < i$ for $m \le L_i$; and $\alpha_m > i$ for $m > L_i$. Acceleration of the Gauss-Seidel procedure can produce improved convergence rates. Instead of using the result of Eq. (F10.2.45) as the $(n + 1)$st iterate, it is treated as an intermediate estimate denote by $\hat{T}_i^{(n+1)}$

$$\hat{T}_i^{(n+1)} = \frac{\Theta\left[\sum_{m=1}^{L_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}^{(n+1)} + \sum_{m=L_{i+1}}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}^{(n)}\right] + H_i}{D_i} . \qquad \text{(F10.2.46)}$$

The $(n+1)$st iterate is defined as

$$T_i^{(n+1)} = T_i^{(n)} + \omega[\hat{T}_i^{(n+1)} - T_i^{(n)}] \qquad \text{(F10.2.47)}$$

or

$$T_i^{(n+1)} = (1 - \omega)T_i^{(n)} + \omega\hat{T}_i^{(n+1)} , \qquad \text{(F10.2.48)}$$

where ω is an acceleration factor. Since the coefficient matrix produced by the heat balance equation is symmetric, positive-definite, the Ostrowski-Reich theorem assures convergence if the acceleration factor is limited to $0 < \omega < 2$. The change in the estimated temperature distribution is greater than $(\omega > 1)$ or less than $(\omega < 1)$ the Gauss-Seidel update. The technique is referred to as underrelaxation and overrelaxation for $\omega < 1$ and $\omega > 1$, respectively. For $\omega = 1$, Gauss-Seidel iteration is recovered. Combining Eqs. (F10.2.46) and (F10.2.48) yields

$$T_i^{(n+1)} = (1 - \omega)T_i^{(n)} + \omega \frac{\Theta\left[\sum_{m=1}^{L_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}^{(n+1)} + \sum_{m=L_{i+1}}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}^{(n)}\right] + H_i}{D_i} \tag{F10.2.49}$$

The method described by Eq. (F10.2.49) is referred to as the point-successive-overrelaxation iterative method or SOR. HEATING applies Eq. (F10.2.49) to all nodes until the convergence criterion has been met.

The convergence criterion is derived as follows. When the *n*th iteration has been completed, substitute the *n*th iterate into Eq. (F10.2.42) and denote the heat residual as

$$R_i^{(n)} = H_i + \Theta\left[\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}^{(n)}\right] - D_i T_i^{(n)} \ . \tag{F10.2.50}$$

Normalizing the heat residual by dividing by the right-hand side of Eq. (F10.2.39) yields

$$\frac{R_i^{(n)}}{H_i} = \frac{H_i + \Theta\left[\sum_{m=1}^{M_i} {}_iK_{\alpha_m}^{n+\Theta}\, T_{\alpha_m}^{(n)}\right] - D_i T_i^{(n)}}{H_i} \ . \tag{F10.2.51}$$

The convergence criterion based on the maximum normalized heat residual occurring at any node is

$$\left|\frac{R_i^{(n)}}{H_i}\right|_{\textit{maximum for all nodes}} \leq \epsilon_1 \ . \tag{F10.2.52}$$

If H_i is zero for a particular node, then the denominator is calculated as the L_1 norm of the numerator (the sum of the absolute value of the terms in the numerator divided by the total number of terms). If this is also zero, the node is not used in the test for convergence.

For the first time step, the starting estimate is equal to the initial temperature distribution. Thereafter, the starting estimate at t_{n+1} is determined by

$$\hat{T}_i^{n+1} = T_i^n + (T_i^n - T_i^{n-1})\frac{\Delta t_{n+1}}{\Delta t_n} \ . \tag{F10.2.53}$$

F10.2.3.3.3 Temperature-dependent properties

For problems involving temperature-dependent thermal properties, the thermal properties are initially evaluated at the initial temperatures. Then the point-successive-overrelaxation iterative method is applied to obtain the temperature distribution at the new time level. However, since none of the thermal properties are updated during this procedure, the converged temperatures are only an estimate to the temperature distribution. Thus the thermal properties are reevaluated, and the entire procedure is repeated until the technique converges to the temperature distribution at the new time level. This process contains two levels of iteration. The inner loop is the basic iterative process in the point-successive-overrelaxation iterative method; the outer loop iterates on the thermal properties. Let $T_i^{n,m}$ denote the temperature of node i after the mth iteration on the outer loop at time t_n. For the (m+1)st iteration on the outer loop, the temperature at which the thermal properties are evaluated is calculated as

$$\hat{T}_i^{n+\Theta} = (1 - \Theta)T_i^n + \Theta T_i^{n+1,m} \ . \qquad \text{(F10.2.54)}$$

The temperature distribution has converged at time t_{n+1} when the relative temperature change averaged over all nodes is less than a prescribed value for successive iterations or

$$\frac{1}{N}\sum_{i=1}^{N}\frac{|T_i^{n+1,m} - T_i^{n+1,m-1}|}{|T_i^{n+1,m} - T_i^n|} \le \epsilon_2 \ . \qquad \text{(F10.2.55)}$$

To avoid division by zero or very small values, the actual test used in the code is

$$\frac{1}{N}\sum_{i=1}^{N}\frac{|T_i^{n+1,m} - T_i^{n+1,m-1}|}{\max\left(|T_i^{n+1,m} - T_i^n|, 1.0\right)} \le \epsilon_2 \ . \qquad \text{(F10.2.56)}$$

F10.2.3.3.4 Estimation of the optimum acceleration parameter

The rate of convergence of the point-successive-overrelaxation iterative method is strongly dependent on the value of the acceleration parameter ω. The optimum value of the parameter, denoted as ω_o, is a function of time for problems whose effective thermal conductances and capacitances vary with time or temperature or whose time step changes during the calculations. Several techniques have been developed to estimate ω_o for transient problems with constant thermal properties. The method developed by Carre[18] has been incorporated into HEATING. This method consists of estimating ω_o based on the behavior of a norm of the residual vector during the iterative procedure. The estimates are computed as a function of the iteration number until the process converges to a best estimate of the optimum value. Thereafter, the code uses this converged value as the acceleration parameter.

This process was not satisfactory for problems involving temperature- and time-dependent conductances and capacitances or for problems whose time step changes during the calculations, so an empirical process was developed and added to HEATING to estimate ω_o. For the initial time step, ω is equal to unity. The code will attempt to update ω every N_ω (an input value) time steps. The criteria which are applied to determine whether or not the current value of ω is a good estimate to ω_o are based on the number of iterations required for the inner iterative loop to converge on the first pass through the outer iterative loop at some time step. When the code determines that an attempt to update ω should be made after completion of a particular time step, then the number of iterations for this time step is compared to the number for the time step immediately following the last modification to ω. If the change in the number of iterations is equal to or exceeds the criterion I_ω (an input value), then ω is increased according to

$$\omega^{n+1} = \omega^{n} + 0.1\left(2.0 - \omega^{n}\right) , \qquad \text{(F10.2.57)}$$

where the superscript n refers to the value of ω at time t_n. On each subsequent time step a new estimate is made for ω_0 using an algorithm similar to Eq. (F10.2.57). However, ω may be either increased or decreased according to Table F10.2.1. When the change in the number of iterations for two consecutive time steps is less than the criterion J_ω (an input value), the code assumes that it has a good estimate for ω_0 and uses this value for the subsequent time steps until it is time to attempt another ω update.

Table F10.2.1. Logic to determine whether ω should be increased or decreased

Last update resulted in ω being	Number of iterations compared to previous time step	
	Increased	Decreased
Increased	Decrease	Increase
Decreased	Increase	Decrease

F10.2.3.3.5 Variable time-step size

The time-step size can be varied several ways during the implicit transient calculations. It can be controlled explicitly by specifying it as a constant size during the entire calculation or during prescribed time intervals throughout the transient. The time-step size can also be varied explicitly by multiplying it by a prescribed factor after each time step subject to maximum and minimum constraints.

The time-step size can be varied implicitly in two ways. One can specify the maximum temperature change and the maximum percentage change in temperature allowed at a node over a time step. The time-step size is decreased if one of the calculated maximum values exceeds its respective criterion. It is increased if both of the calculated maximum values are less than their respective criteria. If one of the calculated values exceeds its respective criterion, then

$$f_{\Delta t} = \min\left[0.95 \times \frac{criterion}{calculated\,value} , f_{input} \right] , \qquad \text{(F10.2.58)}$$

where f_{input} is an input value representing the factor that is multiplied by the old time-step size to obtain the new time-step size. The factor 0.95 ensures the decrease is large enough since the maximum temperature change is not linear with the changing time-step size. The new time-step size is determined as

$$\Delta t_{new} = \max(\Delta t_{min}, f_{\Delta t} \times \Delta t_{old}) , \qquad \text{(F10.2.59)}$$

where Δt_{min} is an input value representing the smallest time-step size allowed and Δt_{old} is the current time-step size. If one of the calculated values is less than its respective criterion, then

$$f_{\Delta t} = \min\left[0.95 \times \frac{criterion}{calculated\, value} , 1.0 \right] , \qquad \text{(F10.2.60)}$$

where the factor 0.95 ensures the increase is not too large since the maximum temperature change is not a linear function of the time-step size. Then

$$f_{\Delta t} = \min(f_{\Delta t}, f_{input}, 2.0) , \qquad \text{(F10.2.61)}$$

where f_{input} was described above. The new time-step size is then calculated as

$$\Delta t_{new} = \min\left(\Delta t_{max}, f_{\Delta t} \times \Delta t_{old}\right) , \qquad \text{(F10.2.62)}$$

where Δt_{max} is an input value representing the largest time-step size allowed. If both options are specified, the code uses the smaller of the two new time steps.

If the new time-step size is greater than or equal to the old one, the code accepts the temperature distribution it has just calculated and moves on to the next time level. If the new time-step size is smaller than the old one, the code rejects the temperature distribution it has just calculated and returns to the old time level. It then calculates a new temperature distribution at the new time level using the smaller time-step size. If the code reduces the time-step size ten times, it writes out a warning message, reduces the time-step size to the minimum value, recalculates the temperature distribution using the new time-step size and moves ahead to the next time level. If a time-step size has been reduced, the code will not allow it to be increased again until five time steps have lapsed.

When the time-step size is varied implicitly, the time-step size and the associated criteria can be greatly affected by discontinuities in the boundary conditions, time-dependent functions, temperature-dependent functions, and the initial conditions.

As the transient calculations approach a printout time, the time-step size is automatically reduced to allow the temperature distribution to be printed at the specified time. The old time-step size is saved so that calculations can resume using the old time step after the printout.

F10.2.4 TEMPERATURE-DEPENDENT THERMAL PROPERTIES

HEATING allows the thermal conductivity, k, the specific heat, c_p, and the density, ρ, to vary with temperature. A temperature-dependent density as implemented in HEATING does not satisfy conservation of mass. The code determines the conductivity of the material between two nodes, i and j, by evaluating the temperature-dependent conductivity at the average temperature of the connected nodes. This temperature is calculated as

$$\bar{T}^{(n)} = \frac{T_i^{(n-1)} + T_j^{(n-1)}}{2} \tag{F10.2.63}$$

for the calculation of the *n*th iteration for steady-state problems, as

$$\bar{T}^{n} = \frac{T_i^{n-1} + T_j^{n-1}}{2} \tag{F10.2.64}$$

for the calculation of the *n*th time step for transient problems involving one of the explicit techniques, and as

$$\bar{T}^{n+\Theta} = \frac{\hat{T}_i^{n+\Theta} + \hat{T}_j^{n+\Theta}}{2} \tag{F10.2.65}$$

for the calculation of the temperature distribution at time t_{n+1} for transient problems involving the implicit procedure. The temperatures denoted as $\hat{T}_i^{n+\Theta}$ in Eq. (F10.2.65) are evaluated according to Eq. (F10.2.54). For transient problems, the specific heat and density are determined for node *i* by evaluating the respective temperature-dependent function at T_i^n after completing the nth time step using one of the explicit techniques and at $\hat{T}_i^{n+\Theta}$ as determined by Eq. (F10.2.54) during the calculation of the temperature distribution at time t_{n+1} using the implicit procedure. In addition, the thermal conductivity of a material can be anisotropic.

F10.2.5 BOUNDARY CONDITIONS

HEATING can accommodate a variety of boundary conditions. A boundary condition is applied along a surface of a region, and heat is transferred from a surface node to a boundary node or from a node on one face of a region to a node on the opposing face of the region. Surface nodes are nodes on the face of a region that are not covered by another region containing a material. Boundary nodes are dummy nodes used to represent the temperature of the environment to which a surface is exposed. Boundary temperatures are specified as input to the code. These temperatures are only used to calculate the heat flow across a boundary surface. The boundary condition types are the following:

1. the temperature on the surface of a region can be constant or a function of time and/or position;

2. the heat flux across the surface of a region can be constant or a function of time, position, and/or surface temperature;

3. the surface heat flux for a region can be specified indirectly by defining the heat transfer mechanism to be forced convection, radiation and/or natural convection; or

4. a combination of 2 and 3.

The temperatures of nodes on surfaces whose temperatures are specified are not calculated from Eq. (F10.2.7) but are set equal to the specified value. The specified heat flux is multiplied by each node's surface area, and the result is added to the heat generation term, P_i^n, in Eq. (F10.2.7).

The boundary condition types are surface-to-environment (Type 1), specified surface temperature (Type 2), or surface-to-surface (Type 3). Boundary conditions of the surface-to-environment type are used to define heat transfer between a surface node and a boundary node. Surface-to-surface boundary conditions are used to define heat transfer between opposing surfaces. In this case, heat is transferred between a node on one surface to the corresponding node on the opposing surface. In Fig. F10.2.3, surface-to-surface boundary conditions could describe the heat transfer between nodes 1 and 2, nodes 3 and 4, and nodes 5 and 6.

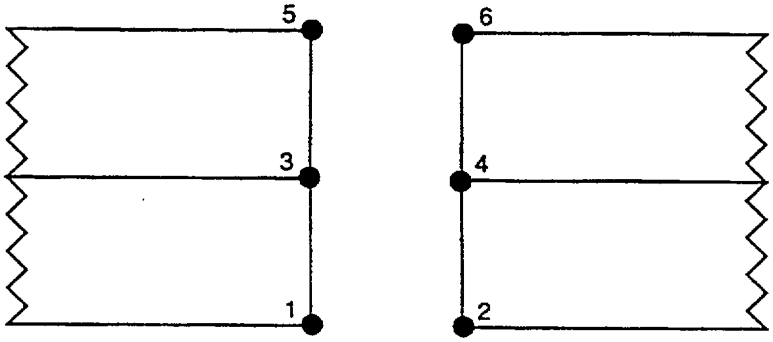


Figure F10.2.3 Surface-to-surface heat transfer

For both surface-to-environment and surface-to-surface boundary conditions, the heat flow term in Eq. (F10.2.7) is calculated as

$$\left[{}_iK_{\alpha_m}(T^n_{\alpha_m} - T_i^n)\right] = {}_iK_b(T_b^n - T_i^n) \;, \qquad \text{(F10.2.66)}$$

where ${}_iK_b$ is the effective conductance from surface node i to boundary node b or the opposing surface node b. T_b^n is either the temperature of boundary node b or the opposing surface node b at time t_n. The effective conductance is calculated as

$${}_iK_b = hA \;, \qquad \text{(F10.2.67)}$$

where h is the effective heat transfer coefficient, and A is the surface area of node i associated with the boundary condition.

The effective heat transfer coefficient is calculated as

$$h = h_c + h_r\left[(T_i^n)^2 + (T_b^n)^2\right]\left[T_i^n + T_b^n\right] + h_n\left|T_i^n - T_b^n\right|^{h_e} \;, \qquad \text{(F10.2.68)}$$

where

h_c = forced convection heat transfer coefficient,

h_r = radiative heat transfer coefficient,

h_n = coefficient for natural convection,

h_e = exponent for natural convection.

The parameters h_c, h_r, h_n, and h_e must be specified by the user and can be time- and/or temperature-dependent. When h_c, h_r, h_n, and h_e are temperature-dependent, they are evaluated at the average temperature of the opposing surface-nodes for surface-to-surface boundary conditions. For surface-to-environment boundary conditions, h_c, h_n, and h_e are evaluated at the average temperature of the related surface node and boundary node, whereas h_r is evaluated at the temperature of the surface node. For surface-to-environment boundary conditions, the boundary temperature, T_b^n, must also be supplied by the user and may be a function of time and/or position. If the temperatures are entered in either °F or °C, the code converts them to absolute degrees when calculating the effective thermal conductance due to radiation. In computing the effective conductance for a surface-to-surface boundary condition across a radial gap, the code uses the surface area at the smaller radius bounding the gap. One may simultaneously model surface-to-surface heat transfer across a region, as well as conduction through the region.

F10.2.6 CHANGE OF PHASE

Selected materials are allowed to undergo up to five phase changes during an explicit transient calculation. Change-of-phase problems cannot be solved using implicit transient solution techniques in this version of HEATING. The temperature of a node that can change phase is calculated normally until a transition temperature is reached. The progress of the phase change at the node is indicated by X_i, which is the ratio of heat that has been absorbed after the transition temperature has been reached to the total heat needed to complete the phase change for a material in node i. If a problem is restarted the initial melting ratio will be read from the restart file, otherwise the initial melting ratio is calculated as

$$X_i = \begin{cases} 0.0 \ , \ T_i^o < T_{melt} \\ 1.0 \ , \ T_i^o \geq T_{melt} \end{cases} , \qquad \text{(F10.2.69)}$$

where

T_i^o = initial temperature of node i,

T_{melt} = phase-change or transition temperature associated with node i.

If the melting ratio of a node is zero, its temperature is allowed to increase. When the temperature reaches the transition temperature the temperature of the node is held at the transition temperature, and the material is allowed to change phase in the following manner. The incremental change in melting ratio over the nth time step is calculated as

$$\Delta X_i^n = \frac{\Delta q_i^n}{\rho_{i,m}^n H_m V_{i,m}} , \tag{F10.2.70}$$

where

Δq_i^n = net heat input to node i during the nth time step,

$\rho_{i,m}^n$ = density of material m evaluated at T_i^n ,

H_m = latent heat of material m,

$V_{i,m}$ = volume of material m associated with node i.

This incremental change is added to the current value of X_i at each time step until X_i exceeds unity. Any excess heat remaining if the ratio exceeds 1.0 is used to change the temperature of the node as follows:

$$\Delta T_i^n = (X_i^n - 1.0)\frac{\rho_{i,m}^n H_m V_{i,m}}{C_i^n} , \tag{F10.2.71}$$

where C_i^n is the heat capacitance of node i during the nth time step. After this temperature adjustment, the melting ratio is set to unity, and the temperature of the node is allowed to increase normally during future time steps.

Conversely, if the melting ratio of a node is unity, its temperature is allowed to decrease until it reaches the transition temperature of the material associated with it. Then, the temperature of the node is held at the transition temperature, and the material is allowed to change phase. The incremental change in melting ratio, Eq. (F10.2.70), is added to the current value of X_i at each time step until X_i is less than zero. Any excess heat remaining after the ratio decreases below zero is used to change the temperature of the node as follows:

$$\Delta T_i^n = X_i^n \frac{\rho_{i,m}^n H_m V_{i,m}}{C_i^n} . \tag{F10.2.72}$$

Then, the melting ratio is set to zero, and the temperature of the node is allowed to decrease for future time steps.

If a node is associated with more than one material that can change phase or with a material with multiple phase changes, then each phase change is handled independently. The phase changes are ordered by increasing transition temperature. If the temperature of the node is increasing, then its temperature is not allowed to exceed the lowest transition temperature until the melting ratio increases from zero to unity. Once the melting ratio reaches unity, it is fixed there, and the temperature of the node is allowed to increase until it reaches the second transition temperature. Then, the melting ratio is reset to zero, and the temperature of the node is not allowed to increase until the melting ratio increases from zero to unity. Once the melting ratio reaches unity, the temperature of the node is allowed to increase again. This process is repeated until all phase changes associated with the node have occurred. The logic is similar when the temperature of the node is decreasing. However, the melting ratio of a node is set to zero when its temperature is between transition temperatures of the materials associated with it. In order to identify the actual phase(s) present in a nodal volume, a phase indicator value is defined as

$$Y_i = (n_i - 1) + X_i \, , \qquad \text{(F10.2.73)}$$

where X_i is the melting ratio for the phase change currently under way (or the last phase change that occurred) for node i, and n_i is the ordinal number of the phase change occurring at node i. For problems involving phase change, this value is written to the plot data file for each output time.

F10.2.7 NODE-TO-NODE CONNECTORS

HEATING allows the user to explicitly specify a thermal connection between any two nodes in the mesh or between a node and a boundary node. Any number of these connections can be specified. These connectors may be defined either in a separate file or in the CONNECTOR data block on the HEATING input file. In formulating the heat balance equation for a node having node-to-node connectors, additional heat flow terms in Eq. (F10.2.7) are calculated as

$$\left[{}_iK_{\alpha_m}(T_{\alpha_m}^n - T_i^n)\right] = {}_iK_b(T_b^n - T_i^n) \, , \qquad \text{(F10.2.74)}$$

where ${}_iK_b$ is the effective conductance from node i to node b (node b is either a node in the mesh or a boundary node). Equation (F10.2.74) is the same as Eq. (F10.2.66) for surface-to-environment or surface-to-surface boundary conditions. However, the effective conductance for node-to-node connectors is calculated as

$$\begin{aligned} {}_iK_b &= {}_iM_b \left\{ h_c + h_r\left[(T_i^n)^2 + (T_b^n)^2\right]\left[T_i^n + T_b^n\right] + h_n\left|T_i^n - T_b^n\right|^{h_e} \right\} \\ &= {}_iM_b h \, , \end{aligned} \qquad \text{(F10.2.75)}$$

where ${}_iM_b$ is a constant multiplier input by the user, and the h terms are defined the same as for a boundary condition. There is a separate ${}_iM_b$ value for every node-to-node connector. The effective heat transfer coefficient, h, is calculated from information supplied on a BOUNDARY CONDITION card in the input. A heat-flow area must be included in the constant multiplier since HEATING does not compute an area for the calculation of the effective conductance.

F10.3 SUGGESTIONS ON THE USE OF HEATING

F10.3.1 GENERAL

This section provides information to aid in the development of models. This section is not intended to stand alone, but, rather, is to be used in conjunction with Sect .
F10.4, which provides a detailed input description. The basic steps involved in developing a HEATING model are the following:

1. select a geometry type compatible with the physical problem;
2. define the model with a set of regions;
3. define all materials referenced in the region definitions;
4. define all initial temperature definitions referenced in the region definitions;
5. define all heat generation referenced in the region definitions;
6. define all boundary conditions referenced in the region definitions;
7. define a grid structure overlaying the entire model;
8. define any analytical or tabular functions used in the definition of materials, initial temperatures, heat generation, or boundary conditions;
9. if the problem involves a transient calculation, specify the times at which temperature distributions are to be stored; and
10. specify the solution type (steady-state, transient, or a sequence of steady-state and transient solutions.)

Simple problems will not require all of these steps, whereas the use of some special features (e.g., enclosure radiation modeling) will require additional steps. A set of instructions and observations are presented in this section to aid the user in developing HEATING models.

F10.3.2 GEOMETRY SELECTION

The first step in developing a model is to choose the HEATING geometry type that is the most representative of the geometry of the physical problem. The available geometries are listed in Table F10.3.1. In the HEATING model, all surfaces must coincide with a coordinate surface in the chosen geometry. This includes all surfaces at the interface between materials within the model, as well as the exterior surfaces of the model. If all surfaces in the physical geometry do not coincide with coordinate surfaces for any of the available geometries, then an approximation of the true geometry will have to be made. An example of a problem whose surfaces do not all coincide with coordinate surfaces is the axisymmetric pressure vessel shown in Fig. F10.3.1.

Table F10.3.1 Geometry types implemented in HEATING

Cylindrical		Cartesian		Spherical	
1	*r*-θ-*z*	6	*x*-*y*-*z*	10	*r*
2	*r*-θ	7	*x*-*y*	11	*r*-ϕ
3	*r*-*z*	8	*x*-*z*	12	*r*-θ-ϕ
4	*r*	9	*x*		
5	*z*				

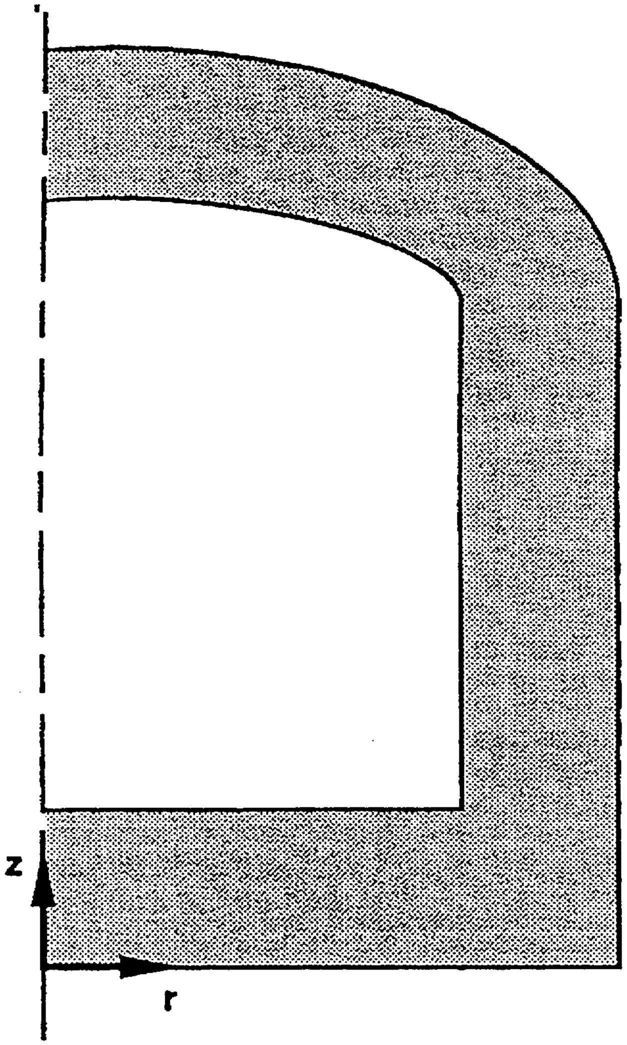


Figure F10.3.1 Physical geometry with surfaces that do not coincide with coordinate surfaces

The surfaces bounding the bottom and side of the vessel coincide with coordinate surfaces in the r-θ geometry, but the surfaces bounding the top of the vessel do not. Two possible modeling alternatives for the top are shown in Fig. F10.3.2. All model surfaces now coincide with coordinate surfaces. Whenever an approximation of the true geometry is made, it will not be possible to accurately approximate the heat capacity (model volume), surface area, and conduction paths at the same time. The user will have to determine which of these aspects to try to match. If the surface area does not match, boundary conditions definitions (e.g., forced convection heat transfer coefficient) may need to be adjusted based on the ratio of actual-to-modeled surface area. Of the two choices illustrated in Fig. F10.3.2, the one on the left is probably the better choice. The stairstep approximation on the right can be improved by using more, finer stair steps, but more regions and probably more nodes will be required to define the model.

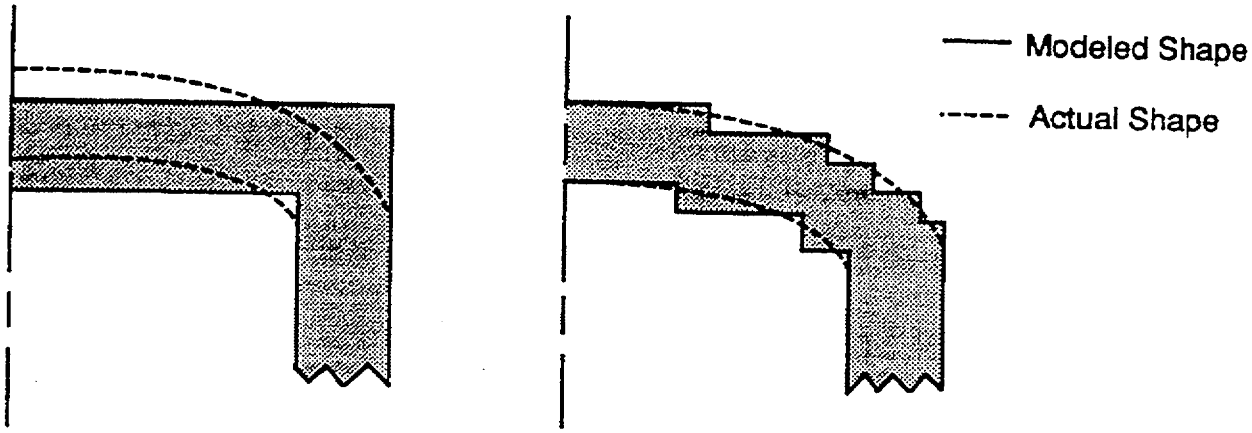


Figure F10.3.2 Alternative approximations of physical geometry

F10.3.3 REGIONS

Regions are used to describe the geometry of the model, the distribution of various materials within the model, the location of boundary conditions on the model surfaces, zones of heat generation within the model, and initial temperature distributions. Complex models are created by assembling several regions.

A region is defined by specifying lower and upper bounding coordinate surfaces along each of the coordinate axes in the model. In three dimensions, six surfaces are required to define a region; in two dimensions, four surfaces (or lines) are required; and in one dimension, two surfaces (or points) are required. All other characteristics of 1-, 2-, and 3-D regions are the same. The characteristics of a region are described below.

1. A region may contain, at most, one material. A gap region that does not contain a material may be included in the model to allow the application of a surface-to-surface boundary condition between parallel surfaces.

2. A heat generation may be specified for a region containing a material. The volumetric heat-generation rate may be time-, position-, and/or temperature-dependent within the region.

3. An initial temperature may be specified for a region. This initial temperature may be position-dependent within the region.

4. A surface-to-environment boundary condition may be defined on any surface of a region if that surface is not covered by an adjacent region containing a material. The parameters specifying the boundary condition may vary with time, position, and/or temperature.

5. A surface-to-surface-type boundary condition may be defined on the two opposing surfaces of a region to model heat transfer between these surfaces (e.g., radiation or natural convection). The parameters specifying the boundary condition may vary with time, position, or temperature.

The geometric information about a region is input in the REGION data block along with reference numbers that identify the material, initial temperature, heat generation, and boundary conditions which are subsequently defined in the MATERIAL, INITIAL TEMPERATURE, HEAT GENERATION, and BOUNDARY CONDITION data blocks, respectively.

Consider, for example, a case consisting of a simple rectangle in *x-y* geometry, half of which contains one material, and the other half, a second material, as depicted in Fig. F10.3.3. This elementary case requires two regions (as indicated), one for each material. If the upper right corner of the rectangle is omitted as in Fig. F10.3.4, three regions are required. Regions 2 and 3 of Fig. F10.3.4 contain the same material.

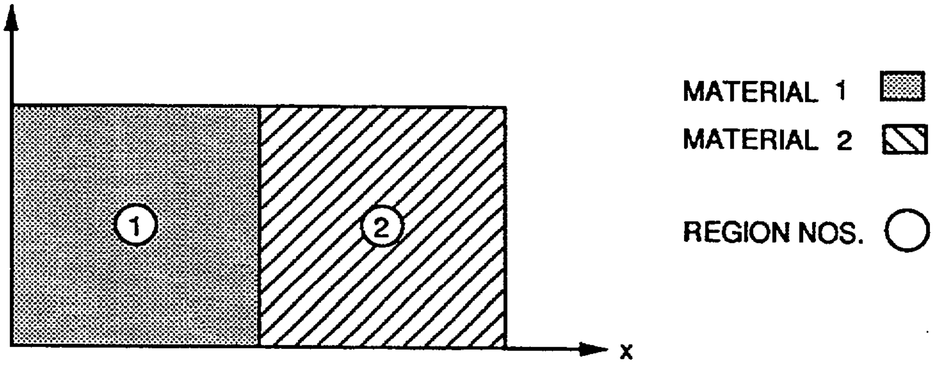


Figure F10.3.3 Regions defining a 2-D rectangular model composed of two materials

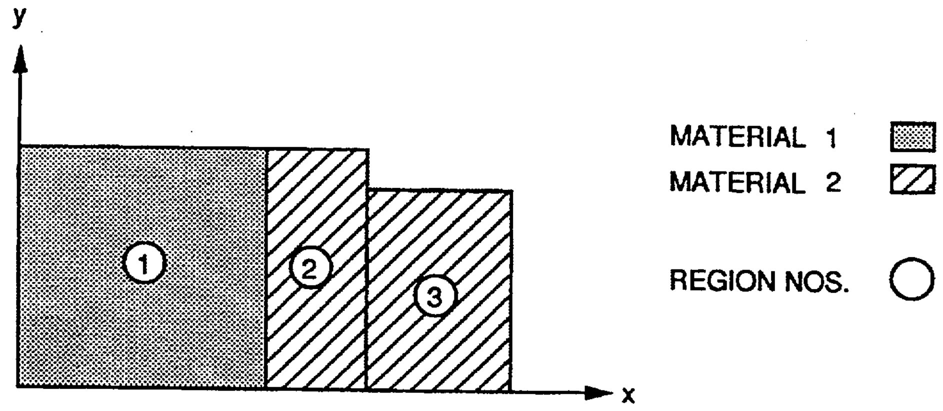


Figure F10.3.4 Regions defining a 2-D rectangular model with indentation

Introducing boundary conditions as in Fig. F10.3.5, the left boundary of the left-most rectangle now contains two different boundary types. Thus, an additional region is required to account for the different boundary conditions. Any uncovered region surface that does not have a boundary condition specified is modeled as an adiabatic (or insulated) surface.

To specify heat transfer between parallel surfaces, a region is defined with a surface-to-surface boundary condition applied to the opposing surfaces. The boundary condition describing the heat transfer process (Type 3, see Sect. F10.2.5) is applied along both of the surfaces of this region. The regions adjoining the parallel surfaces involving the surface-to-surface heat transfer must be defined and must contain a material. The region itself may or may not contain a material.

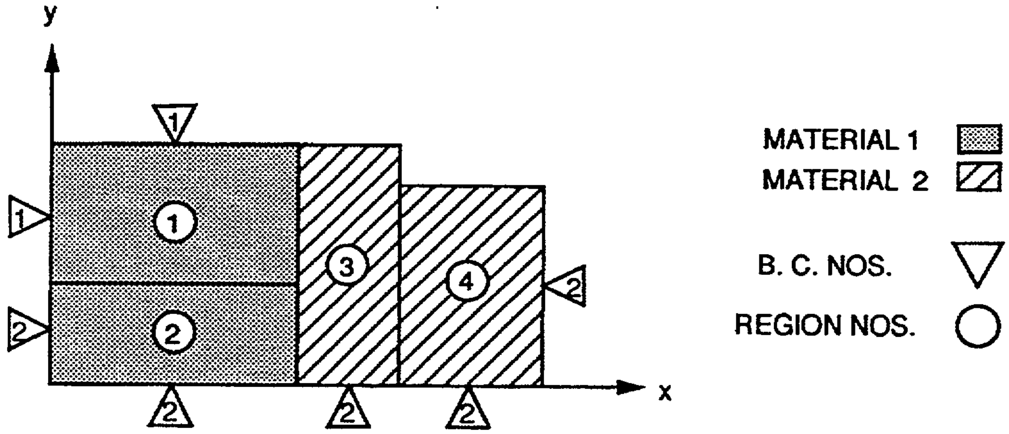


Figure F10.3.5 Regions defining a 2-D rectangular model with boundary conditions

In Fig. F10.3.6, surface-to-surface heat transfer cannot be defined between the left and right boundaries of Region 3 since part of the area adjoining the right boundary is undefined. In Fig. F10.3.7 surface-to-surface boundary conditions can be applied along the left and right sides of Region 3.

If a surface-to-surface (Type 3) boundary condition has been defined along a surface of a region and a surface-to-environment (Type 1) boundary condition is desired along the same surface, then the Type-1 boundary condition must be applied along the surface of the adjoining region. In Fig. F10.3.7, surface-to-surface boundary conditions can be applied along the left and right sides of Region 3, and a surface-to-environment boundary condition can be applied along the left side of Region 4. This boundary condition can only be applied if Region 3 is a gap region (i.e., it does not contain a material).

Typically, a user generates a model by stacking regions as if they were physical building blocks; however, it is permissible to overlap or overlay regions (i.e., two or more regions are defined such that a portion of each occupies the same space.) When overlaying occurs, the material, boundary conditions, and heat generation for the last region defined in the input are used to generate the numerical model for the overlay zone. A summary of any overlaid regions is included in the output from a run.

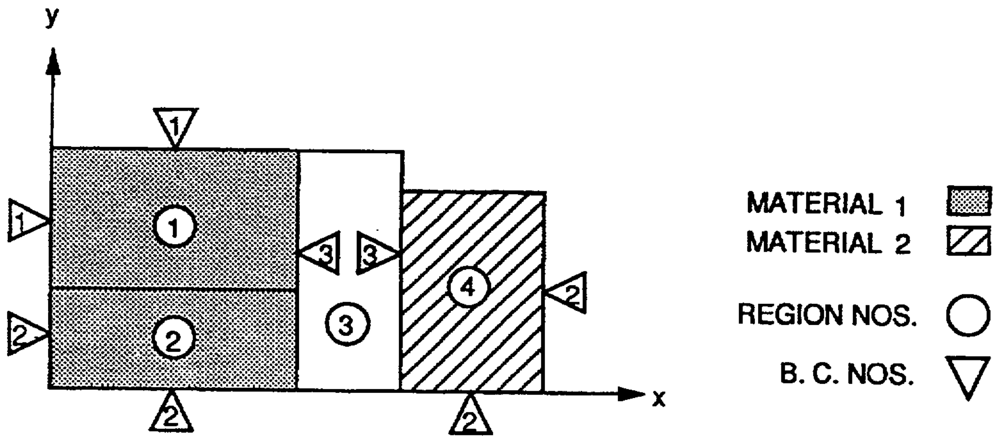


Figure F10.3.6 Regions defining a 2-D rectangular model involving surface-to-surface boundary conditions, incompatible with HEATING

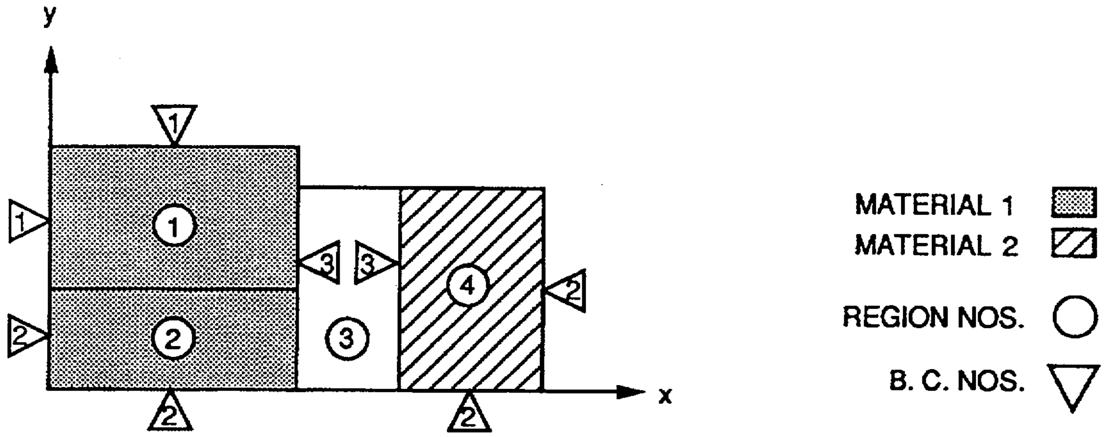


Figure F10.3.7 Regions defining a 2-D rectangular model involving surface-to-surface boundary conditions

F10.3.4 DEFINITION OF REGION ATTRIBUTES

In defining the regions in the REGION data block, reference numbers are included to identify the material, initial temperature, heat generation, and any boundary conditions pertaining to the region. The input parameters that define these quantities are supplied in additional data blocks. Supplied in the MATERIAL data block are the thermal conductivity, specific heat, and density. The initial temperature and heat generation rate are given in the INITIAL TEMPERATURE and HEAT GENERATION data blocks, respectively. The parameters that define an effective heat transfer coefficient in Eq. (F10.2.68) and any boundary heat fluxes are given in the BOUNDARY CONDITION data block.

If any of these parameters are a function of position, time, or temperature, reference numbers need to be supplied to indicate the tabular or analytical function defining this dependence. The tabular and analytical functions are supplied in the TABULAR FUNCTION and ANALYTICAL FUNCTION data blocks, respectively. If the desired dependence cannot be achieved with the built-in analytical function format, the user may supply his own as a user-supplied subroutine. Tabular functions, analytical functions, and user-supplied subroutines are discussed in subsequent sections.

The input parameters may be defined by a function having the following form:

$$P(x,y,z,t,T) \;=\; P_0 \cdot f(x,y,z,t,T) \;, \tag{F10.3.1}$$

where P_0 is a constant factor and $f(x,y,z,t,T)$ may be a product of analytical and tabular functions, such as

$$f(x,y,z,t,T) \;=\; F_i(x) \cdot F_j(y) \cdot F_k(z) \cdot F_l(t) \cdot F_m(T) \;. \tag{F10.3.2}$$

Table F10.3.2 shows the variable dependence of each parameter. If any variable is omitted from the definition of the parameter, then the corresponding factor is set equal to unity. The constant factor, P_0, is part of the input data, and its value appears on the data card that is used to define the parameter. If one of the input parameters is to be a function of position, time, or temperature, then the appropriate index or indices, i, j, k, l, or m, from Eq. (F10.3.2) are entered on a data card, too. If an index is positive, then it defines the number of the analytical function for the respective variable. If it is negative, then the absolute value of the index defines the number of the tabular function for the respective variable. If any of the defined factors for a parameter are omitted from the input data, then that particular factor is set equal to unity in Eq. (F10.3.2). If none of the factors are defined in the input, then that particular parameter is treated as a constant. If the value of P_0 is zero or is left blank on the data card and if the data indicate that the parameter is to be a function of position, time or temperature, then P_0 is set equal to unity in Eq. (F10.3.1).

Table F10.3.2 Dependence of input parameters

Input data block	Parameter	Function of					User-supplied subroutine
		x	y	z	t	T	
MATERIAL	k	(✓)	(✓)	(✓)	(✓)	✓	CONDTN
	ρ				(✓)	✓	DNSITY
	c_p				(✓)	✓	CPHEAT
INITIAL TEMPERATURE	T_o	✓	✓	✓	(✓)		INITTP
HEAT GENERATION	Q	✓	✓	✓	✓	✓	HEATGN
BOUNDARY CONDITIONS	T_b	✓	✓	✓	✓		BNDTMP
	h_c	(✓)	(✓)	(✓)	✓	✓	CONVTN
	h_r	(✓)	(✓)	(✓)	✓	✓	RADITN
	h_n	(✓)	(✓)	(✓)	✓	✓	NATCON
	h_e	(✓)	(✓)	(✓)	✓	✓	NCONEX
	$\mathrm{h_f}$	(✓)	(✓)	(✓)	✓	✓	BNFLUX

NOTE: A ✓ indicates the parameter can be a function of the indicated variable using tabular or analytical functions. A ✓ enclosed in parentheses, (✓), indicates the parameter can only be a function of the indicated variable using the related user-supplied subroutine.

When setting up a tabular or analytical function defining the temperature dependence of any of the parameters associated with a boundary condition, it is necessary to know at what temperature the function is to be evaluated. In some situations the actual nodal temperature on the surface is used, and sometimes an average temperature (average of two nodal temperatures or average of a nodal and boundary temperature) is used. Table F10.3.3 defines which temperature will be used to evaluate temperature-dependent boundary condition parameters in different situations.

Table F10.3.3 Temperature used to evaluate temperature-dependent boundary condition parameters

Parameter	Boundary condition type		
	1 Surface-to-environment	3 Surface-to-surface	1 or 3 Node-to-node
h_c	Average	Average	Average
h_r	Surface	Average	Average
h_n	Average	Average	Average
h_e	Average	Average	Average
h_f	Surface	Surface	Not allowed

NOTE: A 3 indicates the parameter can be a function of the indicated variable using tabular or analytical functions. A 3 enclosed in parentheses, (3), indicates the parameter can only be a function of the indicated variable using the related user-supplied subroutine.

If the parameter cannot be defined by a product of analytical and tabular functions as indicated in Eq. (F10.3.2), then the user may supply a subroutine to evaluate *P(x,y,z,t,T)*. Table F10.3.2 contains each input parameter and the name of the corresponding subroutine that must be supplied if the user wishes to create a function. Table F10.3.2 also includes the variables that may be used to define each parameter. For further details involving user-supplied subroutines, see Sect. F10.3.8.1.

F10.3.5 ANALYTICAL AND TABULAR FUNCTIONS

The analytical and tabular functions are built-in functions that may be used to aid in the description of the input parameters. An analytical function is defined by

$$F(v) = A_1 + A_2 v + A_3 v^2 + A_4 \cos(A_5 v) + A_6 \exp(A_7 v) + A_8 \sin(A_9 v) + A_{10} \ln(A_{11} v) \ . \qquad \text{(F10.3.3)}$$

A tabular function is defined by a set of ordered pairs, $(v_1 , G(v_1))$, $(v_2 , G(v_2))$, . . . $(v_n, G(v_n))$, where the first element is the independent variable, and the second is the corresponding function value. In order to evaluate the tabular function at some point, the program uses linear interpolation in the interval containing the point. The set of ordered pairs must have the independent variables arranged in ascending order, or

$$v_1 < v_2 < v_3 < \cdots < v_{n-1} < v_n \ . \qquad \text{(F10.3.4)}$$

If the function must be evaluated at some point outside of the defined domain of the function, then the value of the function will be

$$G(v) = \begin{cases} G(v_1) \ , \ v < v_1 \\ G(v_n) \ , \ v > v_n \end{cases} \ , \qquad \text{(F10.3.5)}$$

and a warning message is printed to inform the user.

F10.3.6 MESH DEFINITION

Once the basic model has been constructed using regions, a lattice of nodal points at which temperatures are to be calculated is defined. The location of nodal points is determined by constructing a 1-, 2-, or 3-D mesh (or grid) in the applicable coordinate system to encompass the model. In a 3-D problem, the location of a nodal point is defined by the intersection of three coordinate surfaces (one for each of the coordinate axes). In a 2-D problem the intersection of two coordinate surfaces (or lines) define a nodal location, and in a 1-D problem only one coordinate surface (or point) is required. The values of coordinate surfaces intersecting at a nodal point give the coordinates of that point. HEATING constructs the grid with an internal mesh generator.

The definition of the grid is accomplished in two steps: First, a gross grid is defined along an axis by specifying two or more coordinate surfaces in ascending order. This gross grid is then selectively refined by specifying a number of equally spaced subdivisions to be made between each pair of consecutive gross grids. Specifying one subdivision does not insert any additional coordinate surfaces in the grid between those specified in the gross grid; specifying two subdivisions places one additional coordinate surface halfway between those specified in the gross grid, etc. This process is repeated for each coordinate axis appearing in the problem. In preparing input for HEATING this information is entered in the XGRID, YGRID, and/or ZGRID data blocks.

Two basic considerations influence the selection of the coordinate surfaces defining the gross grid:

1. All coordinate surfaces that bound regions should be specified in the appropriate gross grid definitions.

2. Additional coordinate surfaces may be specified between those bounding regions to allow localized refinement of the mesh.

In deciding the number of divisions to include between gross grid lines, the following points should be considered:

1. The mesh spacing needs to be fine enough to accurately represent the spatial variation of temperature. Stated another way, this means that the mesh needs to be fine enough so that the piecewise linear curve

produced by plotting the temperatures vs nodal location along an axis matches the "true" continuous temperature distribution. It may not be possible to accurately judge this until after an initial execution. Sharp corners (large discontinuity in the slope of adjacent linear segments connecting nodal points) in a temperature-vs-distance plot that are not readily explainable in terms of the physical problem may be an indication of a grid that is too coarse.

2. Transitions from zones of fine-grid spacing to zones of coarser-grid spacing should be done gradually. Large step changes in the grid spacing can result in numerical difficulties and inaccurate solutions.

Once the grid has been established, the nodes are numbered starting with the grid location where all of the coordinates are at the minimum value occurring in the grid. If there is a material-containing region (not a void region) which includes this point, it is assigned node No. 1; otherwise, it is not assigned a node number, and the next point is checked. The order in which the grid points are checked is arrived at by varying the x (or r) coordinate most rapidly, the y (or θ) next most rapidly, and the z (or ϕ) least rapidly. Each point in the grid is checked, and those containing a material are numbered consecutively.

After the grid has been established and the nodes numbered, there is additional checking required for 2- and 3-D cylindrical and spherical problems before the solution begins. In these geometries it is possible for more than one node to occupy the same physical location in the model. Numerous nodes can share the same nodal locations whenever there is an inner radius of zero in an r-θ (cylindrical), r-θ-z (cylindrical), r-ϕ (spherical), or r-θ-ϕ (spherical) geometry. There can be two nodes at each of several nodal locations whenever the model subtends an angle of 2π in the θ direction in an r-θ (cylindrical), r-θ-z (cylindrical), or r-θ-ϕ (spherical) geometry. There can also be numerous nodes sharing nodal locations whenever a model is bounded by $\pm\pi/2$ in the ϕ direction in an r-θ-ϕ (spherical) geometry. In the output, all nodes sharing the same location have the same temperature. Information concerning shared nodal locations (if any exist) is summarized in the output and should be checked to confirm that the intended model is being used.

A model is assumed to subtend an angle of 2π in the θ direction if

$$6.2800 < \Delta\theta < 6.2836 \ , \qquad \text{(F10.3.6)}$$

where $\Delta\theta$ is the difference between the first and last θ coordinate surface defining the model. For a model in an r-θ-ϕ (spherical) geometry, the model is assumed to extend to $\pi/2$ in the ϕ direction if

$$1.5700 < \phi_{max} < 1.5709 \ , \qquad \text{(F10.3.7)}$$

where ϕ_{max} is the maximum ϕ coordinate surface defining the model. A similar check is made to see if the model extends to $-\pi/2$ in the ϕ direction.

F10.3.7 SOLUTION TECHNIQUES

The STEADY-STATE and/or TRANSIENT data blocks are used to specify a solution or sequence of solutions to be performed. The order in which these data blocks occur in the input data specifies the order in which the solution will be carried out. Each of these data blocks may occur as many times as desired by the user. The following techniques are available in HEATING: three steady-state solution techniques (SOR, direct, and conjugate gradient), two explicit transient solution techniques (classical explicit and Levy modified explicit), and two implicit transient solution techniques (Crank-Nicolson and fully implicit). HEATING should not be treated as a black box that automatically yields the correct solution. The user must decide upon an appropriate solution technique for a particular problem. Care must be exercised in correctly simulating the physical problem as well as in interpreting the results.

F10.3.7.1 Steady-State Solution Techniques

For steady-state problems, the mesh spacing can significantly affect the solution. (This situation is also true for transient problems.) Until a user has had enough experience to develop a feel for what constitutes an adequate mesh spacing for a particular problem, solutions should be obtained for several, progressively finer mesh spacings and compared. In addition, one must pay particular attention to the convergence criterion.

F10.3.7.1.1 Successive-overrelaxation technique

In the successive-overrelaxation (SOR) technique the convergence check is based on the relative temperature change between two successive iterations, not on an energy balance. It is possible to have a problem that is converging so slowly that the convergence criterion is satisfied even though the last iterate is a poor estimate of the true solution. To ensure that this is not the case, the user should check the overall energy balance on the problem by summing the energy flow rates due to boundary conditions and heat generations. This information is printed in the output at each specified printout time. The SOR solution technique can generally be sped up by providing a good estimate of the initial temperature distribution. A poor estimate of the initial temperature has been observed to prevent convergence to the true solution for some highly nonlinear problems (e.g., a two-part model coupled only by radiation connectors). For nonlinear problems, a faster solution can often be obtained by not reevaluating the temperature-dependent properties every iteration. However, for some problems, convergence problems occur if temperature-dependent properties are not evaluated every iteration. Convergence problems have also been observed in the SOR technique with problems whose temperature range spans zero (i.e., both positive and negative temperatures occur in the solution). Division by zero can occur, or erratic behavior of the extrapolation technique may hinder or even prevent convergence. This situation can be avoided by using an absolute temperature scale for these problems.

Some situations have been observed where, because of loss of significance in the calculations, the solution does not continue to move closer to the solution with additional iterations. This situation can occur when the range of conductances (*kA/l* or *Ha*) from a node to its neighbors spans "several" orders of magnitude because of a large change in mesh spacing and/or large differences in thermal conductivity of adjacent regions. Ideally the model should be redefined to minimize these difficulties, but this is not always possible. These SOR convergence problems prompted the inclusion in HEATING of the direct-solution and conjugate gradient techniques discussed below.

F10.3.7.1.2 Direct-solution technique

The direct-solution technique is implemented for 1- and 2-D problems, and, unless the bandwidth is excessively large, it is the preferred steady-state-solution technique. For a linear problem using the direct-solution technique the initial temperature distribution has no impact on the time required to obtain the solution since only one iteration is ever required. The initial temperature distribution can significantly impact the solution time for highly nonlinear problems. The choice of initial temperature can preclude obtaining a solution when a natural convection boundary condition is specified. If the initial temperature is the same as the boundary temperature, the initial conductance for heat transfer to that boundary is zero which can lead to an oscillation in the solution. Some highly nonlinear problems may tend to oscillate regardless of the initial temperature distribution chosen.

The coefficient matrix used in the steady-state direct-solution technique is a banded matrix. The matrix used by the code is dimensioned MWIDTH by MAXPTS, where MWIDTH is the maximum bandwidth and MAXPTS is the maximum number of nodes. For a 1-D problem with no surface-to-surface or node-to-node connections, the bandwidth, MWIDTH, is three. For 2-D problems with no surface-to-surface or node-to-node connections, the bandwidth is twice the number of fine lattice lines along the *x* (or *r*) axis plus one. For 1- or 2-D problems with surface-to-surface or node-to-node connections, the bandwidth will be twice the maximum difference between the node numbers of connected nodes plus one.

Since the bandwidth for a 2-D problem is a function of the number of fine lattice lines along the *x* (or *r*) axis, the user should set up the geometry of his model to minimize this number if possible. For example, if one is running a 2-D *x-y* problem with no surface-to-surface connections, it would be better to have two fine lattice lines along the *x* axis and ten fine lattice lines along the *y* axis, giving a bandwidth of 5, rather than having ten fine lattice lines along the *x* axis and two fine lattice lines along the *y* axis, giving a bandwidth of 21.

F10.3.7.1.3 Conjugate-gradient technique

A 3-D direct-solution technique is not implemented in HEATING due to the inherently large bandwidths associated with these problems. Therefore, the conjugate-gradient technique is included to give the user an alternative to the SOR technique. The conjugate-gradient technique is more tolerant of a poorly conditioned coefficient matrix than the SOR and will often converge when the SOR will not. Without roundoff the conjugate-gradient procedure converges to the "exact" solution in *n* iterations for a linear problem, where *n* is the number of unknowns. Since poorly conditioned matrices and roundoff are concerns, HEATING allows a maximum of 2*n* iterations to obtain a solution. A better estimate of the initial temperature may reduce the required number of iterations slightly, but testing has shown that the technique is fairly insensitive to the initial temperature distribution.

An outer iterative loop is used to solve nonlinear problems. The linear conjugate-gradient method is used to solve the system of equations, the coefficient matrix and right-hand-side vector are reevaluated based on the new temperature estimates, and the new linear system of equations is solved. This process is continued until the newest temperatures create an energy balance that satisfies the convergence criterion. Since a linear solution is obtained before the properties are updated, the same precautions on choosing the initial temperature distribution discussed for the direct-solution technique apply here. Similarly, the solution may tend to oscillate for highly nonlinear problems.

F10.3.7.2 Transient-Solution Techniques

As with the steady-state solutions, the mesh spacing can significantly impact the accuracy of transient solutions. Once again, the user should experiment with the mesh spacing to ensure that an acceptable solution is obtained. Using an appropriate time step in the calculations is also crucial.

F10.3.7.2.1 Explicit

With the classical explicit solution, the time step must be less than or equal to the stability criterion in order to yield a stable solution. The code calculates the stability criterion and will use it for the time step if the user does not specify that a smaller time step be used. The Levy modification of the classical explicit technique removes the restriction that the time step satisfy the stability criterion in order to produce a stable solution. This technique is stable for any size time step, but an accurate solution can generally only be obtained if the time step is less than the CEP stability criterion for a significant portion of the nodes (Levy suggests "somewhat over half"). However, for either of these techniques, particularly the Levy method, a stable solution does not necessarily mean an accurate solution.

F10.3.7.2.2 Implicit

In general, an implicit solution technique can make use of a time step that is significantly larger than the explicit stability criterion, but considerable care must still be taken in choosing the time step that produces an accurate solution. A solution produced with the Crank-Nicolson technique tends to oscillate with time - particularly when a transient has a step change in a boundary condition. Although this oscillation eventually dies out, it can have a severe impact on the solution. This oscillation can be minimized by using small time steps with the Crank-Nicolson technique early in the transient and larger time steps later, or the transient can begin with an explicit solution and later switch to the Crank-Nicolson technique. The user needs to experiment with the time step to ensure that an accurate solution is obtained.

F10.3.8 USER-SUPPLIED SUBROUTINES

F10.3.8.1 Defining Input Parameters

Subroutines may be supplied by the user to evaluate any of the parameters listed in Table F10.3.2. Thus, if an input parameter cannot be defined as the product of tabular and/or analytical functions as described in Sect. F10.3.5, the user may add his own computational technique for evaluating the parameter. All dependence on time, temperature, and/or position must be included in the user-supplied subroutine (i.e., tabular or analytical functions cannot be used in conjunction with a user-supplied subroutine to define part of the dependence). However, a user-supplied subroutine can access any of the tabular functions defined in the input. The user-supplied subroutine is referenced by specifying one of the factors of the parameter as an analytical function and by specifying no coefficients for the corresponding analytical function (i.e., leave the A2 card blank). Since this analytical function is only a flag to tell the code to call the appropriate user-supplied subroutine, the same analytical function can be specified for more than one parameter. The computational technique is then supplied in the subroutine associated with the parameter of interest (see Table F10.3.2).

HEATING contains dummy subroutines for each of the parameters listed in Table F10.3.2. If the user references one of the routines but fails to supply his own, then the code will write out an error message. User-supplied subroutines DNSITY, CPHEAT, INITTP, HEATGN, BNDTMP, and BNFLUX have the argument list shown in Table F10.3.4. Only those variables that are marked in Table F10.3.2 are initialized prior to each respective subroutine being called. All of these subroutines calculate quantities associated with a node. Subroutine BNDTMP, a typical user-supplied subroutine, is shown in Fig. F10.3.8. With the exception of HEATGN and BNFLUX, all the other user-supplied subroutines listed above are basically the same as BNDTMP, the only differences being the variables which are initialized prior to the call (see Table F10.3.2) and the parameter which is being evaluated. Subroutine HEATGN is shown in Fig. F10.3.9. For the heat generation (and for surface heat flux), the time-dependent factor is evaluated independently of the other factors.

Table F10.3.4 Argument list for DNSITY, CPHEAT, INITTP, HEATGN, BNDTMP and BNFLUX

Variable	Type	Length	Definition/Comments
RVALUE	Real	8	Returned value for parameter being evaluated
R	Real	8	Coordinates for node
TH	Real	8	
Z	Real	8	
TIM	Real	8	Time at which parameter is to be evaluated
TSN	Real	8	Temperature at which parameter is to be evaluated
VALUE	Real	8	Constant value of the parameter appearing on respective input card
NUMBER	Integer	Default	Material, heat generation, or boundary number
N	Integer	Default	Node number
ARG	Real	8	These are arrays that are not used directly by the user-supplied subroutines, but they are required to allow the subroutines to access the HEATING tabular functions.
VAL	Real	8	
NTBPRS	Integer	Default	
NTAB	Integer	Default	
HIVAL	Logical	Default	
LOVAL	Logical	Default	

```
      subroutine bndtmp(rvalue,r,th,z,tim,tsn,value,number,n,arg,val,
     . ntbprs,ntab,hival,loval)
c
      double precision rvalue, r     , th    , z     , tim   , tsn   ,
     .                 value
c
      double precision arg(1)    , val(1)
      integer          ntbprs(1), ntab(1)
      logical          loval(1) , hival(1)
c
     (Insert the algorithm to compute the boundary temperature here.  If
      more than one boundary temperature is defined by this user-supplied
      subroutine, NUMBER defines the boundary condition whose temperature
      is to be calculated for the current call.)
c
      return
      end
```

Figure F10.3.8 Sample user-supplied subroutine BMDTMP

```
      subroutine heatgn(rvalue,r,th,z,tim,tsn,value,number,n,arg,val,
     . ntbprs,ntab,hival,loval)
c
      double precision rvalue, r     , th    , z     , tim   , tsn   ,
     .                 value
c
      double precision arg(1)    , val(1)
      integer          ntbprs(1), ntab(1)
      logical          loval(1) , hival(1)
c
      if(n.eq.0) then
          (Insert algorithm to compute the time dependent portion of the
           volumetric heat generation rate here, if required.)
      else
          (Insert algorithm to compute the position and temperature dependent
           portion of the volumetric heat generation rate here, if required.)
      endif
c
      return
      end
```

Figure F10.3.9 Sample user-supplied subroutine HEATGN

User-supplied subroutines CONVTN, RADITN, NATCON, NCONEX, CONDTN have the argument list shown in Table F10.3.5. All of these subroutines calculate quantities associated with heat flow between nodes or heat flow between a node and the environment. All variables in Table F10.3.2 are initialized prior to calling each of these subroutines. The unit system in the user-supplied subroutines must be consistent with the input data unit system.

Table F10.3.5 Argument list for CONVTN, RADITN, NATCON, NCONEX and CONDTN

Variable	Type	Length	Definition/Comments
RVALUE	Real	8	Returned value for parameter being evaluated
R1	Real	8	Coordinates of first node
TH1	Real	8	
Z1	Real	8	
R2	Real	8	Coordinates of second node (all values are zero for a connection to the environment)
TH2	Real	8	
Z2	Real	8	
TIM	Real	8	Time at which parameter is to be evaluated
TSN	Real	8	Temperature at which parameter is to be evaluated
TN1	Real	8	Temperature of node 1
TN2	Real	8	Temperature of node 2
VALUE	Real	8	Constant value of the parameter appearing on respective input card
NUMBER	Integer	Default	Material, heat generation, or boundary number
N1	Integer	Default	First node number
N2	Integer	Default	Second node number (zero for connection to environment)
ARG	Real	8	These arrays are not used directly by the user-supplied subroutine, but they are required to allow the subroutines to access the HEATING tabular functions.
VAL	Real	8	
NTBPRS	Integer	Default	
NTAB	Integer	Default	
HIVAL	Logical	Default	
LOVAL	Logical	Default	

If the thermal conductivity of a material is anisotropic (i.e., directionally dependent), then it must be defined in user-supplied CONDTN (see Sect. F10.4.5 and Example Problem 4 in Appendix F10.C).

A user-supplied subroutine may access any of the tabular functions that have been entered by calling subroutine TABLE. The calling sequence is

CALL TABLE(ARG,VAL,NTBPRS,NTAB,HIVAL,LOVAL,NTABLE,XIN,YOUT) ,

where the arguments are defined in Table F10.3.6.

Table F10.3.6 Argument list for TABLE

Variable	Type	Length	Definition/Comments
ARG	Real	8	Array containing independent variables for all tables
VAL	Real	8	Array containing dependent variables for all tables
NTBPRS	Integer	Default	Number of entries in table
NTAB	Integer	Default	Translation table that relates the internal table numbers to the external (user-defined) numbers
HIVAL	Logical	Default	Array used to indicate whether table has ever been evaluated above the range of independent variables
LOVAL	Logical	Default	Array used to indicate whether table has ever been evaluated below the range of independent variables
NTABLE	Integer	Default	Table number (internal numbering system) to be evaluated
XIN	Real	8	Value of independent variable
YOUT	Real	8	Value of dependent variable returned by TABLE

The user should not change the values of any of these arguments except NTABLE, XIN, and YOUT. The value for NTABLE must be supplied in the numbering system used internal to the code, which is not necessarily the number entered in the TABULAR FUNCTIONS data block when defining the table. The first table defined is given the internal number of 1, the second defined is 2, et cetera. However, the internal numbering is less obvious when the material properties library is referenced. Each material referenced in the library can cause zero, one, or two tables to be defined. If the TABULAR FUNCTION data block precedes the MATERIAL data block in the input, the first table defined by the user will always be Table 1 in the internal numbering system.

F10.3.8.2 Monitoring Solution

Subroutine UMONTR is a user-supplied subroutine that gives the user additional capability to monitor the solution. UMONTR is called at the following times:

1. just prior to the evaluation of thermophysical parameters during initial model setup,

2. at the beginning of each steady-state or transient calculation,

3. at the completion of calculations for each time level for transient problems, and

4. at the end of each iteration for steady-state problems.

Entry USRPRT in UMONTR is called after each printout of the temperature distribution to allow the user to calculate and print additional information. All variably dimensioned arrays are available to the subroutine through the master arrays, C, IC, and LC, and the array pointers in labeled commons. The majority of pointers of interest in monitoring the solution are found in labeled common /P0034/, but on occassion the pointers in labeled commons /P0004/ and /P0234/ may be needed. Pointers to arrays containing information related to various data blocks in the input file are contained in labeled commons /MONREG/, /MONMAT/, /MONINT/, /MONHGN/, /MONBDC/, /MONGRD/, /MONSTR/, and /MONCON/.

For example, the array containing the volumetric heat capacity is SMRCV. Its pointer is ISMRCV in labeled common /P0034/ or SMRCV(1)=C(ISMRCV). Thus, the volumetric heat capacity [see Eq. (F10.2.2)] for node 25 is obtained by adding labeled common /p0034/ to subroutine UMONTR and referencing the ISMRCV plus 24th element in master array C. A dummy subroutine UMONTR is depicted in Fig. F10.3.10.

Variables such as the current time, current time step, iteration number, etc., can be made available to UMONTR by adding the appropriate labeled common blocks. After obtaining some familiarity with the code, a user can incorporate coding in subroutine UMONTR to monitor certain values at specified nodes and times or to perform additional calculations. The user should not alter the value of any variable passed from HEATING. A few of the array pointers are listed in Table F10.3.7. More pointers are not defined here because use of most other variables requires a more detailed understanding of the code. If further information is desired, please contact the developers.

F10.3.9 ENCLOSURE RADIATION MODELING

Two approaches are available in HEATING for modeling radiation in enclosures. The first, and simplest to use, is the gap radiation model accessed by specifying a surface-to-surface boundary condition across a region. With this approach, radiation is modeled as strictly 1-D (i.e., a node can exchange heat by radiation only with the node that is directly across a region from itself). This approximation is accurate only for narrow gaps. For diffuse, gray surfaces, the h_r term in Eq. (F10.2.68) is typically defined as

$$h_r = \frac{\sigma}{1/\epsilon_1 + (A_1/A_2)(1/\epsilon_2 - 1)} , \qquad \text{(F10.3.8)}$$

where σ is the Stefan-Boltzmann constant, A is an area, and ϵ is an emissivity. The subscripts distinguish between the two nodes, with 1 indicating a node on the surface having the smaller area (if the areas are not the same) and 2, a node on the surface with the larger area. This model is input by specifying the surfaces defining the gap in the REGION data block and the surface-to-surface boundary condition in the BOUNDARY CONDITION data block.

```
      subroutine umontr(c,ic,lc)
c **********************************************************************
c    Subroutine UMONTR allows the user access to any of the variably
c    dimensioned arrays at the end of each steady-state iteration or
c    transient time step.
c    Initially called from PHASE4.  Called from SSSOR, DIRSOL, SSCGRD,
c    TRANEX, and TRANIM at the beginning of each solution and at end of
c    each iteration or time step.
c    USRPRT called from WRITE4 at output times.
c **********************************************************************
      implicit double precision (a-h,o-z)
c
      dimension        c(1)
      integer          ic(1)
      logical          lc(1)
c **********************************************************************
c        these common blocks may be placed in umontr to allow user
c        access to selected arrays during the steady state or
c        transient computations.
c
      common /monreg/ noreg,matl,mats,intem,its,ngen,ngens,nregbc,
     . nregdm,iregdi
      common /monmat/ imatna,icondu,idenst,isphea,ixd,ixc,ixk,ixt,ixtp,
     . mat,ncontp,ndentp,nsphtp,munch,mcp
      common /monint/ intm,nitpos,itempi
      common /monhgn/ igen0,iqhgnu,iqhgnn,ismhgu,ismhgn,ngn,ngnfcn
      common /monbdc/ nbdtp,nbytyp,nbytfn,nbtpos,ibhflg,nbctim,nbctem,
     . ibytem,ibcdef,iqbndu,iqbndn,ismbcu,ismbcn,lpdbtp
      common /mongrd/ ir,ith,iz,irg,ithg,izg,ndrg,ndthg,ndzg
      common /monstr/ nanalt,nparm,ia,iarg,ival,ntbprs,ntab,lhival,
     . loval,iprtim,nds
      common /moncon/ nclist,nbnnc,icnlis,nltmp
      common /p0234/ npdbtb,npdbti,nnb,neqnod,nodneq,nns,ixcoor,iycoor,
     . izcoor
      common /p0034/ itdum,itdum0,itdum1,itl,ismrcv,islcap,ixl,nphsch,
     . npclst,nsv,nsvreg,isbvol,nvg,nglist,ivgvol,ivglis,nne,nelst1,
     . nelst2,ienare,ienlis,nnbupr,nbupr,ncmech,icdgeo,icdlis,nblist,
     . icdpos,nfxlst,ncdlst,nbclst,ncplst,ngnlst
      common /p0004/ iptmf,iftmf,id,iu,iaa,it2,it3,it4,idiag,ih,ismht,
     . ismvg
c
c **********************************************************************
c       these common blocks supply the number of nodes in the problem
c       and the current time (nt and tim, respectively).
c
      common /prbtyp/ nt    , ngeom , ntype , nset  , kf    , idegre ,
     .iqsum
      common /transt/ tim   , ftime , deltat, ktmfct, nstpex, shchgu ,
     .shchgn , ltrans
      logical         ltrans
c
c **********************************************************************
       (Enter Fortran for any calculations and printout to be done at the
        end of each steady-state iteration or transient time step here.)
      return
      entry usrprt(c,ic,lc)
       (Enter Fortran for any calculations and printout to be done at
        normal printout times here.)
      return
      end
```

Figure F10.3.10 Dummy user-supplied subroutine UMONTR

Table F10.3.7 Pointers for variably dimensioned arrays

HEATING array	Location in master array	Array definition
T1	C(IT1)	Temperature at nodal points
X1	C(IX1)	Phase fraction for nodal volume
SMRCV	C(ISMRCV)	Sum of $\rho \cdot c_p \cdot V$ for nodal volume
XCOORD	C(IXCOOR)	Nodal coordinate (first axis)
YCOORD	C(IYCOOR)	Nodal coordinate (second axis)
ZCOORD	C(IZCOOR)	Nodal coordinate (third axis)

heat transfer mechanism is entered in the BOUNDARY CONDITION data block. An example of radiation modeling using node-to-node connectors is included as Sample Problem 2 in Appendix F10.C.

F10.3.10 INITIAL TEMPERATURE DEFINITION

The initial temperatures of nodes are assigned at four separate stages in the input processing, with initial temperatures assigned at a later stage overriding any previously assigned value. The sequence involved in determining the initial temperature of a node is as follows:

1. All nodes have a default initial temperature of zero.

2. If a node is in a region for which an initial temperature has been specified, the initial temperature is determined from the information in the appropriate INITIAL TEMPERATURE data block in the input. If a node is partially contained in more than one region, the initial temperature is calculated as a heat-content-average (volume-average if a steady-state problem where density and specific heat are not defined) of the specified initial temperatures of the materials associated with the node.

3. If a problem is being restarted, nodal temperatures are read from the plot file created by a previous run.

4. Nodes on boundaries having a specified surface temperature are assigned the temperature given by the appropriate BOUNDARY CONDITION data in the input.

F10.3.11 MATERIAL PROPERTIES LIBRARIES

The program can extract material properties from a material properties library. Documentation for this library is included as Sect. M.5 of this document. The default units are cm-g-s-cal-°C, but HEATING has the capability to convert these to a units system consistent with the other input data (Sect. F10.4.5.4).

The material properties given in the libraries are density, thermal conductivity, specific heat, transition temperature, and latent heat. The density of the material is normally either the value at or near room temperature, or the lowest temperature for which specific heat or conductivity is tabulated, whichever is highest. A single value for the thermal conductivity is given if it is constant or if the temperature dependence is unknown. When a table of conductivity vs temperature is listed, the table will be stored and used as a tabular function. A single value for specific heat is given if it is constant or if the temperature dependence is unknown. When a table of specific heat vs temperature is listed in the library, the table will be stored and used as a tabular function. The transition temperature is the temperature at which either a phase change or a solid-state transition occurs. The latent heat is the amount of heat absorbed by the material when the temperature is increased past the transition temperature. The transition temperature and the latent heat of a material will be stored only if the ninth entry on the ML card (Sect. F10.4.5.4) is not equal to zero.

Material conductivity and specific heat temperature-dependent functions will be used whenever they are given in the material properties library. The function numbers assigned to tabular functions read from the material properties library for the thermal conductivity and the specific heat of a material are MATNO+1000 and MATNO+10000 where MATNO is the material number as read from the material properties library. (See Sect. M5 for a list of the materials and associated number.) When the material properties library is being used, tabular function numbers in the range 1001-9007 or 11001-19007 should not be assigned to the user-supplied tabular functions. A maximum of five phase changes are allowed, including phase changes in the materials from the material properties library.

Users may wish to create their own material properties library for a specific project or problem. The format is described in Sect. M.5. This user-created library can simplify model development and minimize the possibility of entering property data incorrectly. Instructions for specifying that HEATING use the user-created library rather that the default library are given in Appendix F10.A. A user-created library can be in any consistent units system, but care must be taken not to use any of the automatic unit conversion features built into HEATING if the default system is not used.

F10.4 INPUT DESCRIPTION

F10.4.1 GENERAL

The input data for a HEATING run consist of a title card, a parameter card, data blocks defining the model, data blocks defining output, and data blocks defining the solution technique. There are eleven data blocks that can be used to define the model, two data blocks to define output options, and two data blocks to define the solution sequence. A data block consists of a keyword card, followed by the data cards for that block. The keyword starts in column 1, and at least the first four characters of the keyword must be entered without any embedded blanks. The data block keywords are given in Table F10.4.1.

The data blocks that define the model may appear in any order. A case always contains a REGIONS data block, a MATERIALS data block, and one or more of the grid-definition data blocks - XGRID, YGRID, and ZGRID. The inclusion of any other data block is dependent on the particular problem being solved. Input data cards are read until either a STEADY-STATE or TRANSIENT keyword is encountered, at which point the input phase ceases, the data are processed, and the model generated. The solution sequence indicated by STEADY-STATE and/or TRANSIENT data blocks is then performed. The final data card for a case is an end-of-data indicator card that contains a percent sign (%) in column 1 and a blank in column 2.

The user-defined data identification numbers for the regions, materials, initial temperatures, heat generations, boundary conditions, analytical functions, and tabular functions may be any unique positive integer value accepted by the computer. They do not have to be numbered consecutively.

The input data are read using free-form reading subroutines that allow data to be entered in an unstructured manner. Data items are separated by one or more blanks or by a comma. If a comma is used, there can be no intervening blanks between the previous data item and the comma. All character data, such as the material names, must be delimited by a blank, not by a comma. With a few exceptions noted below, data may be entered anywhere in columns 1 through 72. Labels may be entered in columns 73 through 80 of the data cards to aid in identifying the data. If the data will not fit on one card, an "@" appears in column 1 of each continuation card. There is no limit to the number of continuation cards. A continuation card may not immediately follow a title card or a keyword card. Comment cards may appear anywhere in the input data deck following the title card. The comment cards are identified by an asterisk (*) in column 1.

Real numbers may be entered in several forms (e.g., 1.733 x 10^{-4} may be entered as 1.733-4, 1.733E-4, 1.733D-4, or 0.0001733). Multiple entries of the same data value can be achieved by entering the number of repeats, followed by an asterisk (*), followed by the data value to be repeated. For example, 5*2 enters five successive values of 2 in the input data. There must not be any blanks between the number of repeats and the repeat flag (*). Trailing zero entries at the end of a data card may be omitted.

Any consistent set of units may be used. For problems involving radiation, the default for temperature units is degrees Fahrenheit. If the temperature units are either degrees Celsius or absolute degrees, the temperature units flag must be set (see Sect. F10.4.3.1). The units associated with the algorithms that appear in user-supplied subroutines must be consistent with those of the input data.

In the sections that follow, the characters (A), (I), or (R) appearing after the heading for some of the entries indicate that the entry is alphanumeric, integer or real, respectively.

F10.4.2 CASE TITLE

This card contains a descriptive title for the problem in the first 72 columns. The card itself cannot be omitted although it may be left blank. This title serves to identify the output associated with a case.

Table F10.4.1 Data blocks

Data block keyword	Required characters	Defining
REGIONS	REGI	Model
MATERIALS	MATE	
INITIAL CONDITIONS	INIT	
HEAT GENERATION	HEAT	
BOUNDARY CONDITIONS	BOUN	
XGRID	XGRI	
YGRID	YGRI	
ZGRID	ZGRI	
CONNECTORS	CONN	
ANALYTICAL FUNCTIONS	ANAL	
TABULAR FUNCTIONS	TABU	
PRINTOUT TIMES	PRIN	Output
NODES MONITORED	NODE	
STEADY-STATE	STEA	Solution sequence
TRANSIENT	TRAN	

F10.4.3 PARAMETER CARD

F10.4.3.1 Card P

1. CPU Time for Problem (R)

   This entry is the amount of CPU time (in seconds) to be allocated for the problem to run. If the CPU time at the end of an iteration or time step is greater than or equal to this entry, the calculations are terminated.

2. Geometry Type (I)

   This entry specifies the geometry to be used for this problem. The available geometries are listed below:

Cylindrical		Cartesian		Spherical	
1	r-θ-z	6	x-y-z	10	r
2	r-θ	7	x-y	11	r-ϕ
3	r-z	8	x-z	12	r-θ-ϕ
4	r	9	x		
5	z				

3. Initial Time (R)

If this is not a restart of a previous problem, this entry indicates the initial time for transient problems and the time at which the time-dependent functions are evaluated for steady-state problems.

If this is a restart, the problem will be restarted at the last output time on the previous plot data file that is less than or equal to this entry. If this entry is zero or blank, the restart transient will resume at the time that the previous one terminated.

4. Temperature Units (I)

This entry is only needed for problems involving radiation. It indicates the units of temperature:

Entry	Temperature units
0	°F
1	°C
2	°R or K

5. Net Transient Energy Flow Calculation Flag

A nonzero value for this entry indicates that the code is to calculate and print out information on net energy flow into (or out of) the model during a transient calculation. HEATING, by default, calculates and prints the instantaneous rate of energy input to (or removal from) the model due to each boundary condition and heat generation function at normal print times. Selecting this option causes HEATING to integrate these energy rates over time to give the net energy change for the model resulting from each boundary condition and heat generation. The changes in sensible heat and latent heat for the transient are also calculated. This information is printed out at normal print times. Two columns of numbers are printed — the first column is the energy actually used in energy balances in HEATING; the second column is neglected energy (i.e., energy not used in the energy balances because it is associated with a node on a specified-surface-temperature boundary). For boundary conditions, neglected energy only occurs when a node has a specified-temperature (Type 2) boundary condition on part of its surface and a Type-1 boundary condition on another part of its surface. No heat flow is modeled from the Type-1 boundary condition to the specified-surface-temperature node. Any energy resulting from heat generation in the nodal volume associated with a specified-temperature-node is neglected. If a specified-surface-temperature boundary is time dependent, the change in sensible heat for the nodes on the surface is neglected. Net heat flow for surface-to-surface boundary conditions will always be zero.

6. Convergence Information Flag (I)

By default convergence information is not written to a file since it can be quite voluminous, particularly for implicit transient calculations. A nonzero value for this entry causes the convergence information to be written to a separate file created for this purpose.

7. Output of Selected Information (I)

The option to output selected information during the calculations exists to aid the user in the detection of input data errors and to assist in debugging and understanding cases that are not performing as anticipated. Several types of output are available using this feature: (1) a nodal description that consists of tables indicating the neighbors and regions corresponding to each node; (2) a list of nodal connectors for surface-to-surface boundary conditions; (3) thermal property tables containing the temperature, capacitance, power, and ordered pairs consisting of each neighbor and the effective conductance between the node and the neighbor for each node in the model along with information concerning the temperature dependence for the heat capacitance, effective conductance, and power associated with each node; tables containing phase change information at each node; and the temperature distribution after each iteration or time step. The thermal property tables may be output either for the initial time level only or for each time the thermal properties are reevaluated. The temperature distribution may be output at the end of each iteration in the steady-state techniques, at the end of each time step for the explicit transient methods, and at the end of each iteration in either the nonlinear or the linear loop for the implicit transient calculations. All output using this feature is generated on the standard output unit. The user should exercise caution in using this feature since it can generate excessive output. The value of the flag for each desired option is located in the following list, and the sum of these values is the integer that defines this option:

Option	Value of Flag
Nodal Description	1
Capacitance, Power, Conductance at Initial Time Level	10
Capacitance, Power, Conductance at Every Reevaluation	20
Temperature Distribution Every Steady-State Iteration	100 or 200
Temperature Distribution Every Transient Explicit Time Step	100 or 200
Temperature Distribution Every Nonlinear, Transient Implicit Iteration	100
Temperature Distribution Every Linear, Transient Implicit Iteration	200

Only one option involving output of thermal properties may be specified for a case. Also, only one option involving output of the temperature distribution may be specified for a case.

As an example, if the user wants the thermal properties output at the initial time level (value of flag is 10) and the temperature distribution output at every linear iteration (value of flag is 200), then a value of 210 (10 + 200) is entered.

F10.4.4 REGION DATA BLOCK

Each region is described by a pair of cards. At least one region must be defined.

F10.4.4.1 Keyword Card

REGIONS

F10.4.4.2 Card R1

1. Region Number (I)

   This entry contains a user-defined identification number for the region. Region numbers must be unique positive integers, but they need not be numbered consecutively.

2. Material in Region (I)

   This entry is the identification number for the material that occupies the region. An entry of zero indicates the region does not contain a material (gap region).

3. Smaller X (or R) Region Dimension (R)

4. Larger X (or R) Region Dimension (R)

5. Smaller Y (or Θ) Region Dimension (R)

6. Larger Y (or Θ) Region Dimension (R)

7. Smaller Z (or Φ) Region Dimension (R)

8. Larger Z (or Φ) Region Dimension (R)

If the problem is 1- or 2-D, then the region dimensions for the unused coordinate or coordinates should be zero. For cylindrical or spherical geometries, *r* must be greater than or equal to zero. The values for θ and ϕ must be in radians. For spherical geometries, ϕ must fall in the range $-\pi/2 \leq \phi \leq \pi/2$. All bounding coordinate surface values entered in the REGION data block must also appear as gross grid lines in appropriate XGRID, YGRID, or ZGRID data blocks.

F10.4.4.3 Card R2

1. Initial Temperature of Region (I)

   This entry is the identification number for the initial temperature to be used for the region. If this entry is zero, then the initial temperature for the region is zero. This entry should be zero for a gap region.

2. Heat Generation of Region (I)

   This entry is the identification number for the heat generation to be used for the region. If this entry is zero, then the code assumes that there is no heat generation for this region. This entry should be zero for a gap region.

3. Boundary Condition on Smaller X (or R) Region Boundary (I)

4. Boundary Condition on Larger X (or R) Region Boundary (I)

5. Boundary Condition on Smaller Y (or Θ) Region Boundary (I)

6. Boundary Condition on Larger Y (or Θ) Region Boundary (I)

7. Boundary Condition on Smaller Z (or Φ) Region Boundary (I)

8. Boundary Condition on Larger Z (or Φ) Region Boundary (I)

   Entries 3 through 8 are the identification numbers for the boundary conditions to be applied to the six surfaces of the region. A boundary condition should not be specified on a surface shared by adjacent regions unless it is a Type-3 boundary condition, or unless one of the regions is a gap region. An entry of zero indicates an insulated surface.

F10.4.5 MATERIAL DATA BLOCK (CARDS M, PC, AND ML)

One material should be defined for each unique material identification number appearing in the region definitions. An M card (and possibly a PC card) is required to describe each material when the user furnishes the material properties data. Alternatively, an ML card is required when the material data are to be read from the Lawrence Livermore National Laboratory's material properties library (Sect. M.5).

The user can specify that materials be allowed to undergo phase changes when using one of the explicit solution techniques. Phase change is not implemented for the implicit transient solution techniques. Any combination of single- and multiple-phase-change materials can be modeled as long as the total number of phase changes does not exceed five (e.g., one material can undergo up to five phase changes or five materials can each undergo a single phase change.)

Anisotropic thermal conductivity can be modeled by specifying the conductivity as temperature-dependent and the associated temperature-dependent function as user-supplied. The user supplies the anisotropic algorithm for the material in subroutine CONDTN (Sect. F10.3.8). In this subroutine the coordinates of the two nodes need to be checked to determine the direction of the conduction being calculated (see Example Problem 4 in Appendix F10.C).

F10.4.5.1 Keyword Card

MATERIALS

F10.4.5.2 Card M (User-defined Material Properties)

1. Material Number (I)

This entry contains the user-defined identification number for this material. Each material has a unique, positive identification number.

2. Material Name (A)

This entry contains the name of the material. This name, which may consist of up to eight alphanumeric characters, is used only to aid in identification. It is terminated by one or more blanks (not a comma) and, thus, cannot contain any embedded blanks.

3. Conductivity (R)

This entry contains the thermal conductivity if it is a constant. If the conductivity is time- and/or temperature-dependent, this entry contains a constant that is multiplied by the function value(s) for the corresponding time- and/or temperature-dependence.

4. Density (R)

This entry contains the density if it is a constant. If the density is time- and/or temperature-dependent, this entry contains a constant that is multiplied by the function value(s) for the corresponding time- and/or temperature-dependence. However, the user should take into consideration that mass is not conserved when a time- or temperature-dependent density is specified.

5. Specific Heat (R)

This entry contains the specific heat if it is a constant. If the specific heat is time- and/or temperature-dependent, this entry contains a constant that is multiplied by the function value(s) for the corresponding time- and/or temperature-dependence.

6. Conductivity Temperature-Dependent Function (I)

This entry identifies the analytical or tabular function defining the thermal conductivity dependence on temperature. A positive integer indicates an analytical function, and a negative integer indicates a tabular function (the table number to be used is the absolute value of this entry).

7. Density Temperature-Dependent Function (I)

This entry identifies the analytical or tabular function defining the density dependence on temperature. A positive integer indicates an analytical function, and a negative integer indicates a tabular function (the table number to be used is the absolute value of this entry).

8. Specific Heat Temperature-Dependent Function (I)

This entry identifies the analytical or tabular function defining the specific heat dependence on temperature. A positive integer indicates an analytical function, and a negative integer indicates a tabular function (the table number to be used is the absolute value of this entry).

9. Phase-Change Flag (I)

A positive entry indicates the material can undergo a change of phase. A PC card immediately follows this M card for a phase-change material. Change of phase is only implemented in the explicit solution techniques – not in the implicit solution techniques.

F10.4.5.3 Card PC (Phase Change Parameters)

This card is included only if a positive value is entered as the ninth entry on the preceding M card. If a material can undergo multiple phase changes, entries 1 and 2 are repeated as many times as necessary. There can be a maximum of five phase changes in all of the materials.

1. Phase Change or Transition Temperature (R)

2. Latent Heat or Heat of Transformation (R)

F10.4.5.4 Card ML (Material Properties Library)

1. Material Number (I)

This entry contains the user-defined identification number for this material. Each material has a unique, positive identification number.

2. Material Name (A)

This entry contains the material number from the material properties library preceded by an asterisk (*). This five-character name is used to locate the related data in the library. It is terminated by one or more blanks (not a comma) and, thus, cannot contain any embedded blanks.

3. Data Unit Conversion Control (I)

This entry provides for data conversion from the unit system of the material properties library to a unit system consistent with the rest of the input. The conversion will be performed in accordance with the following table:

Entry	Problem Units
0	cm-g-s-cal-°C (library units, no conversion)
1	m-kg-s-J-°C
2	m-kg-s-J-K
3	ft-lb-hr-Btu-°F
4	in.-lb-s-Btu-°F
5	user-defined (conversion based on entries 4 through 8)

4. Unit Conversion Factor for Density (R)

   This entry contains the conversion factor X_d for the density where the conversion will be performed as

$$\rho = X_d \cdot \rho_{library} .$$

5. Unit Conversion Factor for Specific Heat (R)

   This entry contains the conversion factor X_c for specific heat where the conversion will be performed as

$$c_p = X_c \cdot c_{p,library} .$$

6. Unit Conversion Factor for Thermal Conductivity (R)

   This entry contains the conversion factor X_k for thermal conductivity where the conversion will be performed as

$$k = X_k \cdot k_{library} .$$

7. Unit Conversion Factor for Temperature (R)

   This entry contains the conversion factor X_T for temperature where the conversion is described in 8 below.

8. Unit Conversion Factor for Temperature (R)

   This entry contains the conversion factor X_{TP} for temperature where the conversion will be performed as

$$T = X_T \cdot T_{library} + X_{TP} .$$

9. Phase Change Flag (I)

A positive value for this entry indicates the material can undergo a change of phase. If this entry is zero, the latent heat and transition temperature for the material is not read from the material properties library. If the third entry on this card is 5, then the latent heat is converted from the unit system in the library to the unit system of the problem by

$$H = X_c \cdot X_T \cdot H_{library} ,$$

where X_c and X_T are defined above.

F10.4.6 INITIAL TEMPERATURE DATA BLOCK

One initial temperature should be defined for each unique initial temperature identification number appearing in the region definitions for this problem. In some steady-state solutions, particularly nonlinear problems using the direct or conjugate gradient solutions, a poor estimate of initial conditions can prevent obtaining a solution. Therefore, an effort should be made to define a *reasonable* initial temperature.

F10.4.6.1 Keyword Card

INITIAL TEMPERATURES

F10.4.6.2 Card I

1. Initial Temperature Identification Number (I)

This entry contains the user-defined identification number for this initial temperature. Each initial temperature has a unique, positive identification number.

2. Initial Temperature (R)

This entry contains the initial temperature if it is constant. If the initial temperature is position-dependent, this entry contains a constant that is multiplied by the function value(s) for the corresponding position dependence.

3. X (or R) Position-Dependent Function Number (I)

4. Y (or Θ) Position-Dependent Function Number (I)

5. Z (or Φ) Position-Dependent Function Number (I)

Entries 3, 4, and 5 identify analytical or tabular functions, defining the x (or r), y (or θ), and z (or ϕ) dependence. If the problem is 1- or 2-D or if the initial temperature does not vary along a particular axis of the region, then the position-dependent function parameter associated with that coordinate must be zero. A positive integer indicates an analytical function, and a negative integer indicates a tabular function.

F10.4.7 HEAT GENERATION DATA BLOCK

One heat generation should be defined for each unique heat generation identification number appearing in the region definitions for this problem.

F10.4.7.1 Keyword Card

HEAT GENERATIONS

F10.4.7.2 Card G

1. Heat Generation Identification Number (I)

   This entry contains the user-defined identification number for this heat generation. Each heat generation has a unique, positive identification number.

2. Volumetric Heat Generation Rate (R)

   This entry contains the volumetric heat generation rate if it is a constant. If the heat generation is position-, time-, and/or temperature-dependent, this entry contains a constant that is multiplied by the function value(s) for the corresponding position, time, and/or temperature dependence. The units on heat generation are energy per unit volume per unit time. The heat generation rate for a region may be positive (heat source) or negative (heat sink).

3. Time-dependent Function Number (I)

4. Temperature-dependent Function Number (I)

5. X (or R) Position-dependent Function Number (I)

6. Y (or Θ) Position-dependent Function Number (I)

7. Z (or Φ) Position-dependent Function Number (I)

   Entries 3 through 7 give the analytical or tabular function identification numbers defining the time, temperature, and position dependence of the heat generation. If the heat generation rate per unit volume does not vary with time or temperature, then the time- or temperature-dependent function parameter must be zero. If the heat generation rate per unit volume does not vary along an axis or if the problem is 1- or 2-D, then the position-dependent function parameter corresponding to that coordinate must be zero.

F10.4.8 BOUNDARY CONDITION DATA BLOCK

One boundary condition should be defined for each unique boundary condition identification number appearing in the region definitions for this problem. Every boundary condition must have a B1 and B2 card. The B3 and B4 cards are optional. The B3 and B4 cards, if any, follow the B1 and B2 cards.

F10.4.8.1 Keyword Card

BOUNDARY CONDITIONS

F10.4.8.2 Card B1

1. Boundary Condition Number (I)

   This entry contains the user-defined identification number for this boundary condition. Each boundary condition has a unique, positive identification number.

2. Boundary Condition Type (I)

   This entry defines the boundary condition type according to the following table:

Entry	Boundary Condition Type
1	Surface-to-environment
2	Prescribed surface temperature
3	Surface-to-surface

3. Boundary Temperature (R)

   This entry contains the boundary temperature if it is constant. If the boundary temperature is time- and/or position-dependent, this entry contains a constant that is multiplied by the function value(s) for the corresponding position dependence. This entry is not used if the boundary condition is Type 3 or if the boundary condition is Type 2 with only a flux defined.

4. Time-Dependent Function Number (I)

   This parameter identifies an analytical or tabular function. If the boundary temperature is independent of time or if the boundary Type is 3, then this entry may be zero. A positive integer indicates an analytical function, and a negative integer indicates a tabular function.

5. X (or R) Position-Dependent Function Numbers (I)

6. Y (or Θ) Position-Dependent Function Numbers (I)

7. Z (or Φ) Position-Dependent Function Numbers (I)

Entries 5, 6, and 7 refer to analytical or tabular functions that define the position dependence along each axis. If the boundary temperature does not vary along an axis or if the problem is 1- or 2-D, then the position-dependent function parameter corresponding to that coordinate must be zero. A positive integer indicates an analytical function, and a negative integer indicates a tabular function.

F10.4.8.3 Card B2

Up to six entries may be specified on this card. The first four entries are used to determine an effective heat transfer coefficient as given by

$$h_{eff} = h_c + h_r[T_s^2 + T_b^2][T_s + T_b] + h_n|T_s - T_b|^{h_e} ,$$

where

T_s = surface temperature,
T_b = boundary temperature,
$h_?$ = heat transfer terms as defined below.

This card is left blank for a Type-2 boundary condition.

1. Forced Convection Heat Transfer Coefficient (R)

This entry contains a value for h_c, if h_c is constant. If h_c is time- and/or temperature-dependent, this entry contains a constant that is multiplied by the function value(s) for the corresponding time- and/or temperature-dependence.

2. Coefficient for Radiation (R)

This entry contains a value for h_r, if h_r is constant. If h_r is time- and/or temperature-dependent, this entry contains a constant that is multiplied by the function value(s) for the corresponding time- and/or temperature-dependence.

3. Natural Convection Multiplier Term (R)

This entry contains a value for h_n, if h_n is constant. If h_n is time- and/or temperature-dependent, this entry contains a constant that is multiplied by the function value(s) for the corresponding time- and/or temperature-dependence.

4. Natural Convection Exponent Term (R)

This entry contains a value for h_e, if h_e is constant. If h_e is time- and/or temperature-dependent, this entry contains a constant which is multiplied by the function value(s) for the corresponding time- and/or temperature-dependence.

5. Prescribed Heat Flux (R)

This entry contains a value for surface heat flux. If the heat flux is time- and/or temperature-dependent, then this value is multiplied by the function value(s) for the corresponding time- and/or temperature-dependence. For a Type-1 (surface-to-environment) boundary condition a positive heat flux indicates heat gain to the model from the surroundings, and a negative heat flux represents heat loss from the model to the surroundings. For a Type-3 (surface-to-surface) boundary condition a positive heat flux indicates heat flow from the surface with the smaller coordinate value to the opposing surface (i.e., in the positive coordinate direction), and a negative heat flux indicates heat flow in the negative coordinate direction. If the opposing surfaces do not have the same area (e.g., radial heat flow in a cylindrical or spherical problem), the surface having the smaller area is used to calculate the heat flow.

6. Time- and Temperature-Dependence Flag (I)

If any of the five preceding parameters are functions of time or temperature, then additional information must be entered on a B3 and/or B4 card. The flag for time- and temperature-dependence indicates whether or not the B3 and B4 cards are present for this particular boundary condition. Its value is determined according to the following table:

Dependence	Entry	Additional cards
None	0	None
Time	1	B3 only
Temperature	2	B4 only
Time and Temperature	3	B3 and B4

F10.4.8.4 Card B3

This card is only supplied when entry 6 on card B2 is equal to 1 or 3. Each entry identifies an analytical or tabular function that defines the time-dependent function associated with the respective parameter on Card B2. Each entry corresponds to subscript *l* in Eq. (F10.3.2). If an entry is zero, then the associated parameter is not time-dependent. A positive integer indicates an analytical function, and a negative integer indicates a tabular function.

1. Forced Convection Coefficient Time-Dependent Function Number (I)

2. Radiation Coefficient Time-Dependent Function Number (I)

3. Natural Convection Multiplier Time-Dependent Function Number (I)

4. Natural Convection Exponent Time-Dependent Function Number (I)

5. Heat Flux Time-Dependent Function Number (I)

F10.4.8.5 Card B4

This card is only supplied when entry 6 on card B2 is equal to 2 or 3. This card is just like Card B3, except each entry identifies the analytical or tabular function that defines the temperature-dependent function associated with the respective parameter on Card B2. Each entry corresponds to subscript *m* in Eq.(F10.3.2). A positive integer indicates an analytical function, and a negative integer indicates a tabular function.

1. Forced Convection Coefficient Temperature-Dependent Function Number (I)
2. Radiation Coefficient Temperature-Dependent Function Number (I)
3. Natural Convection Multiplier Temperature-Dependent Function Number (I)
4. Natural Convection Exponent Temperature-Dependent Function Number (I)
5. Heat Flux Temperature-Dependent Function Number (I)

F10.4.9 XGRID DATA BLOCK

The XGRID data block should be omitted for a geometry Type of 5. The gross grid lines along the *x* (or *r*) axis are entered on the X1 card in ascending order. All bounding *x* (or *r*) coordinate surfaces entered in the REGION data block must also appear as gross grid lines on the X1 card.

The X2 card specifies the number of equally spaced fine grid lines to be included between two adjacent gross grid lines. In particular, an entry on an X2 card specifies the number of equal increments between the gross grid line in the corresponding entry on the X1 card and the gross grid line immediately following it. Any X1 or X2 card that is a continuation card contains an "@" in column 1.

F10.4.9.1 Keyword Card

XGRID

F10.4.9.2 Card X1

The X1 cards correspond to the *x* (or *r*) coordinate.

F10.4.9.3 Card X2

The X2 cards correspond to the *x* (or *r*) coordinate. There will be one less entry on this card than there are on the X1 cards.

F10.4.10 YGRID DATA BLOCK

The YGRID data block should be omitted for geometry types of 3, 4, 5, 8, 9, 10, and 11. The gross grid lines along the y (or θ) axis are entered on the Y1 card in ascending order. All bounding y (or θ) coordinate surfaces entered in the REGION data block must also appear as gross grid lines on the Y1 card.

The Y2 card specifies the number of equally spaced fine grid lines to be included between two adjacent gross grid lines. In particular, an entry on a Y2 card specifies the number of equal increments between the gross grid line in the corresponding entry on the Y1 card and the gross grid line immediately following it. Any Y1 or Y2 card that is a continuation card contains an "@" in column 1.

F10.4.10.1 Keyword Card

YGRID

F10.4.10.2 Card Y1

The Y1 cards correspond to the y (or θ) coordinate.

F10.4.10.3 Card Y2

The Y2 cards correspond to the y (or θ) coordinate. There will be one less entry here than there are on the Y1 cards.

F10.4.11 ZGRID DATA BLOCK

The ZGRID data block should be omitted for geometry types of 2, 4, 5, 7, 9, and 10. The gross grid lines along the z (or ϕ) axis are entered on the Z1 card in ascending order. All bounding z (or ϕ) coordinate surfaces entered in the REGION data block must also appear as gross grid lines on the Z1 card.

The Z2 card specifies the number of equally spaced fine grid lines to be included between two adjacent gross grid lines. In particular, an entry on a Z2 card specifies the number of equal increments between the gross grid line in the corresponding entry on the Z1 card and the gross grid line immediately following it. Any Z1 or Z2 card that is a continuation card contains an "@" in column 1.

F10.4.11.1 Keyword Card

ZGRID

F10.4.11.2 Card Z1

The Z1 cards correspond to the z (or ϕ) coordinate.

F10.4.11.3 Card Z2

The Z2 cards correspond to the z (or ϕ) coordinate. There will be one less entry here than there are on the Z1 cards.

F10.4.12 CONNECTOR DATA BLOCK

The node-to-node connector data are entered in pairs of cards, C1 and C2. These paired cards may be repeated as many times as necessary to input all of the connector data.

F10.4.12.1 Keyword Card

<u>CONN</u>ECTORS

F10.4.12.2 Card C1

1. Base Node Number (I)

   When the energy equation is generated by HEATING for this base node, an additional term will be added for each node-to-node connector specified in the input. An additional term will also be added to the equation for a node connected to this base node if the reciprocal connections flag (entry 4) is set to 1.

2. Number of Nodes Connected to the Base Node (I)

3. Boundary Condition Number (I)

   This entry is the identification number of the boundary condition defining an effective heat transfer coefficient for the node-to-node connectors.

4. Reciprocal Connections Flag (I)

   This entry tells HEATING if the reciprocal connections of those connections defined by this C1-C2 card pair are explicitly defined elsewhere in the input. If the reciprocal connections are included in the input then a value of 0 should be entered, otherwise a 1 should be entered. As an example, if there is a connection defined from base node 57 to node 145 and elsewhere in the input there is a connection defined from base node 145 to node 57, then this entry would be 0 on the C1 cards for both base nodes.

F10.4.12.3 Card C2

The C2 card contains a set of paired entries. The number of pairs is equal to the number of connections to be added to the base node (entry 2 of card C1). The first number of each pair is an integer and gives the number of the node to be connected to the base node. A negative number entered as the node number indicates the connection is to a boundary node rather than a normal node. The temperature of the indicated boundary is used in solving the energy equation for the base node. This boundary condition does not have to be the same boundary condition entered in position three of the C1 card. The second number of each pair is the constant multiplier for the effective heat transfer coefficient given by the boundary condition used for this connection (entry 3 of card C1). As many pairs may be entered on a single card as can be fit into columns 1–72. The C2 card can be continued on additional cards as necessary by placing an "@" in column 1 of the continuation cards.

F10.4.13 ANALYTICAL FUNCTION DATA BLOCK

One analytical function should be defined for each unique analytical function identification number appearing in the definition of materials, initial temperatures, heat generations, and boundary conditions for this problem. Each analytical function is described by an A1 card and one or more A2 cards. An analytical function is defined by

$$F(v) = A_1 + A_2 v + A_3 v^2 + A_4 \cos(A_5 v) + A_6 \exp(A_7 v) + A_8 \sin(A_9 v) + A_{10} \ln(A_{11} v) .$$

F10.4.13.1 Keyword Card

ANALYTICAL FUNCTIONS

F10.4.13.2 Card A1

1. Analytical Function Number (I)

   This entry is the user-defined identification number for this analytical function. Each analytical function has a unique, positive identification number.

F10.4.13.3 Card A2

The A2 card contains a set of ordered pairs of data. If this card is left blank or if the first entry is a zero, a user-supplied subroutine will be used to evaluate the appropriate function. A continuation card must contain an "@" in column 1. Each ordered pair is defined as follows:

1. Coefficient Identification Number (I)

   This entry is an integer between 1 and 11 that identifies the coefficient whose value is the next entry on this card.

2. Coefficient Value (R)

F10.4.14 TABULAR FUNCTION DATA BLOCK

One tabular function should be defined for each unique tabular function identification number appearing in the definition of materials, initial temperatures, heat generations, and boundary conditions for this problem. Each tabular function is described by a T1 card and one or more T2 cards. A tabular function is defined by specifying a set of ordered pairs. Each ordered pair contains an independent variable value and its dependent functional value; viz., v_1, $F(v_1)$; v_2, $F(v_2)$; etc. The values of v_i are entered in ascending order. Linear interpolation is used to evaluate the function between the points. If the function is evaluated either below or above the range of independent variable values, the first or last dependent functional value, respectively, will be used. At least two data pairs are required.

F10.4.14.1 Keyword Card

TABULAR FUNCTIONS

F10.4.14.2 Card T1

1. Tabular Function Number (I)

   This entry contains the user-defined identification number for this tabular function. Each tabular function has a unique, positive identification number.

F10.4.14.3 Card T2

The T2 card contains the ordered pairs defining the tabular function. A continuation card must contain an "@" in column 1. Each ordered pair is defined as follows:

1. Independent Variable (R)

2. Dependent Variable (R)

F10.4.15 PRINTOUT TIMES DATA BLOCK

This data block defines the times at which a solution summary is written to the print file and the temperature and phase distributions are written to the plot data file. This data block is not used for steady-state problems.

F10.4.15.1 Keyword Card

PRINTOUT TIMES

F10.4.15.2 Card O

This card contains the times in ascending order at which the standard output is performed. Each entry is a floating-point number. If the data will not fit on one card, continuation cards must contain an "@" in column 1.

F10.4.16 NODES MONITORED DATA BLOCK

This data block allows the user to monitor the temperature of a few nodes throughout a steady-state or transient calculation. A table of nodal temperatures is written to a separate print file at the requested frequency. This file is written in an ASCII format that is compatible with the TECPLOT plotting program so that temperature-vs-time plots can be generated quickly. (Before running TECPLOT, the TECPLOT ASCII file needs to converted to a TECPLOT binary file by running PREPLOT.)

F10.4.16.1 Keyword Card

NODES MONITORED

F10.4.16.2 Card S

1. Frequency for Special Monitoring of Temperatures (I)

   This entry specifies the number of steady-state iterations or transient time steps between printouts of the temperature of the nodes specified on the remainder of this card.

2. Node(s) to be Monitored (I)

   Nodes whose temperatures will be printed as a function of the number of iterations for steady-state calculations or the number of time steps for transient calculations. Any continuation card must contain an "@" in column 1.

F10.4.17 STEADY-STATE DATA BLOCK

This data block is included whenever a steady-state solution is desired.

F10.4.17.1 Keyword Card

STEADY-STATE

F10.4.17.2 Card SS

1. Steady-State Solution Technique (I)

   This entry specifies the solution technique to be used for this steady-state calculation according to the following table.

Entry	Solution Technique
1	SOR
2	Direct-solution (1- or 2-D problems only)
3	Conjugate gradient

2. Maximum Number of Steady-State Iterations Allowed (I)

This entry defines the maximum number of iterations allowed before the steady-state calculations are terminated. For linear problems, the direct-solution technique requires only one iteration to obtain the solution. If this entry is zero, the code will use a default of 500 iterations for the SOR technique or 20 iterations for the direct technique. For the conjugate gradient technique, this entry is the number of iterations in the nonlinear (outer) loop. The default value is 20. For the linear (inner) loop the maximum number of iterations is twice the number of nodes in the problem.

3. Steady-State Convergence Criterion (R)

For the SOR technique, this entry contains the value of ϵ in Eq. (F10.2.14) which defines the steady-state convergence criterion. For the direct-solution technique, this entry contains the convergence criterion, as discussed in Sect. F10.2.2.2. The steady-state calculation will continue until the convergence criterion is met. Since the criterion that ensures convergence varies from case to case, the user must rely on judgment and experience in determining the correct value for his particular problem. If this entry is zero, the code uses a default of 10^{-5}.

4. Steady-State Overrelaxation Factor (R)

This entry is the value of β [see Eq. (F10.2.13)]. Its value must be in the range $1 \leq \beta < 2$. If a zero is entered for steady-state problems using the SOR solution technique, the code uses a default of 1.9.

5. Updating Temperature-Dependent Properties (I)

This entry specifies the number of iterations that are allowed before the temperature-dependent thermal properties are reevaluated for steady-state problems using the SOR solution technique. Once the convergence criterion has been satisfied, this value is set to one. The calculations are continued until the convergence criterion is satisfied a second time. Most nonlinear problems will require less cpu time to reach convergence if the thermal properties are not evaluated at each iteration. The default value is 1.

6. Time (R)

This entry allows the problem time associated with this steady-state calculation to be reset. If this entry is blank or zero, the time will be the value entered on the parameter card if this steady state is the first solution requested in the solution sequence or the final time for a previous transient if this steady-state solution follows a transient in the solution sequence. All time-dependent fuctions will be evaluated at this time. Even if there are no time-dependent functions, it may be advantageous to reset the time for steady-state solutions that follow a transient solution if temperature-vs-time plots are desired.

F10.4.18 TRANSIENT DATA BLOCK

This data block is included whenever a transient solution is desired.

F10.4.18.1 Keyword Card

TRANSIENT

F10.4.18.2 Card TR

1. Transient Solution Technique (I)

   This entry indicates the solution technique that is to be used for this transient calculation.

Entry	Technique	Additional Cards Required
1	Explicit	TR1
2	Implicit	TR2 and one or more TR3

2. Final Time (R)

   This entry indicates the final time for the transient.

F10.4.18.3 Card TR1

This card must be included whenever an explicit transient solution is desired (i.e., when entry 1 on the TR card is 1.)

1. Initial Time Step (R)

   This entry contains the initial time step for the explicit transient calculation. If this value is less than or equal to the stability criterion Eq. (F10.2.35), the input value is used. If the input value is larger than the stability criterion (but not more than a factor of 10 larger), the time step is set equal to the stability criterion. If the input value is more than a factor of 10 larger than the stability criterion, the case is terminated. If a value of zero is entered, the initial time step is set equal to the stability criterion.

2. Levy Technique Option (I)

   This entry is the factor by which the stable time increment is multiplied to form the time increment for Levy's explicit method. If the entry is zero or 1, Levy's method will not be used. If Levy's explicit method is specified (greater than 1), the initial time step in the previous entry will not be used. The time increment is initially set to the stability criterion.

3. Updating Temperature-Dependent Properties (I)

   This entry is the number of time steps between reevaluation of temperature-dependent properties. If the Levy technique is to be used or if this entry is zero or blank, the properties will be reevaluated every time step. The maximum value permitted for this entry is 10.

F10.4.18.4 Card TR2

This card must be included whenever an implicit transient solution is desired (i.e., when entry 1 on the TR card is 2.) This entire card may be left blank or any of its entries may be zero, and the default values will be used. Default values are based on previous experience with the code but are not necessarily the best values for a given problem. They are probably good starting points.

1. Implicit Time-Differencing Scheme (R)

   This entry defines the implicit technique that will be used to solve the transient problem. It refers to Θ in Eq. (F10.2.38) and must in the range $0.5 \leq \Theta \leq 1.0$. The default is 0.5.

2. Convergence Criterion for Solution of Linear Equations (R)

   This is the convergence criterion that must be met in order for the iterative technique to terminate successfully at each time step. This convergence criterion corresponds to ϵ_1 in Eq. (F10.2.52). The default is 10^{-5}.

3. Number of Iterations Between Convergence Checks for Linear Loop (I)

   This entry indicates the number of iterations between tests for convergence in the linear loop. Experience has shown that for problems requiring large numbers of iterations (over 50) for the linear loop to converge, a significant savings in computer time can be achieved by not performing the calculations necessary for convergence tests at every iteration. However, the number entered applies for all of the transient calculations. The default value is 1.

4. Convergence Criterion for Problems Involving Temperature-Dependent Parameters (R)

   This convergence criterion applies to the average relative change in temperature over all nodes between two consecutive nonlinear iterations. It corresponds to ϵ_2 in Eq. (F10.2.55). The default is 10^{-5}.

5. Additional Convergence Criterion for Problems Involving Temperature-Dependent Parameters (R)

   This convergence criterion applies to the maximum absolute change in temperature at any node between two nonlinear iterations. The default is zero, so this criterion is not used unless explicitly specified by the user. This option is useful for problems that are only slightly nonlinear. By entering an appropriate value for this entry the nonlinear iteration within a time step can be eliminated, and temperature-dependent properties will be reevaluated only once per time step.

6. SOR Acceleration Parameter Initial Value (R)

   This defines the initial value of the point-successive-overrelaxation iteration acceleration parameter [ω in Eq. (F10.2.57)]. It also defines the method that will be used to update the acceleration parameter. If this entry is positive, then the acceleration parameter will remain constant throughout the calculations and will be equal to the value of this entry. If it is zero, then the acceleration parameter will be optimized empirically as a function of time. This appears to be the best option for nonlinear problems. If it is negative, then the acceleration parameter will be calculated using Carre's technique.[18] The absolute value of this entry must be less than 2.0.

7. Time Steps Between Acceleration Parameter Optimization (I)

This entry defines the number of time steps between attempts to optimize the acceleration parameter empirically [referred to as N_ω in Sect. F10.2.3.3.4]. It is used only when entry 6 is zero. The default value is 1.

8. Number-of-Iterations Criteria for Acceleration Parameter Updates (I)

For the case when the acceleration parameter will be updated empirically (entry 6 is zero), then this entry, the seventh, defines the change-in-number-of-iterations criterion [referred to as I_ω in Sect. F10.2.3.3.4] which must be met before the acceleration parameter will be updated. The default is 5. For the case when the SOR acceleration parameter will be updated using Carre's technique, this entry defines the number of iterations between updates. The default is 12.

9. Change-in-Number-of-Iterations Criteria (I)

The change-in-number-of-iterations criterion [referred to as J_ω in Sect. F10.2.3.3.4] which is used to determine when a good estimate to the optimum acceleration parameter has been found. This entry is used only when the acceleration parameter will be updated empirically (entry 6 on this card is zero). The default is 2.

F10.4.18.5 Card TR3

This card must be included whenever an implicit transient solution is desired (i.e., when entry 1 on the TR card is 2.) When an implicit scheme is used to solve a transient problem, the time step may be variable. This condition allows the time step to increase as the solution smooths out and to decrease when some parameter varies rapidly with time. The information controlling the value of the time step is contained on one or more TR3 cards. The size of the time step is automatically adjusted in order to get printouts of the temperature distribution at the specified time. If the size of the coefficients in the system of equations varies by orders of magnitude (10^5 or greater), it has been observed that point-successive-overrelaxation iteration may converge very slowly (it may not converge at all). This situation occurs when the grid spacing or thermal properties vary by orders of magnitude over the problem. It can be observed by examining the stability criterion table in the output. If this appears to be happening, either further subdivide some of the larger nodes or combine some of the smaller ones. In some cases, it may help to use a larger time step.

1. Initial Time Step (R)

If this entry is zero for the initial TR3 card, the initial time step will be equal to the stability criterion for the Classical Explicit Procedure. If this entry is zero for any TR3 card after the first one, the time step will be equal to the last one used subject to any constraints following on this card.

2. Time Step Multiplication Factor (R)

After the temperature distribution is calculated at a time level, the current time step is multiplied by a factor to determine a new time step. The default value is 1.0. For many problems whose parameters vary mildly with time and/or temperature, values between 1.0 and 1.1 have been acceptable.

3. Maximum Time Step (R)

Once the time step reaches this value, it is no longer increased. The default is 10^{50}.

4. Time at which New TR3 Card is to be Read (R)

This entry contains the maximum time at which the time step information on this card applies. When the problem time reaches this value, a new TR3 card is read. The default is 10^{50}.

5. Maximum Temperature Change per Time Step (R)

This entry contains the maximum absolute temperature change allowed at a node from one time level to the next. The time step is adjusted to try to obtain this maximum temperature change per time step. This adjustment may not be achieved due to other constraints placed on the time step. If this entry is zero, then this feature is not invoked in calculating the time step.

6. Maximum Percentage Temperature Change per Time Step (R)

This entry contains the maximum percentage temperature change allowed at a node from one time level to the next. The time step is adjusted to try to obtain this maximum percent temperature change for each time step. This adjustment may not be achieved because of other constraints placed on the time step. If this entry is zero, then this feature is not invoked in calculating the time step.

7. Minimum Time Step (R)

This entry contains the minimum value of the time step. Once the time-step size reaches this value, it is no longer decreased. The default is one-tenth of the initial time-step size.

F10.4.19 DATA-TERMINATION CARD

Each problem must be terminated with a data-termination card. This card consists of a percent sign (%) in column 1, followed by a blank in column 2.

F10.4.20 INPUT SUMMARY

Table F10.4.2 contains a summary of the information required for all the data blocks. Each input parameter is contained within a box in the table. Since HEATING uses free-form input, the parameter boxes do not correspond to any particular columns in the input file. However, the last column of boxes (columns 73 through 80 of each card) are reserved for a card identification and are not read by HEATING. This last column contains an optional card identification and, in parentheses, the section in the report where additional information can be found. The first line in the parameter input boxes contains the variable name that is used in the program and, in parentheses, the type of that variable. The remainder of the box contains a short explanation of the input parameter. Where space permits, additional notes have been included to further describe the input.

Table F10.4.2 Summary of input data for HEATING 7.2

JOBDES Job Description - Up to 72 Alphanumeric Characters									**Title Card** **(4.2)**
MXCPU (I) CPU time in seconds for problem execution	NGEOM (I) Geometry Type Cylindrical Cartesian 1 r-θ-z 6 x-y-z 2 r-θ 7 x-y 3 r-z 8 x-z 4 r 9 x 5 z Spherical 10 r 11 rφ 12 rθ-φ	TIM (I) Initial time (See Sect. F10.4.3.1 for more information)	IDEGRE (I) Temperature units (Only used for radiation) 0 = F 1 = C 2 = Absolute	IQSUM (I) Flag to calculate net energy changes for transient 0 = do not calculate 1 = calculate	JCNVRG (I) Flag to output convergence info during calculations 0 = do not output 1 = output	IMONTR (I) Flag to output selected information during calculations (See table in Sect. F10.4.3.1 for values)			**P** **(F10.4.3.1)**

REGIONS

NOREG(N) (I) Region number	MATS(N) (I) Region material number (Card M or ML)	REGDIM(1,N) (R) Smaller x or r region dimension	REGDIM(2,N) (R) Larger x or r region dimension	REGDIM(3,N) (R) Smaller y or θ region dimension	REGDIM(4,N) (R) Larger y or θ region dimension	REGDIM(5,N) (R) Smaller z or φ region dimension	REGDIM(6,N) (R) Larger z or φ region dimension		**R1** **(F10.4.4.2)**
ITS(N) (I) Region initial temperature number (Card I)	NGENS(N) (I) Region heat generation number (Card G)	NREGBC(1,N) (I) Boundary condition on smaller x or r (Card B1,...)	NREGBC(2,N) (I) Boundary condition on larger x or r (Card B1,...)	NREGBC(3,N) (I) Boundary condition on smaller y or θ (Card B1,...)	NREGBC(4,N) (I) Boundary condition on larger y or θ (Card B1,...)	NREGBC(5,N) (I) Boundary condition on smaller z or φ (Card B1,...)	NREGBC(6,N) (I) Boundary condition on larger z or φ (Card B1,...)		**R2** **(F10.4.4.3)**

MATERIALS

MAT(N) (I) Material number	MATNAM(N) (A) Material name (Maximum of 8 characters)	CONDUC(N) (R) Material conductivity	DENSTY(N) (R) Material density. Can be zero for steady-state only	SPHEAT(N) (R) Material specific heat. Can be zero for steady-state only	NCONTP(N) (I) Conductivity temperature-dependent function number	NDENTP(N) (I) Density temperature-dependent function number	NSPHTP(N) (I) Specific heat temperature-dependent function number	MCP(N) (I) Phase change flag 0 = no phase change 1 = phase change (PC card must be supplied)	**M** **(F10.4.5.2)**
					NOTE: For the above entries, a positive integer references an analytical function and a negative integer a tabular function				
SLTM (R) First phase-change or transition temperature	SLHM (R) Latent heat for first phase change	SLTM (R) Second phase-change or transition temperature	SLHM (R) Latent heat for second phase change	. . .	. . .		NOTE: Change-of-phase modeling is implemented in the explicit solution techniques only		**PC** **(F10.4.5.3)**
MAT(N) (I) Material number	MATNAM(N) (A) Material name. Library material number preceded by an asterisk (*)	MUNCH(N) (I) Data units conversion flag	XD(N) (R) Unit conversion factor for density	XC(N) (R) Unit conversion factor for specific heat	XK(N) (R) Unit conversion factor for conductivity	XT(N) (R) Unit conversion factor (multiplier) for temperature	XTP(N) (R) Unit conversion factor (additive) for temperature	MCP(N) (I) Phase change flag 0 = no phase change 1 = phase change if included in library data	**ML** **(F10.4.5.4)**
			NOTE: The above entries are only used for units conversion if entry 3 on this card has a value of 5.						

Table F10.4.2 (continued)

INITIAL TEMPERATURES

INT(N) (I) Initial temperature number	TEMPIN(N) (R) Initial temperature	NITPOS(1,N) (I) x- or r-dependent function number	NITPOS(2,N) (I) y- or θ-dependent function number	NITPOS(3,N) (I) z- or ϕ-dependent function number					I (F10.4.6.2)
		NOTE: For the above entries, a positive integer references an analytical function and a negative integer a tabular function.							

HEAT GENERATIONS

NGN(N) (I) Heat generation number	GEN0(N) (R) Volumetric heat generation rate	NGNFCN(1,N) (I) Time-dependent function number	NGNFCN(2,N) (I) Temperature-dependent function number	NGNFCN(3,N) (I) x- or r-dependent function number	NGNFCN(4,N) (I) y- or θ-dependent function number	NGNFCN(5,N) (I) z- or ϕ-dependent function number			G (F10.4.7.2)
		NOTE: For the above entries, a positive integer references an analytical function and a negative integer a tabular function.							

BOUNDARY CONDITIONS

NBDTP(N) (I) Boundary condition number	NBYTYP(N) (I) Boundary condition type 1 = surface-to-boundary 2 = prescribed surface temperature 3 = surface-to-surface	BYTEMP(N) (R) Boundary temperature (Not used for NBYTYP=3)	NBTFN(N) (I) Time-dependent function number	NBTPOS(1,N) (I) x- or r-dependent function number	NBTPOS(2,N) (I) y- or θ-dependent function number	NBTPOS(3,N) (I) z- or ϕ-dependent function number			B1 (F10.4.8.2)
			NOTE: For the above entries, a positive integer references an analytical function and a negative integer a tabular function. These entries are not used for surface-to-surface boundary conditions (NTYPE=3).						
BCDEF(1,N) (R) Forced convection heat transfer coefficient, h_c	BCDEF(2,N) (R) Radiation coefficient, h_r	BCDEF(3,N) (R) Natural convection multiplier term, h_n	BCDEF(4,N) (R) Natural convection exponent term, h_e	BCDEF(5,N) (R) Prescribed heat flux, h_f. (Positive for heat addition to the model)	IBHFLF (I) Parameter flag 0=no additional cards 1=B3 card only 2=B4 card only 3=B3 and B4 cards				B2 (F10.4.8.3)
NBCTIM(1,N) (I) Forced convection time-dependent function	NBCTIM(2,N) (I) Radiation time-dependent function	NBCTIM(3,N) (I) Natural convection multiplier time-dependent function	NBCTIM(4,N) (I) Natural convection exponent time-dependent function	NBCTIM(5,N) (I) Heat flux time-dependent function	NOTE: For the entries on this card, a positive integer references an analytical function and a negative integer a tabular function.				B3 (F10.4.8.4)
NBCTEM(1,N) (I) Forced convection temperature-dependent function	NBCTEM(2,N) (I) Radiation temperature-dependent function	NBCTEM(3,N) (I) Natural convection multiplier temperature-dependent function	NBCTEM(4,N) (I) Natural convection exponent temperature-dependent function	NBCTEM(5,N) (I) Heat flux temperature-dependent function	NOTE: For the entries on this card, a positive integer references an analytical function and a negative integer a tabular function.				B4 (F10.4.8.5)

Table F10.4.2 (continued)

XGRID									
RG(1) (R) Smallest x or r gross grid line	RG(2) (R) Next larger x or r gross grid line	RG(3) (R) Next larger x or r gross grid line	. . .	RG(N) (R) Largest x or r gross grid line					**X1** (F10.4.9.2)
NDRG(1) (I) Number of divisions between first and second gross grid lines	NDRG(2) (I) Number of divisions between second and third gross grid lines	. . .	NDRG(N-1) (I) Number of divisions between gross grid lines n-1 and n						**X2** (F10.4.9.3)

YGRID									
THG(1) (R) Smallest y or θ gross grid line	THG(2) (R) Next larger y or θ gross grid line	THG(3) (R) Next larger y or θ gross grid line	. . .	THG(N) (R) Largest y or θ gross grid line					**Y1** (F10.4.10.2)
NDTHG(1) (I) Number of divisions between first and second gross grid lines	NDTHG(2) (I) Number of divisions between second and third gross grid lines	. . .	NDTHG(N-1) (I) Number of divisions between gross grid lines n-1 and n						**Y2** (F10.4.10.3)

ZGRID									
ZG(1) (R) Smallest z or ϕ gross grid line	ZG(2) (R) Next larger z or ϕ gross grid line	ZG(3) (R) Next larger z or ϕ gross grid line	. . .	ZG(N) (R) Largest z or ϕ gross grid line					**Z1** (F10.4.11.2)
NDZG(1) (I) Number of divisions between first and second gross grid lines	NDZG(2) (I) Number of divisions between second and third gross grid lines	. . .	NDZG(N-1) (I) Number of divisions between gross grid lines n-1 and n						**Z2** (F10.4.11.3)

CONNECTORS									
N (I) Base node number to which connections are to be made	NNC (I) Number of nodes connected to base node	NBNNC (I) Boundary condition number defining heat transfer mechanism	IRCPRO (I) Reciprocal connections are entered. 0 = yes 1 = no						**C1** (F10.4.12.2)
NCLIST (I) First node connected to base node	CNLIST (R) Constant multiplier portion of connectivity	NCLIST (I) Second node connected to base node	CNLIST (R) Constant multiplier portion of connectivity	. . .	. . .	NOTE: Enter total of NNC pairs.			**C2** (F10.4.12.3)

Table F10.4.2 (continued)

ANALYTICAL FUNCTIONS

NANALT (I) Analytical function number					NOTE: Entries on A2 Card define an analytical function of the following form. Only those terms having non-zero values need to be entered. $F(V) = A_1 + A_2V + A_3V^2 + A_4cos(A_5V) + A_6exp(A_7V) + A_8sin(A_9V) + A_{10}ln(A_{11}V)$				A1 (F10.4.13.2)
NPRM (I) Coefficient index, I	A(NPRM) (R) Coefficient value, A_I	NPRM (I) Coefficient index, I	A(NPRM) (R) Coefficient value, A_I	. . .	. . .	NOTE: To indicate a function is defined in a user-supplied subroutine, leave Card A2 blank.			A2 (F10.4.13.3)

TABULAR FUNCTIONS

NTABL (I) Tabular function number									T1 (F10.4.14.2)
ARG(1) (R) First independent value	VAL(1) (R) First dependent value	ARG(2) (R) Second independent value	VAL(2) (R) Second dependent value	. . .	. . .				T2 (F10.4.14.3)

PRINTOUT TIMES

PRTIME(1) (R) First printout time	PRTIME(2) (R) Second printout time	PRTIME(3) (R) Third printout time	. . .						O (F10.4.15.2)

NODES MONITORED

NTS (I) Number of iterations or time steps between output	NDS(1) (I) First node monitored	NDS(2) (I) Second node monitored	. . .						S (F10.4.16.2)

STEADY-STATE

NTYPE (I) Solution technique 1 = SOR 2 = Direct 3 = Conjugate gradient	NOITX (I) Maximum number of steady-state iterations (Defaults: 500 for SOR, 20 nonlinear iterations for direct and conjugate gradient)	EPI (R) Steady-state convergence criterion (Default: 10^5)	BETA (R) SOR overrelaxation factor $1.0 \leq \beta < 2.0$ (Default: 1.9)	MCOUNT (I) Number of iterations between evaluation of temperature-dependent thermal properties for SOR (Default: 1) Maximum = 10	TIM (R) Value to reset problem time to. (Normally used only following a transient solution)				SS (F10.4.17.2)

Table F10.4.2 (continued)

TRANSIENT										
NTYPE (I) Solution Technique 1 = Explicit 2 = Implicit	FTIME (R) Final time			NOTE: For NTYPE = 1, supply TR1 Card. For NTYPE = 2, supply TR2 and one or more TR3 Cards.						**TR** (F10.4.18.2)
DELTAT (R) Time step	KTMFCT (I) Factor by which stable time step is increased with Levy's method	NSTPEX (I) Number of time steps between evaluation of temperature-dependent properties for CEP								**TR1** (F10.4.18.3)
THETA (R) Parameter defining differencing technique for transient equation. Corresponds to Θ in Eq. F10.2.38. $0.5 \leq \Theta \leq 1.0$ 0.5 = Crank-Nicolson 1.0 = Backwards Euler	RESDUL (R) Convergence criterion for implicit solution. Corresponds to ϵ_1 in Eq. F10.2.52 (Default: 10^5)	NITREZ (I) Number of iterations in linear loop between tests for convergence (Default: 1)	RELDIF (R) Relative convergence criterion for nonlinear (temperature-dependent properties) loop. Corresponds to ϵ_2 in Eq. F10.2.55 (Default: 10^5)	ABSDIF (R) Absolute convergence criterion for nonlinear (temperature-dependent properties) loop (Default: 0)	BETAT (R) Initial value for SOR acceleration parameter. Corresponds to ω in Eq. F10.2.57. =0, optimize empirically <0, Carre's optimization >0, use constant value entered BETAT < 2	NUPBTA (I) Number of time steps between attempted acceleration parameter updates. Used when BETAT=0. Corresponds to N_c in Sect. F10.2.3.3.4 (Default: 1)	ITLRCO (I) For BETAT=0, Number-of-iterations criterion to initiate acceleration parameter updates. Corresponds to L_c in Sect. F10.2.3.3.4 (Default: 5) For BETAT<0, Number of time steps between SOR acceleration parameter updates (Default: 12)	ITLRCI (I) Number-of-iterations criterion to terminate acceleration parameter updates. Corresponds to J_c in Sect. F10.2.3.3.4 (Default: 2)	**TR2** (F10.4.18.4)
DELTAT (R) Initial time step for implicit solution. (Default: previous time step if other that first TR3 card)	TSFACT (R) Factor by which the current time step is multiplied at each time step (Default: 1.0)	TSMAX (R) Maximum time step size (Default: 10^{50})	TSCHGE (R) Maximum time for which this series of time steps applies. When the current time exceeds this value, another TR3 card is read (Default: 10^{50})	TPCGMX (R) Maximum temperature change allowed at any node over a time step	PTPCGM (R) Maximum percentage change in temperature allowed at any node over a time step	TSMIN (R) Minimum time step size (Default: DELTAT/10)			**TR3** (F10.4.18.5)

%	Data termination card.	(F10.4.19)

F10.5 OUTPUT DESCRIPTION

F10.5.1 GENERAL

Each execution of HEATING automatically produces two output files: a print file containing a processed version of the input data with a solution status summary and an unformatted plot data file containing the model geometry description, temperature distributions, and phase distributions. Output times at which solution information is written to the print and plot files are specified by the user in the PRINTOUT TIMES data block. A more detailed description of these files is given in the sections below.

Two optional files can be produced. If specified on the Parameter Card (Sect. F10.4.3), all of the convergence information will be written to the convergence information file rather than to the normal print file. If a NODES MONITORED data block is included in the input file, a file containing the temperature history of specified nodes will be created. On UNIX systems there is also a file produced by the shell script used to execute HEATING. This file lists the time and date of the run, the names of all input files supplied, and the names of all files generated by the HEATING run.

F10.5.2 PRINT FILE

The print output generated by the HEATING computer code is best illustrated by example, and the reader is referred to Appendix F10.C. This section identifies the types of output available and gives a brief description of some of the features. The code automatically lists the input data card images, tabulates the input data with descriptive headings, and lists some information generated by the code from the input data. The code also automatically prints information giving the progress of the calculations. Temperatures of selected nodes can be printed at a user-specified frequency to trace the progress of a transient or steady-state calculation (see Sect. F10.4.16).

F10.5.2.1 Input Return

HEATING automatically lists the card images, with each card image numbered. Since this serves to document the input data exactly as they were supplied to the code, the feature assists the user in locating data errors identified later in the calculations. The output also contains information pertaining to the amount of computer memory required for the run, and whether the specified amount of computer memory must be increased to continue the calculations.

For each problem in a run, the standard output contains a heading identifying the version of the code, the date and time of the run, the job name, and the computer on which the job was executed. This information is followed by tables that indicate the maximum number of parameters and the features selected for the problem. These values are followed by a tabulation of the remaining input data with descriptive headings. The code then tabulates some of the data it generates from the input data. These data include the fine lattice lines along each axis, the total number of nodes, and the stability criterion at each node for transient problems.

F10.5.2.2 Steady-State Convergence Information

For steady-state successive-overrelaxation iteration, a table is generated giving the status of the calculations. Every five iterations an entry is made in the table indicating the iteration number, the maximum relative change in temperature over the last iteration (along with its sign), the node where the above maximum occurred, the temperature of the node, and the extrapolation factor as computed by the code. Whenever the overrelaxation factor is modified or whenever an extrapolation occurs, a related message is written in the table.

For the nonlinear direct-solution technique, the following output is generated every iteration to indicate the status of the calculations: the iteration number, the maximum relative residual and the node where it occurred, along with the node's total heat flow, average heat flow, heat residual, and temperature; and the iteration number, the maximum residual and the node where it occurred, along with the node's total heat flow, average heat flow, and temperature.

For the conjugate gradient solution technique, a table is printed out that gives the status of the solution every fifth iteration. This table includes the number of iterations completed, the average residual for all nodes, the maximum residual, the node at which the maximum residual occurred, and the temperature of this node. For a nonlinear solution, a message is printed out indicating when the linear problem has converged and properties are being reevaluated. A new convergence table is then started with the linear iteration counter reset to zero.

F10.5.2.3 Implicit Transient Convergence Information

For the transient implicit procedure, a table is generated giving the status of the calculations. This table appears between normal printouts and includes the time-level number, the time level, the number of iterations required for both the linear and nonlinear loops to converge, the values that must satisfy the convergence criterion for both the linear and nonlinear loops, the maximum absolute temperature change over the time increment and the node where it occurred, the maximum percentage relative change in temperature over the time increment and the node where it occurred, and other information concerning the status of the numerical procedure being used.

F10.5.2.4 Solution Summary at Printout Times

At specified printout times, as well as at the completion of a transient or steady-state solution, a summary of the solution status is printed. This summary consists of the current time-step size (transient cases only), the solution time, and the elapsed cpu time since the calculation was started. The maximum and minimum temperatures and the nodes where they occur in the problem are then written. If any heat generation functions are specified, the current rate of energy input to the model due to each function is listed in columns identified as modeled and neglected heat generation. Neglected heat generation is that which is associated with nodes that are on specified-surface-temperature boundaries. The neglected heat generation column will always be zero if there are no Type-2 boundary conditions specified. A summary of boundary temperatures and the heat flow rates on all boundaries follows. The column labeled "neglected heat flow" will only have a value in it if there is a node that has a specified-surface-temperature (Type-2) boundary condition applied to one surface and a different boundary condition applied to another surface. The heat flow listed for each boundary does not include any specified fluxes. The net heat flow due to specified fluxes for all boundary conditions is listed separately. If the neglected heat generation or the neglected heat flow on a boundary is very significant compared with the modeled values, the user may need to reevaluate the model. For models having a boundary condition that has a position-dependent temperature, a table is output giving the node number and coordinates of each internal node on that boundary, along with the value for the position-dependent boundary temperature.

F10.5.2.5 Special Monitoring of Temperatures

The temperatures at a few specified nodes may be tabulated as a function of the number of time steps for transient problems and as a function of the number of iterations for steady-state problems. This editing feature allows the monitoring of the temperatures at a few nodes of interest. See Sect. F10.4.16 for more details on how to use this option.

F10.5.2.6 Output of Selected Information During Calculations

Selected information generated during the calculations may be output to assist the user in the detection of input data errors, in better understanding cases that are not performing as anticipated or that are suspected of being in error, and in debugging changes to the code.

This output consists of the following tables: the location of each node and a list of its neighbors; a list of nodal connectors for surface-to-surface boundary conditions; the regions that comprise each node; the temperature, heat capacitance, and power associated with each node and an indicator of whether or not the heat capacitance, effective conductance, and power at each node are dependent on temperature; each neighbor and related effective conductance of each connector for every node; phase-change information at each node; and the steady-state or transient temperature distributions at the specified times. See Sect. F10.4.3 for more details on the use of this feature.

F10.5.3 UNFORMATTED PLOT DATA FILE

An unformatted plot data file is produced by every successful execution of HEATING. This file, which is used for graphical postprocessing and restarting calculations, contains the model geometry description, temperature distributions, and phase distributions. This file automatically contains the initial temperature and phase distributions for each transient calculation and the final distributions for all steady-state and transient calculations. Additional output times are specified by the user in the PRINTOUT TIMES data block.

Several postprocessing options are available to produce graphical output from the HEATING solution stored in the unformatted plot data file. Three interface codes (H7TECPLOT, and H7PATRAN) are available that read the unformatted plot data file and produce formatted files for postprocessing with TECPLOT, and PATRAN, respectively. If the user wishes to produce customized tabular output or specialized graphics, it may be necessary to access this file in order to produce the desired output. For that reason the format of the file and a description of the information stored in it are given in Table F10.5.1. Even though HEATING is a finite-volume code, the plot file contains element definitions. These element definitions are used to describe the geometry to the postprocessor codes - HEATING does not use them internally. In Table F10.5.1, character, integer, and real variables are identified with C, I, and R, respectively. All real variables are eight bytes in length, and all integer variables are the default length for the computer on which HEATING is being run (eight bytes on a CRAY and four bytes on most other machines.)

Table F10.5.1 Contents of unformatted plot data file

Record 1: JOBDES, NGEOM, IT, JT, KT, NT, NBDTPT, MATSL, IERR, IWARN, CURTIM, VERSUN

JOBDES	C*72	Job description
NGEOM	I	Geometry type
IT	I	Number of grid lines along first axis
JT	I	Number of grid lines along second axis
KT	I	Number of grid lines along third axis
NT	I	Number of nodes
NBDTPT	I	Number of boundary conditions
MATSL	I	Number of unique phase changes
IERR	I	Number of errors produced during model generation
IWARN	I	Number of warnings produced during model generation
CURTIM	C*24	Time and date of HEATING run
VERSUN	C*15	Version used for analysis (e.g., HEATING 7.2b)

Record 2: (R(I),I=1,IT)

R	R	Array of grid line values along first axis

Record 3: (TH(I),I=1,JT)

TH	R	Array of grid line values along second axis

Record 4: (Z(I),I=1,KT)

Z	R	Array of grid line values along third axis

Table F10.5.1 (continued)

Record 5: (N,NTPI(N),NTPJ(N),NTPK(N),I=1,NT)

N	I	Node number
NTPI	I	Grid line passing through node (first axis)
NTPJ	I	Grid line passing through node (second axis)
NTPK	I	Grid line passing through node (third axis)

Record 6: NELEM,NODES

NELEM	I	Number of elements
NODES	I	Number of nodes defining each element

Records 7 through NELEM+6: MATL,(NODE(I),I=1,NODES)

MATL	I	Material in element
NODE	I	Array of nodes defining element

Record NELEM+7: NSET, NOIT, DELTAT, TIM, IERR, IWARN

NSET	I	Solution type: -1=transient, +1=steady-state
NOIT	I	Number of steady-state iterations or transient time steps
DELTAT	R	Time step
TIM	R	Time
IERR	I	Number of errors produced by run to this point
IWARN	I	Number of warnings produced by run to this point

Record NELEM+8:

No Phase Change	(T1(I),I=1,NT),(TDUM(I),I=1,NBDTPT)
Phase Change	(T1(I),I=1,NT),(X1(I),I=1,NT),(TDUM(I),I=1,NBDTPT)

T1	R	Temperature array
X1	R	Phase fraction array
TDUM	R	Boundary temperature array

Additional Records: The previous two records are repeated for each temperature distribution that is output.

F10.6 REFERENCES

1. T. B. Fowler and E. R. Volk, *Generalized Heat Conduction Code for the IBM-704 Computer*, ORNL-2734, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., October 1959.

2. R. R. Liguori and J. W. Stephenson, *The HEATING Program,* ASTRA 417-5.0, ASTRA, Inc., Raleigh, N.C., January 1961.

3. W. D. Turner and J. S. Crowell, *Notes on HEATING—An IBM 360 Heat Conduction Program*, CTC/INF-980, Union Carbide Corp., Nucl. Div., November 1969.

4. W. D. Turner and M. Siman-Tov, *HEATING3—An IBM 360 Heat Conduction Program*, ORNL/TM-3208, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., February 1971.

5. W. D. Turner, D. C. Elrod, and I. I. Siman-Tov, *HEATING5—An IBM 360 Heat Conduction Program*, ORNL/CSD/TM-15, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., March 1977.

6. D. C. Elrod, G. E. Giles, and W. D. Turner, "HEATING6: A Multidimensional Heat Conduction Analysis with the Finite-Difference Formulation," Sect. F10 of *SCALE: A Modular Code System for Performing Standardized Computer Analyses for Licensing Evaluation*, NUREG/CR-0200, Rev. 2, U.S. Nuclear Regulatory Commission, 1984.

7. K. W. Childs, G. E. Giles, C. B. Bryan, and C. K. Cobb, "HEATING: A Computer Program for Multidimensional Heat Transfer Analysis (Version 6.1)," Sect. F10 of *SCALE: A Modular Code System for Performing Standardized Computer Analyses for Licensing Evaluation,* ORNL/NUREG/CSD-2/V2/R3, Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., draft February 1990. Available from Radiation Shielding Information Center as CCC-545.

8. K. W. Childs, *HEATING 7.0 User's Manual*, K/CSD/INF/90-32, Martin Marietta Energy Systems, Inc., July 1990.

9. K. W. Childs, *HEATING 7.1 User's Manual*, K/CSD/TM-96, Martin Marietta Energy Systems, Inc., Oak Ridge Gaseous Diffusion Plant, July 1991.

10. L. M. Petrie and N. F. Cross, *KENO IV—An Improved Monte Carlo Criticality Program*, ORNL-4938, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., November 1975.

11. C. B. Bryan, K. W. Childs, and G. E. Giles, *HEATING6 Verification*, K/CSD/TM-61, Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., December 1986.

12. R. S. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1962.

13. J. Dongarra, C. B. Moler, J. R. Bunch, and G. W. Stewart, "LINPACK User's Guide," *SIAM*, Philadelphia, PA, 1979.

14. B. R. Becker, *A Direct Solution Technique for Solving Steady-State Problems Using the HEATING6 Heat Transfer Code*, K/CSD/TM-15, Union Carbide Corp., Nucl. Div., Oak Ridge Gaseous Diffusion Plant, October 1977.

15. G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2nd Ed., The Johns Hopkins University Press, Baltimore, MD, 1989.

16. S. K. Hellman, G. Habetler, and H. Babrov, "Use of Numerical Analysis in the Transient Solution of Two-Dimensional Heat-Transfer Problem with Natural and Forced Convection," *Trans. Am. Soc. Mech. Engrs.* **78**, 1155-61 (1956).

17. S. Levy, *Use of "Explicit Method" in Heat Transfer Calculations with an Arbitrary Time Step*, 68-C-282, General Electric, Schenectady, NY, August 1968.

18. B. A. Carre, "The Determination of the Optimum Accelerating Factor for Successive Over-relaxation," *Comp. J.* **4**, 73-78 (1961).

F10.A RUNNING HEATING IN SCALE

HEATING is run in SCALE using the same shell script as other SCALE modules. The input and output files and the file naming convention used by the SCALE shell script are listed below.

HEATING INPUT AND OUTPUT FILES

File description	File name [a]
Input file	*ifile*.input
System-generated messages and a list of names of all files used or generated by this execution of HEATING.	*ifile*.msgs [b]
Print file. This file is created for every execution of HEATING.	*ifile*.output [b]
Plot file. This file is created for every execution of HEATING.	plot*jid* [c] *ifile*.plot*jid* [b]
Restart file. Provided by the user for restart cases.	*ifile*.restart *ifile*.rst [d]
Node map file	map*jid* [c] *ifile*.map*jid* [b]
Unformatted (binary) node connectors file	*ifile*.connect *ifile*.con [d]
Nodes monitored file. This file is created if a NODES MONITORED data block is included in the input.	h7node*jid* [c] *ifile*.h7node*jid* [b]
Convergence information file. This file is created if the sixth entry on the Parameter Card is greater than zero.	h7cvrg*jid* [c] *ifile*.h7cvrg*jid* [b]

[a] *jid* is an integer number used to uniquely identify each output file from a particular job.

[b] As written by SCALE shell scripts to user's directory.

[c] As written by HEATING to working directory.

[d] PC version only.

F10.B VERIFICATION PROCEDURE

The HEATING series of general-purpose, finite-difference, conduction heat transfer codes have been in use for many years. During this time the codes have been used extensively, and a general confidence has been developed in regard to their accuracy. In the development of each new version of HEATING, great care was taken to ensure that the code did produce accurate solutions to heat transfer problems. Unfortunately, for the most part, the checking that was done was not documented, nor was a formal verification ever presented in a citable document for any version of the code prior to HEATING6. In 1985 and 1986 a concerted effort was undertaken to formally verify and document HEATING6.[1] A rigorous verification was carried out in which HEATING temperature solutions were compared against analytical solutions obtained from the literature. Twenty-three analytical solutions were chosen in order to test various HEATING analysis options. The verification cases focused on 1-D Cartesian problems because of the abundance of available analytical solutions, but all of the HEATING geometry options were checked. There have been some minor modifications made to the verification cases as a result of changes in the capability of newer versions of HEATING.

The comparison-to-analytical-solution cases are not sufficient to adequately check all features of the code since they only exercise one option at a time. Any problems resulting as a consequence of the interaction of two or more features would not be discovered. Therefore, three additional reference cases were developed which exercise several analysis options simultaneously. These problems were intended to serve as a basis of comparison for any future modifications to the code or implementation of the code on other computer systems.

Once a version of HEATING has been formally verified against analytical solutions, then that version can be used to verify subsequent versions of the code. In this manner HEATING6 was used to verify HEATING 6.1 by manually comparing the printed output produced by the two codes for all of the verification and reference cases. Even though this approach is more expedient than comparing against the analytical solution, it is very laborious and potentially error prone. For these reasons a computer program, HEATCHEK,[2] was developed to automate the verification procedure. HEATCHEK compares the information written to the plot data file by the old and new versions of HEATING. This version of HEATING was verified using the HEATCHEK verification procedure.

REFERENCES

1. C. B. Bryan, K. W. Childs, and G. E. Giles, *HEATING6 Verification*, K/CSD/TM-61, Martin Marietta Energy Systems, Inc., Oak Ridge Gaseous Diffusion Plant, December 1986.

2. W. Chu, *HEATCHEK: A Computer Program to Automate Verification of New Versions of HEATING*, K/CSD/INF-89/4, Martin Marietta Energy Systems Inc., Oak Ridge Gaseous Diffusion Plant, March 1989.

F10.C SAMPLE PROBLEMS

F10.C.1 SAMPLE PROBLEM 1

A stainless steel wire with a radius of 1.5 mm has an electric current passing through it. The resulting internal heat generation rate per unit volume is 1.12 × 10^9 W/m^3. The surface of the wire loses heat by convection to the environment. The convective heat transfer coefficient and ambient temperature are 4000 W/m^2-K and 110°C, respectively. The conductivity of stainless steel is 19 W/m-K. The steady-state temperature distribution within the wire is to be determined. The centerline temperature is of particular interest since this is the hottest location.

This problem can be modeled in a 1-D, *r*-cylindrical geometry. The HEATING input file for this case is shown in Fig. F10.C.1. The units given in the problem definition are not consistent since the wire radius is given in millimeters and other length units are in meters. In the input data millimeters are used for the length unit throughout.

```
Sample Problem 1
* 3 mm stainless steel wire with internal heat generation
* Units:  J, kg, s, mm, C
10  4                                                                   P
REGIONS
1  1  0.0  1.5                                                          R1
0  1   0    1                                                           R2
MATERIALS
1  S-Steel  0.019                                                       M
HEAT GENERATIONS
1  1.12                                                                 G
BOUNDARY CONDITIONS
1  1  110.0                                                             B1
4.0e-3                                                                  B2
XGRID
0.0  1.5                                                                X1
    8                                                                   X2
STEADY-STATE
2                                                                       SS
%
```

Figure F10.C.1 Input data for sample problem 1

The first card in the data file is the Title Card. Immediately following the Title Card are some comment cards (indicated by the '*' in column 1). It is a good practice to include comment cards at this location to document what the file is for future reference. Parameter card 1 indicates that a maximum of 10 s of cpu time is allowed before terminating the solution and that the geometry is one-dimensional *r*-cylindrical. Since an initial time is not entered, it defaults to zero. Even though the temperature units for this problem are °C, it is not necessary to enter a value for it on the parameter card since radiation is not being modeled. None of the remaining options on the parameter card are being exercised, so the remainder of the card is left blank.

One region is defined in the REGIONS data block. This region definition references material number 1, heat generation number 1, and boundary condition number 1, which are all defined in their corresponding data blocks. The only material property required is the conductivity since only a steady-state solution is requested. The XGRID data block defines the mesh spacing. The radius of the wire is divided into eight equal divisions, which results in nine nodes being defined. Since a steady-state solution is desired, a STEADY-

STATE data block is included. The entry of 2 on the SS card indicates that the direct-solution technique is to be used. The card identifications contained in columns 73–80 are not required.

Every successful HEATING execution produces a print file and a plot file. Additional files may be generated optionally. The plot file is an unformatted file that is used as input for postprocessing with one of the codes discussed in Appendix F10.D. A sample print file for this case is given in Fig. F10.C.2.

As is evident from the listing in Fig. F10.C.2, the only nodal temperatures in the print file are the minimum and maximum temperatures occurring in the model. To obtain information about the temperature at other locations it is necessary to postprocess the information in the plot file. Several graphical and tabular forms of output are available (see Appendix F10.D for details.) As an example of tabular output, Fig. F10.C.3 presents the temperature distribution obtained by postprocessing with H7MAP.

If the user decides that the temperature solution during the transient cooldown following a power shutoff is needed after completing the steady-state solution, a transient case can be run using the previous steady-state solution as the initial conditions. The original input file can be easily modified to produce the input file for the transient. The contents of this modified file are shown in Fig. F10.C.4.

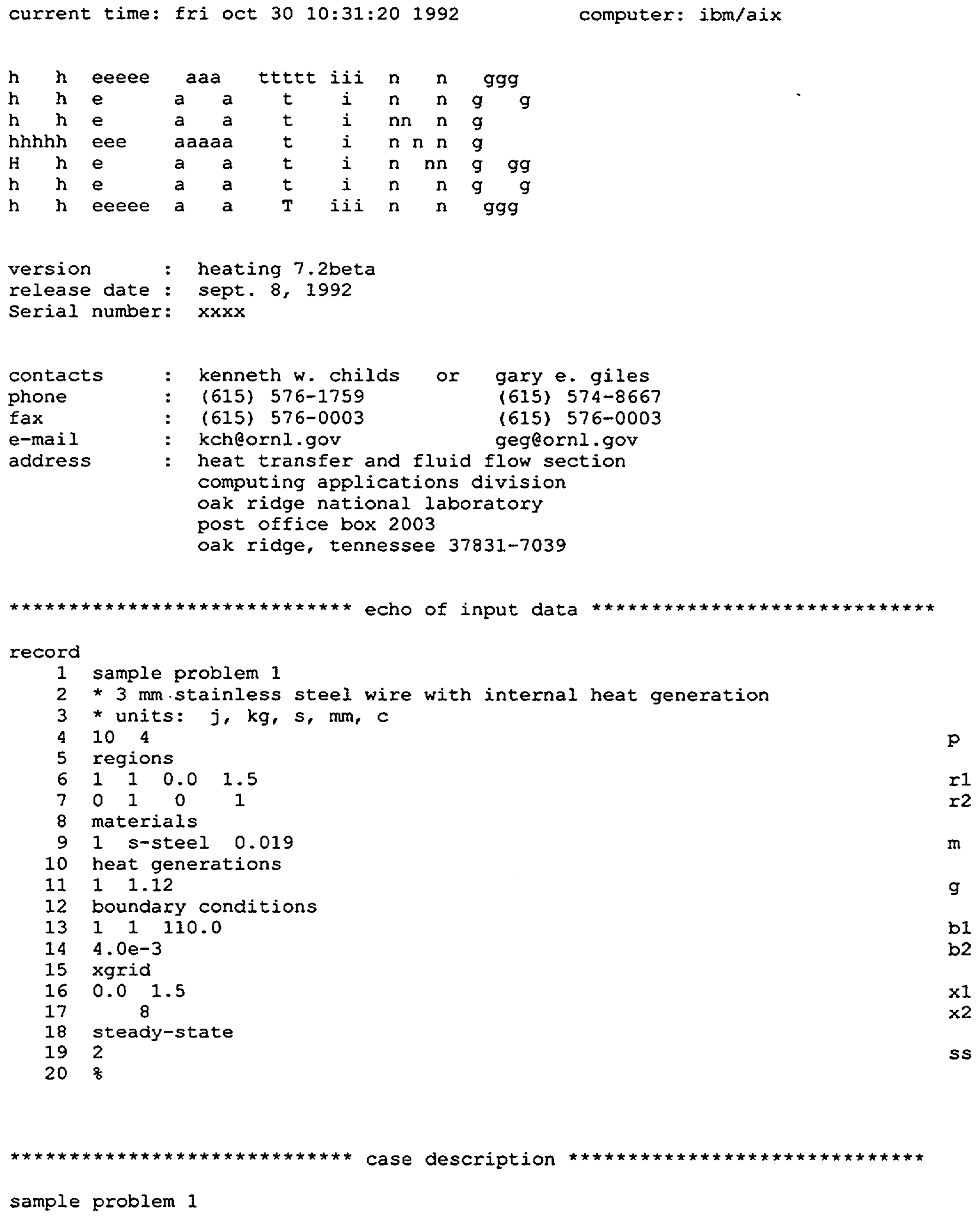
Several things should be noted about the modifications to the input file for the transient case. First, the entry for heat generation number on the region definition (second entry on card R2) has been changed to a '0'. The HEAT GENERATION data block was also removed, but this was not actually necessary. Second, density and specific heat for stainless steel must now be supplied. The values for density and specific heat are 7865 kg/m^3 and 460 J/kg-°C, respectively. Since the length unit in the input is mm, the value for density had to be converted. Third, a PRINTOUT TIMES data block has been added. If this data block had not been included the only information that would have been written to the print and plot files would be for times 0 s and 5 s (i.e., the beginning and ending times of the transient solution.) Fourth, the STEADY-STATE data block has been replaced with a TRANSIENT data block. The TR card indicates that an explicit transient solution is to be performed with a final time of 5 s.

The final portion of the print file for this problem is listed in Fig. F10.C.5 (the beginning of the file is very similar to the print file for sample problem 1).

The steady-state and transient solutions could have both been performed in a single execution of HEATING. The input file for accomplishing this is given in Fig. F10.C.6. The time-dependence of the heat generation is defined by tabular function number 1. At time zero the function has a value of 1.0 which drops to 0.0 over the first 1.0×10^{-6} s of the transient and remains there for the remainder of the transient. This value is multiplied with the constant heat generation value (1.12 W/mm^3) to give the actual heat generation. A subtle difference is noted between this approach and the two-step solution presented earlier. For the first transient time step the heat generation will be evaluated at a time of zero. Thus the heat generation will not be turned off until the beginning of the second time step. Since the time step is quite small, this difference has no practical significance.

F10.C.2 SAMPLE PROBLEM 2

This problem was selected to demonstrate the enclosure radiation-modeling capabilities of HEATING. The model consists of the finned surface, shown in Fig. F10.C.7. A flat plate comprises the base, which has rectangular fins on its exterior surface. The base has a thickness of 100 mm. The fins are 10 mm thick, 150 mm long, and are spaced on a pitch of 60 mm. The base and fins are made of a mild steel with a conductivity of 50 W/m-K, a specific heat of 500 J/kg-K, and a density of 7800 kg/m^3. The finned surface is



```
current time: fri oct 30 10:31:20 1992        computer: ibm/aix


h   h  eeeee   aaa   ttttt iii  n   n   ggg
h   h  e      a   a    t    i   n   n  g   g
h   h  e      a   a    t    i   nn  n  g
hhhhh  eee    aaaaa    t    i   n n n  g
H   h  e      a   a    t    i   n  nn  g  gg
h   h  e      a   a    t    i   n   n  g   g
h   h  eeeee  a   a    T   iii  n   n   ggg


version       :  heating 7.2beta
release date :  sept. 8, 1992
Serial number:  xxxx


contacts      :  kenneth w. childs   or   gary e. giles
phone         :  (615) 576-1759           (615) 574-8667
fax           :  (615) 576-0003           (615) 576-0003
e-mail        :  kch@ornl.gov             geg@ornl.gov
address       :  heat transfer and fluid flow section
                 computing applications division
                 oak ridge national laboratory
                 post office box 2003
                 oak ridge, tennessee 37831-7039


***************************** echo of input data *****************************

record
    1   sample problem 1
    2   * 3 mm stainless steel wire with internal heat generation
    3   * units:  j, kg, s, mm, c
    4   10  4                                                                p
    5   regions
    6   1  1  0.0  1.5                                                       r1
    7   0  1   0    1                                                        r2
    8   materials
    9   1  s-steel  0.019                                                    m
   10   heat generations
   11   1  1.12                                                              g
   12   boundary conditions
   13   1  1  110.0                                                          b1
   14   4.0e-3                                                               b2
   15   xgrid
   16   0.0  1.5                                                             x1
   17       8                                                                x2
   18   steady-state
   19   2                                                                    ss
   20   %


***************************** case description ******************************

sample problem 1
```

Figure F10.C.2 Output for sample problem 1

```
********************** summary of parameter card data **********************

maximum cpu time     -    10.00 seconds
geometry type number -      4 (or r-cy)
initial time         -   0.0000000d+00
temperature units    -   fahrenheit (significant only if radiation involved)

this is a restart of previous case             - no
read node-to-node connector data file          - no
redirect or suppress convergence information   - yes (suppress)

output selected information during calculations - no


*************************** summary of region data **************************

region   material  initial    heat gen.
number    number   temp. No.   number
     1         1         0         1


        ------------------- dimensions / boundary numbers -------------------
 region       First axis             second axis             third axis
 number   smaller     Larger      smaller     larger      smaller     larger
     1  0.0000e+00  1.5000e+00  0.0000e+00  0.0000e+00  0.0000e+00  0.0000e+00
            0           1           0           0           0           0


************************** summary of material data *************************

material   material     ------------ thermal parameters ------------   phase
 number      name       -- temperature-dependent function numbers --  change
                        conductivity      density      specific heat
    1       s-steel     1.900000d-02   0.000000d+00   0.000000d+00     no
                              0              0              0


******************** summary of heat generation rate data *******************

number    power    time-, temperature-, and position-dependent function numbers
         density        time    temperature  x or r    y or theta   z or phi
   1   1.12000d+00        0          0          0          0           0


********************* summary of boundary condition data ********************

number:     1                      type:  surface-to-environment
   temperature and any functions used to define dependence:
             temperature        :  1.100000e+02
   heat transfer coefficients and any functions used to define dependence:
             forced convection  :  4.000000e-03


************************* summary of grid structure *************************

x (or r) gross grid lines and number of divisions
   0.000000e+00   1.500000e+00
              8
```

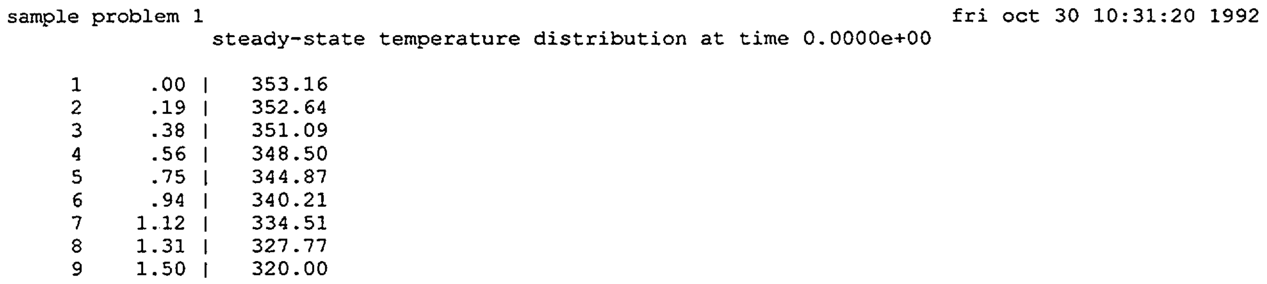
Figure F10.C.2 (continued)

```
x (or r) fine grid lines generated by heating
     1  0.00000e+00      2  1.87500e-01      3  3.75000e-01      4  5.62500e-01
     5  7.50000e-01      6  9.37500e-01      7  1.12500e+00      8  1.31250e+00
     9  1.50000e+00


******************* sources of non-linearity in the model *******************

   the model is linear.


************** number of parameters specified by the input data **************

  regions                                                  1
  materials                                                1
  phase changes                                            0
  initial temperatures                                     0
  heat generations                                         1
  boundary conditions                                      1
  gross grid lines along x or r axis                       2
  fine  grid lines along x or r axis                       9
  gross grid lines along y or theta axis                   1
  fine  grid lines along y or theta axis                   1
  gross grid lines along z or phi axis                     1
  fine  grid lines along z or phi axis                     1
  analytic functions                                       0
  tabular functions                                        0
  node-to-node connectors                                  0
  transient printout times                                 0
  nodes for monitoring of temperatures                     0
  number of nodes                                          9
  number of specified-temperature nodes                    0
  position-dependent boundary temperature nodes            1
  bandwidth for direct solution                            3


************ memory requirements for variably dimensioned arrays ************

  phase 1     3k
  phase 2     4k
  phase 3     4k
  phase 4     5k


***************************** initial conditions ****************************

number of iterations completed  =              0
current problem time            =  0.00000000d+00
elapsed cpu time (hr:min:sec)   =     00:00:00.20

minimum temperature = 0.00000e+00   at node     1
maximum temperature = 0.00000e+00   at node     1

heat generation
  number               current rate (energy/time)
                         (modeled)    (neglected)
     1                  7.91681e+00   0.00000e+00
```

Figure F10.C.2 (continued)

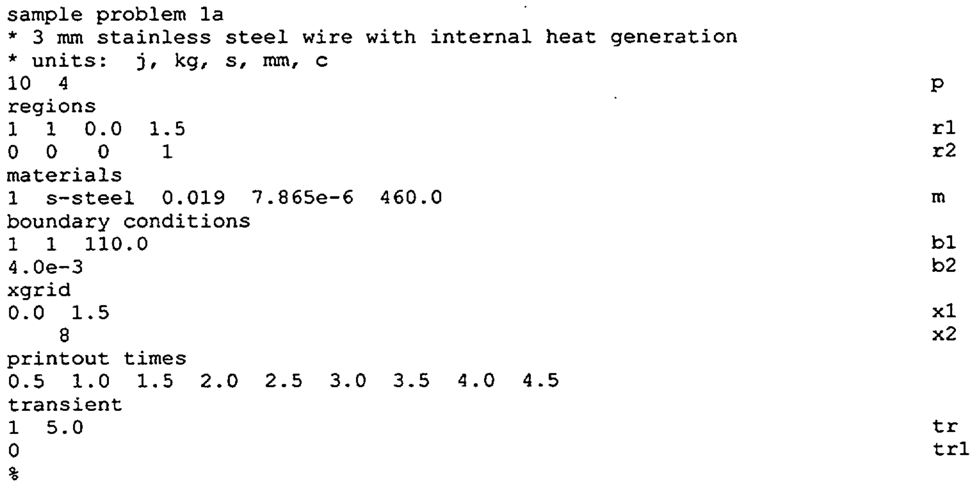
```
boundary heat flow
  number  environment  current rate (energy/time)
          temperature    (modeled)     (neglected)
     1    1.10000e+02   4.14690e+00   0.00000e+00


******************************************************************************

          begin steady state calculation - direct solution technique

******************************************************************************

maximum number of iterations      =    20
convergence criterion             =  1.0000000d-05


************************ steady-state solution output ************************

number of iterations completed  =                1
current problem time            =  0.00000000d+00
elapsed cpu time (hr:min:sec)   =     00:00:00.21

minimum temperature = 3.20000e+02   at node     9
maximum temperature = 3.53158e+02   at node     1

heat generation
  number               current rate (energy/time)
                         (modeled)     (neglected)
     1                  7.91681e+00   0.00000e+00

boundary heat flow
  number  environment  current rate (energy/time)
          temperature    (modeled)     (neglected)
     1    1.10000e+02  -7.91681e+00   0.00000e+00

************************** end of heating execution **************************

sample problem 1

***** number of warnings  --    0
***** number of errors    --    0
```

Figure F10.C.2 (continued)



```
sample problem 1                                                   fri oct 30 10:31:20 1992
                     steady-state temperature distribution at time 0.0000e+00

       1     .00 |   353.16
       2     .19 |   352.64
       3     .38 |   351.09
       4     .56 |   348.50
       5     .75 |   344.87
       6     .94 |   340.21
       7    1.12 |   334.51
       8    1.31 |   327.77
       9    1.50 |   320.00
```

Figure F10.C.3 Contents of file created by H7MAP



```
sample problem 1a
* 3 mm stainless steel wire with internal heat generation
* units:  j, kg, s, mm, c
10  4                                                                    p
regions
1  1  0.0  1.5                                                           r1
0  0   0    1                                                            r2
materials
1  s-steel  0.019  7.865e-6  460.0                                       m
boundary conditions
1  1  110.0                                                              b1
4.0e-3                                                                   b2
xgrid
0.0  1.5                                                                 x1
    8                                                                    x2
printout times
0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5
transient
1  5.0                                                                   tr
0                                                                        tr1
%
```

Figure F10.C.4 Input data for sample problem 1a

```
***************************** initial conditions *****************************

number of time steps completed  =              0
current time step               =  0.00000000d+00
current problem time            =  0.00000000d+00
elapsed cpu time (hr:min:sec)   =     00:00:00.18

minimum temperature = 3.20000e+02   at node     9
maximum temperature = 3.53158e+02   at node     1

heat generation
  number                current rate (energy/time)
                          (modeled)    (neglected)
     1                   0.00000e+00   0.00000e+00

boundary heat flow
  number  environment  current rate (energy/time)
          temperature    (modeled)     (neglected)
     1    1.10000e+02  -7.91681e+00   0.00000e+00


******************************************************************************

               begin transient calculation - explicit technique

******************************************************************************


maximum of the stability criterion -  3.3471525d-03
median of the stability criterion  -  3.3471525d-03
minimum of the stability criterion -  1.6735763d-03 for point     1

the input time step size is  0.0000000d+00.
the time step size will be set to the stability criterion of  1.6735763d-03.

************************** transient solution output *************************

number of time steps completed  =            299
current time step               =  1.67357627d-03
current problem time            =  5.00399306d-01
elapsed cpu time (hr:min:sec)   =     00:00:00.20

minimum temperature = 2.15800e+02   at node     9
maximum temperature = 2.33167e+02   at node     1

heat generation
  number                current rate (energy/time)
                          (modeled)    (neglected)
     1                   0.00000e+00   0.00000e+00

boundary heat flow
  number  environment  current rate (energy/time)
          temperature    (modeled)     (neglected)
     1    1.10000e+02  -3.98858e+00   0.00000e+00
```

Figure F10.C.5 Partial output listing for sample problem 1a

```
************************* transient solution output ************************

number of time steps completed  =             598
current time step               =  1.67357627d-03
current problem time            =  1.00079861d+00
elapsed cpu time (hr:min:sec)   =     00:00:00.22

minimum temperature = 1.63412e+02   at node     9
maximum temperature = 1.72179e+02   at node     1

heat generation
  number                current rate  (energy/time)
                          (modeled)     (neglected)
     1                   0.00000e+00   0.00000e+00

boundary heat flow
  number   environment  current rate  (energy/time)
           temperature    (modeled)     (neglected)
     1     1.10000e+02  -2.01358e+00   0.00000e+00

************************* transient solution output ************************

number of time steps completed  =             896
current time step               =  1.67357627d-03
current problem time            =  1.49952434d+00
elapsed cpu time (hr:min:sec)   =     00:00:00.25

minimum temperature = 1.37026e+02   at node     9
maximum temperature = 1.41462e+02   at node     1

heat generation
  number                current rate  (energy/time)
                          (modeled)     (neglected)
     1                   0.00000e+00   0.00000e+00

boundary heat flow
  number   environment  current rate  (energy/time)
           temperature    (modeled)     (neglected)
     1     1.10000e+02  -1.01886e+00   0.00000e+00

************************* transient solution output ************************

number of time steps completed  =            1195
current time step               =  1.67357627d-03
current problem time            =  1.99992365d+00
elapsed cpu time (hr:min:sec)   =     00:00:00.28

minimum temperature = 1.23644e+02   at node     9
maximum temperature = 1.25883e+02   at node     1

heat generation
  number                current rate  (energy/time)
                          (modeled)     (neglected)
     1                   0.00000e+00   0.00000e+00
```

Figure F10.C.5 (continued)

```
boundary heat flow
  number  environment  current rate (energy/time)
          temperature    (modeled)    (neglected)
     1    1.10000e+02  -5.14358e-01   0.00000e+00


************************* transient solution output *************************

number of time steps completed  =            1494
current time step               =  1.67357627d-03
current problem time            =  2.50032295d+00
elapsed cpu time (hr:min:sec)   =     00:00:00.32

minimum temperature = 1.16888e+02   at node     9
maximum temperature = 1.18019e+02   at node     1

heat generation
  number               current rate (energy/time)
                         (modeled)    (neglected)
     1                  0.00000e+00   0.00000e+00

boundary heat flow
  number  environment  current rate (energy/time)
          temperature    (modeled)    (neglected)
     1    1.10000e+02  -2.59667e-01   0.00000e+00


************************* transient solution output *************************

number of time steps completed  =            1793
current time step               =  1.67357627d-03
current problem time            =  3.00072226d+00
elapsed cpu time (hr:min:sec)   =     00:00:00.33

minimum temperature = 1.13477e+02   at node     9
maximum temperature = 1.14048e+02   at node     1

heat generation
  number               current rate (energy/time)
                         (modeled)    (neglected)
     1                  0.00000e+00   0.00000e+00

boundary heat flow
  number  environment  current rate (energy/time)
          temperature    (modeled)    (neglected)
     1    1.10000e+02  -1.31090e-01   0.00000e+00


************************* transient solution output *************************

number of time steps completed  =            2091
current time step               =  1.67357627d-03
current problem time            =  3.49944799d+00
elapsed cpu time (hr:min:sec)   =     00:00:00.37

minimum temperature = 1.11759e+02   at node     9
maximum temperature = 1.12048e+02   at node     1
```

Figure F10.C.5 (continued)

```
heat generation
  number                current rate (energy/time)
                          (modeled)    (neglected)
     1                   0.00000e+00   0.00000e+00

boundary heat flow
  number  environment   current rate (energy/time)
          temperature     (modeled)    (neglected)
     1    1.10000e+02  -6.63305e-02   0.00000e+00

************************** transient solution output *************************

number of time steps completed   =            2390
current time step                =   1.67357627d-03
current problem time             =   3.99984730d+00
elapsed cpu time (hr:min:sec)    =      00:00:00.38

minimum temperature = 1.10888e+02   at node     9
maximum temperature = 1.11034e+02   at node     1

heat generation
  number                current rate (energy/time)
                          (modeled)    (neglected)
     1                   0.00000e+00   0.00000e+00

boundary heat flow
  number  environment   current rate (energy/time)
          temperature     (modeled)    (neglected)
     1    1.10000e+02  -3.34861e-02   0.00000e+00

************************** transient solution output *************************

number of time steps completed   =            2689
current time step                =   1.67357627d-03
current problem time             =   4.50024660d+00
elapsed cpu time (hr:min:sec)    =      00:00:00.41

minimum temperature = 1.10448e+02   at node     9
maximum temperature = 1.10522e+02   at node     1

heat generation
  number                current rate (energy/time)
                          (modeled)    (neglected)
     1                   0.00000e+00   0.00000e+00

boundary heat flow
  number  environment   current rate (energy/time)
          temperature     (modeled)    (neglected)
     1    1.10000e+02  -1.69051e-02   0.00000e+00
```

Figure F10.C.5 (continued)

```
************************* transient solution output *************************

number of time steps completed  =           2988
current time step               =  1.67357627d-03
current problem time            =  5.00064591d+00
elapsed cpu time (hr:min:sec)   =     00:00:00.43

minimum temperature = 1.10226e+02   at node     9
maximum temperature = 1.10264e+02   at node     1

heat generation
  number                current rate (energy/time)
                          (modeled)    (neglected)
     1                   0.00000e+00   0.00000e+00

boundary heat flow
  number  environment  current rate (energy/time)
          temperature    (modeled)    (neglected)
     1    1.10000e+02  -8.53431e-03   0.00000e+00


the transient calculations have been completed.
final time is     5.00065d+00
number of time steps completed =  2988

************************* end of heating execution *************************

sample problem 1a

***** number of warnings  --    0
***** number of errors    --    0
```

Figure F10.C.5 (continued)

```
sample problem 1b
* 3 mm stainless steel wire with internal heat generation
* units:  j, kg, s, mm, c
10  4                                                                   p
regions
1  1  0.0  1.5                                                          r1
0  1   0    1                                                           r2
materials
1  s-steel  0.019   7.865e-6  460.0                                     m
heat generations
1  1.12  -1                                                             g
boundary conditions
1  1  110.0                                                             b1
4.0e-3                                                                  b2
xgrid
0.0  1.5                                                                x1
    8                                                                   x2
tabular functions
1                                                                       t1
0.0  1.0  1.0e-6  0.0  5.0  0.0                                         t2
printout times
0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5
steady-state
2                                                                       ss
transient
1   5.0                                                                 tr
0                                                                       tr1
%
```

Figure F10.C.6 Input data for sample problem 1b

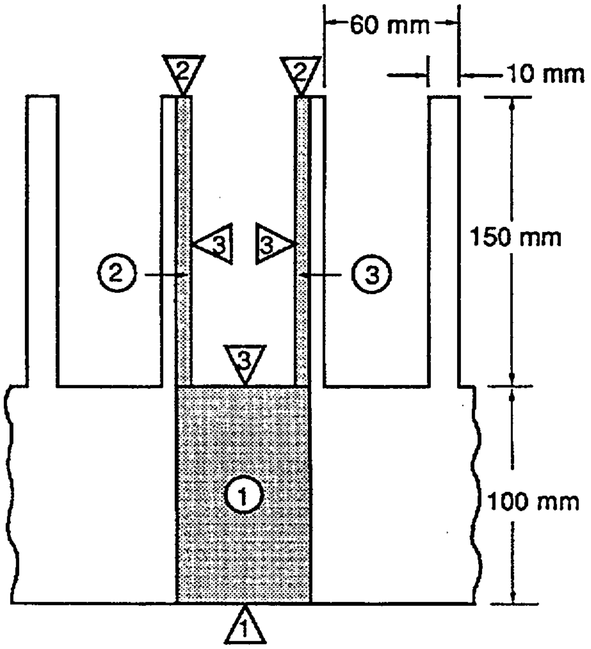


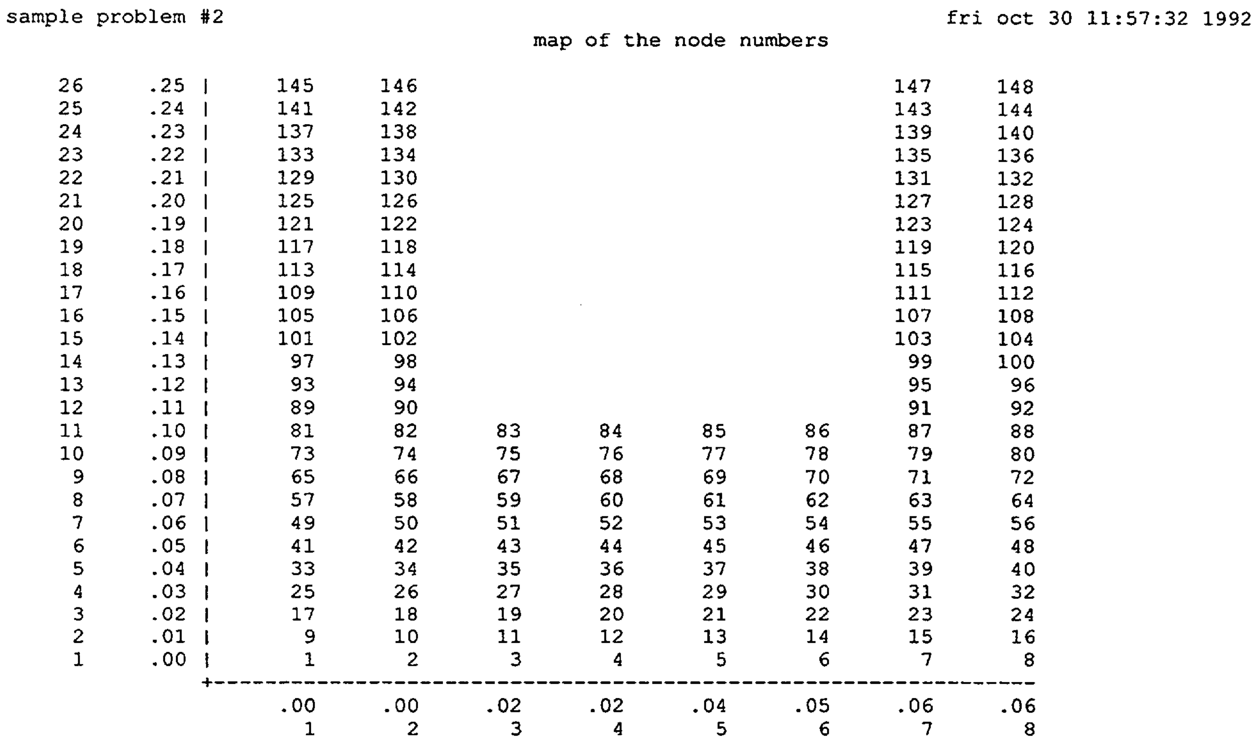
Figure F10.C.7 Sample problem 2 schematic

surface is assumed to be a diffusely emitting and reflecting gray surface with an emissivity of 0.8. Heat transfer on the outside (finned) surface is by natural convection and radiation. The environmental emissivity is assumed to be 1.0. The external environment is at 38°C. The natural convection heat transfer coefficient is $2.0 |T_s - T_a|^{1/3}$ W/m^2-K, where T_s is the surface temperature and T_a is the ambient temperature. Heat transfer on the inside surface is by forced convection, with a heat transfer coefficient of 1000.0 W/m^2-K. The temperature of the fluid adjacent to the inside surface is 100°C. A steady-state solution is desired.

The input data for this sample problem are given in Fig. F10.C.8. In the input data the units are the following: energy, J; length, m; time, s; mass, kg; and temperature, °C. Three regions are used to model the problem. Boundary condition 1 models the forced convection on the inside surface. Boundary condition 2 models the combined radiation and natural convection heat transfer from the ends of the fins. Boundary condition 3 models the natural convection portion of the heat transfer from the surface of the cavity formed by adjacent fins. The radiation portion of the heat transfer from these surfaces is modeled with node-to-node connectors. Boundary condition 4 specifies that radiation is the only active heat transfer mechanism for the node-to-node connectors. A preliminary run of HEATING without the node-to-node connectors produces a plot data file. This file can then be used to determine the node numbers on the surfaces of interest with H7MAP, TECPLOT, or PATRAN (see Appendix F10.D). A node number map produced by H7MAP is presented in Fig. F10.C.9.

```
sample problem #2
10 7 0 1
regions
1   1   0.0    0.060  0.0     0.10
1   0   0      0      1       3
2   1   0.0    0.005  0.10    0.25
1   0   0      3      0       2
3   1   0.055  0.060  0.10    0.25
1   0   3      0      0       2
materials
1   mldsteel 50.0 7800.0  500.0
initial temperatures
1 1.0 0 -1
boundary conditions
1  1  100.0
1000.0
2  1  38.0
0  4.5359d-8  2.0  0.33
3  1  38.0
0  0  2.0  0.33
4  1  38.0
0  5.6699d-8
xgrid
0.00  0.005  0.055  0.060
     1      5      1
ygrid
0.0   0.10  0.25
   10     15
tabular functions
1
0.0  65.0   0.250  95.0
steady state
2  20  1.0-8
%
```

Figure F10.C.8 Input data for sample problem 2



```
sample problem #2                                                          fri oct 30 11:57:32 1992
                                        map of the node numbers
     26      .25 |     145      146                                           147      148
     25      .24 |     141      142                                           143      144
     24      .23 |     137      138                                           139      140
     23      .22 |     133      134                                           135      136
     22      .21 |     129      130                                           131      132
     21      .20 |     125      126                                           127      128
     20      .19 |     121      122                                           123      124
     19      .18 |     117      118                                           119      120
     18      .17 |     113      114                                           115      116
     17      .16 |     109      110                                           111      112
     16      .15 |     105      106                                           107      108
     15      .14 |     101      102                                           103      104
     14      .13 |      97       98                                            99      100
     13      .12 |      93       94                                            95       96
     12      .11 |      89       90                                            91       92
     11      .10 |      81       82       83       84       85       86       87       88
     10      .09 |      73       74       75       76       77       78       79       80
      9      .08 |      65       66       67       68       69       70       71       72
      8      .07 |      57       58       59       60       61       62       63       64
      7      .06 |      49       50       51       52       53       54       55       56
      6      .05 |      41       42       43       44       45       46       47       48
      5      .04 |      33       34       35       36       37       38       39       40
      4      .03 |      25       26       27       28       29       30       31       32
      3      .02 |      17       18       19       20       21       22       23       24
      2      .01 |       9       10       11       12       13       14       15       16
      1      .00 |       1        2        3        4        5        6        7        8
                 +----------------------------------------------------------------------
                       .00      .00      .02      .02      .04      .05      .06      .06
                         1        2        3        4        5        6        7        8
```

Figure F10.C.9 Node numbers on surface of fins

Node-to-node connector data must be calculated external to HEATING and supplied either in an unformatted node-to-node connector file or in the CONNECTOR data block. Since connector data for a radiation problem can be quite voluminous, it is generally more convenient to supply an unformatted connector file to HEATING as is done in this sample problem. In the calculation of the radiation exchange factors, it is necessary to define a complete enclosure to account for all of the radiation heat transfer. Since there is not a physically defined enclosure in this problem, the plane extending between the tips of adjacent fins can be included as a fictitious surface. A negative node number (e.g., -4, indicating the external environment modeled by boundary condition 4) can used to indicate radiation to this fictitious surface.

The printed output for running this sample problem with node-to-node connectors is not presented in this report.

F10.C.3 SAMPLE PROBLEM 3

This sample problem was defined for instructive purposes and is not meant to represent a real engineering problem. The problem is two-dimensional in *x-y* coordinates and consists of three materials. Its configuration is shown in Fig. F10.C.10. Numbers in circles identify regions, and numbers in triangles identify boundary conditions. The units used were Btu, °F, lb, in., and min.

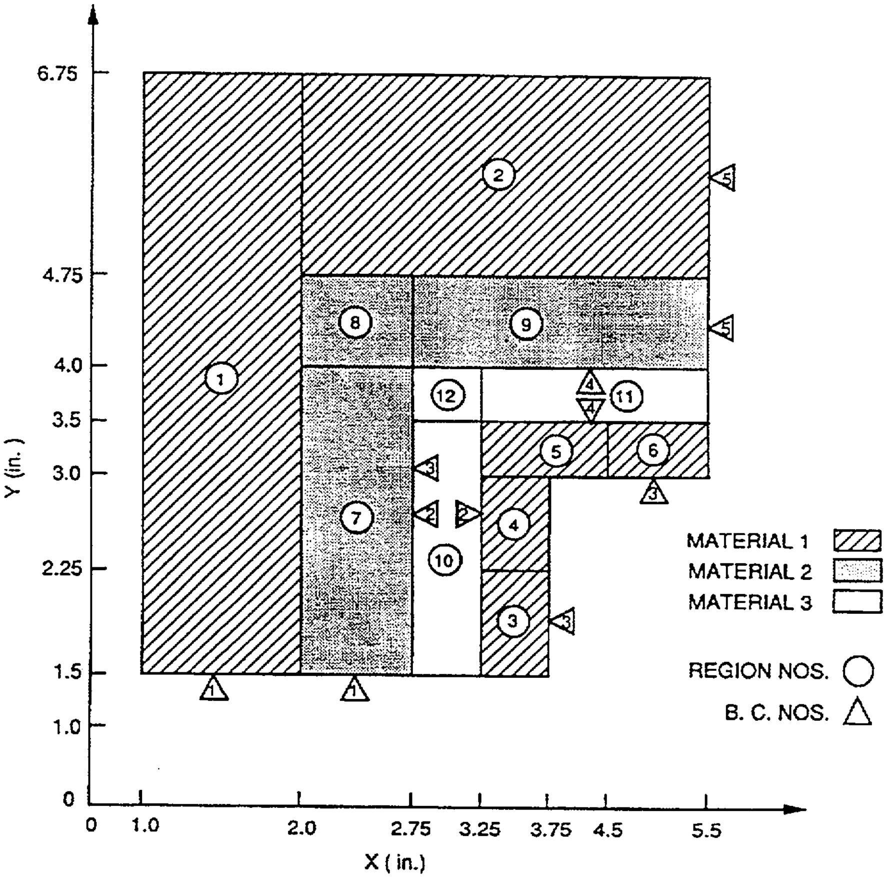


Figure F10.C.10 Sample problem 3 schematic

Regions 1 to 6 contain material 1 (iron); regions 7 to 9 contain material 2 (stainless steel). An air gap between the two metals is modeled by regions 10, 11 and 12. The physical properties of these materials are given in Table F10.C.1.

Table F10.C.1 Material physical properties for sample problem 3

Property/Material	Iron (Material No. 1)	Stainless Steel (Material No. 2)	Air (Material No. 3)
Conductivity Btu/min-in.-°F	0.0296 at 0°F 0.0264 at 752°F 0.0222 at 1832°F	0.013 at 0°F 0.0153 at 752°F 0.025 at 1832°F	1.82×10^{-5} at 0°F 3.41×10^{-5} at 500°F 4.68×10^{-5} at 1000°F 5.75×10^{-5} at 1500°F
Density lb/in.^{3}	0.2801	0.2824	5.00×10^{-5} at 0°F 2.39×10^{-5} at 500°F 1.57×10^{-5} at 1000°F 1.17×10^{-5} at 1500°F
Specific Heat Btu/lb-°F	0.116	0.11	0.25

A spatially uniform heat generation exists in regions 1 and 2 at the rate of 1.0 Btu/(min-in.^{3}), which varies according to time function 2 given in Fig. F10.C.11. The initial temperature is a uniform 100°F. The boundary conditions on each of the faces are shown in Fig. F10.C.10, and they are numbered in triangular frames. Surfaces with boundary condition 1 are in perfect thermal contact with a fluid. The fluid temperature is initially 200°F but varies with time according to time function 1 given in Fig. F10.C.11. Boundary condition 2 is radiation across an air gap (region 10) between the two metals (emissivity $\epsilon = 0.8$). Conduction and natural convection are neglected. Boundary condition 3 is forced convection to a fluid at 68°F (one face of region 10 only). The heat transfer coefficient is 0.006 Btu/(min·in.^{2}·°F). Boundary condition 4 is combined heat transfer by radiation and natural convection across an air gap (region 11) between two metal surfaces (emissivity $\epsilon = 0.8$). Heat is also transferred by conduction through the air. The natural convection heat transfer coefficient is given by

$$h = 2.56 \times 10^{-5} \Delta T^{0.33} . \qquad \text{(F10.C.1)}$$

Boundary condition 5 has a time-dependent heat flux given by

$$h_f = 0.03 \cos\left[\frac{\pi}{360} t\right] , \qquad \text{(F10.C.2)}$$

and cooling by radiation and natural convection to the ambient at 100°F. The rest of the boundaries are insulated. Region 12 cannot be described for surface-to-surface radiation or natural convection because of the lack of opposing surfaces. Conduction through air could be taken into account but is neglected in this case. Therefore, the region definition is not included in the input file. The transient temperature distributions at 30 and 60 min are output, along with the steady-state temperature distribution, resulting from evaluating all time functions at 50 min. The temperatures at points (1.0,1.5) (3.75,3.0), (2.75,4.0), (5.5,4.0), and (5.5,6.75) are monitored every 10 time steps or iterations. Sixty seconds of CPU time are requested for problem execution.

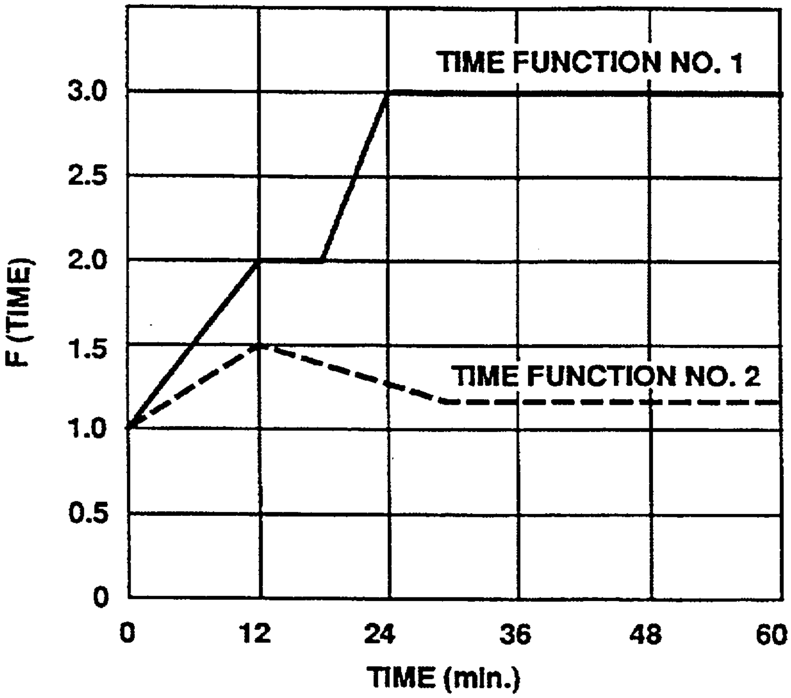


Figure F10.C.11 Time-dependent functions for sample problem 3

The input data file for this problem is given in Fig. F10.C.12. The transient calculations use the Crank-Nicolson procedure, with an initial time-step size of 0.1 min. The time-step size is then allowed to vary by keeping the maximum relative temperature change at a node to 2.5% over a time step. The steady-state solution is obtained by using the direct-solution technique.

F10.C.4 SAMPLE PROBLEM 4

Changes are made to sample problem 3 to demonstrate some additional capabilities of HEATING. The initial temperature varies as a function of y according to the following expression

$$T_o(y) = 235 - 20y \,, \qquad \text{(F10.C.3)}$$

and the heat generation rate in region 1 is a sum of exponentials defined by the expression

$$Q_1(t) = \sum_{i=1}^{3} C_{i1} e^{-\lambda_{1i} t} \,, \qquad \text{(F10.C.4)}$$

```
Sample Problem 3                                                          title
60 7 0.0                                                                    p
regions
1 1 1.0 2.0 1.5 6.75                                                        r1
1 1  0   0   1                                                              r2
2 1 2.0 5.5 4.75 6.75                                                       r1
1 1  0   5                                                                  r2
3 1 3.25 3.75 1.5 2.25                                                      r1
1 0  0    3                                                                 r2
4 1 3.25 3.75 2.25 3.0                                                      r1
1                                                                           r2
5 1 3.25 4.5 3.0 3.5                                                        r1
1                                                                           r2
6 1 4.5 5.5 3.0 3.5                                                         r1
1 0  0   0   3                                                              r2
7 2 2.0 2.75 1.5 4.0                                                        r1
1 0  0   3    1                                                             r2
8 2 2.0 2.75 4.0 4.75                                                       r1
1                                                                           r2
9 2 2.75 5.5 4.0 4.75                                                       r1
1 0  0    5                                                                 r2
10 0 2.75 3.25 1.5 3.5                                                      r1
1 0   2    2                                                                r2
11 3 3.25 5.5 3.5 4.0                                                       r1
1 0   0    0   4   4                                                        r2
materials
1 iron 0 0.2801 0.116 -3                                                     m
2 stainlss 0 0.2824 0.11 -4                                                  m
3 air 0 0 0.25 -5 -6                                                         m
initial temperatures
1 100.0                                                                      i
heat generations
1 1.0 -2                                                                     g
boundary conditions
1 2 200.0 -1                                                                b1
                                                                            b2
2 3                                                                         b1
0 1.58d-13                                                                  b2
3 1 68.0                                                                    b1
6.0d-3                                                                      b2
4 3                                                                         b1
0 1.58d-13 2.56d-05 0.33                                                    b2
5 1 100.0                                                                   b1
0 1.58d-13 2.56d-5 0.33 0 1                                                 b2
0 0 0 0 1                                                                   b3
xgrid
1.0 2.0 2.75 3.25 3.75 4.5 5.5                                              l1
   2   1    1    1    1   1                                                 n1
ygrid
1.5 2.25 3.0 3.5 4.0 4.75 6.75                                              l2
   1    1   1   1   1    4                                                  n2
analytical functions
1                                                                           a1
4 0.03 5 0.0087266                                                          a2
tabular functions
1                                                                           t1
0.0 1.0, 12.0 2.0, 18.0 2.0, 24.0 3.0                                       t2
```

Figure F10.C.12 Input data for sample problem 3

```
2                                                                            t1
0.0 1.0, 12.0 1.5, 30.0 1.125                                                t2
3                                                                            t1
0.0 0.0296, 752.0 0.0264, 1832.0 0.0222                                      t2
4                                                                            t1
0.0 0.013, 752.0 0.0153, 1832.0 0.025                                        t2
5                                                                            t1
0.0 1.82d-5, 500.0 3.41d-5, 1000.0 4.68d-5, 1500.0 5.75d-5                   t2
6                                                                            t1
0.0 5.0d-5, 500.0 2.39d-5, 1000.0 1.57d-5, 1500.0 1.17d-5                    t2
printout times
30.0 60.0                                                                    o
nodes monitored
10 5 1 18 32 36 76                                                           s
transient
2 60.0                                                                       tp
                                                                             tr2
0.1 0 5 0 0 2.5                                                              tr3
steady state
2 20                                                                         ss
%
```

Figure F10.C.12 (continued)

where the parameters are defined as

i	C_{i1}	λ_{1i}
1	0.5	0.0115525
2	0.3	0.0231049
3	0.2	0.0462098 ;

the heat generation rate in region 2 is a sum of exponentials defined by the expression

$$Q_2(t) = \sum_{i=1}^{2} C_{i2} e^{-\lambda_{2i} t} , \qquad \text{(F10.C.5)}$$

where the parameters are defined as

i	C_{i2}	λ_{2i}
1	0.6	0.0115525
2	0.4	0.0462098 .

Furthermore, the thermal conductivity for iron is assumed to be anisotropic, with the conductivity along the *y*-axis equal to twice that along the *x*-axis, as presented in Table F10.C.1. The initial temperature is input to the code as an analytical function, but the two heat generation rates and the conductivity for iron have to be defined by user-supplied subroutines. The input data are presented in Fig. F10.C.13. Tabular function numbers 2 and 3 are part of the input data but are not used. The user-supplied subroutines for the heat generation rate and the thermal conductivity for iron are presented in Fig. F10.C.14 and F10.C.15, respectively. These subroutines are stored in the file *heat4.f.* The printed output for this sample problem is not presented in this report.

```
sample problem 4                                         title
60 7 0.0                                                   p
regions
1 1 1.0 2.0 1.5 6.75                                       r1
1 1  0   0   1                                             r2
2 1 2.0 5.5 4.75 6.75                                      r1
1 1  0   5                                                 r2
3 1 3.25 3.75 1.5 2.25                                     r1
1 0  0    3                                                r2
4 1 3.25 3.75 2.25 3.0                                     r1
1                                                          r2
5 1 3.25 4.5 3.0 3.5                                       r1
1                                                          r2
6 1 4.5 5.5 3.0 3.5                                        r1
1 0  0   0   3                                             r2
7 2 2.0 2.75 1.5 4.0                                       r1
1 0  0   3    1                                            r2
8 2 2.0 2.75 4.0 4.75                                      r1
1                                                          r2
9 2 2.75 5.5 4.0 4.75                                      r1
1 0  0    5                                                r2
10 0 2.75 3.25 1.5 3.5                                     r1
1 0   2    2                                               r2
11 3 3.25 5.5 3.5 4.0                                      r1
1 0   0    0   4   4                                       r2
materials
1 iron 0 0.2801 0.116  3                                    m
2 stainlss 0 0.2824 0.11 -4                                 m
3 air 0 0 0.25 -5 -6                                        m
initial temperatures
1 0 0 2                                                     i
heat generations
1 1.0 3                                                     g
boundary conditions
1 2 200.0 -1                                               b1
                                                           B2
2 3                                                        b1
0 1.58d-13                                                 b2
3 1 68.0                                                   b1
6.0d-3                                                     b2
4 3                                                        b1
0 1.58d-13 2.56d-05 0.33                                   b2
5 1 100.0                                                  b1
0 1.58d-13 2.56d-5 0.33 0 1                                b2
0 0 0 0 1                                                  b3
xgrid
1.0 2.0 2.75 3.25 3.75 4.5 5.5                             x1
   2   1    1    1    1   1                                x2
ygrid
1.5 2.25 3.0 3.5 4.0 4.75 6.75                             y1
   1    1   1   1   1    4                                 y2
analytical functions
1                                                          a1
4 0.03 5 0.0087266                                         a2
2                                                          a1
1 235.0 2 -20.0                                            a2
3                                                          A1
                                                           A2
```

Figure F10.C.13 Input data for sample problem 4

```
tabular functions
1                                                                            t1
0.0 1.0, 12.0 2.0, 18.0 2.0, 24.0 3.0                                        t2
2                                                                            t1
0.0 1.0, 12.0 1.5, 30.0 1.125                                                t2
3                                                                            t1
0.0 0.0296, 752.0 0.0264, 1832.0 0.0222                                      t2
4                                                                            t1
0.0 0.013, 752.0 0.0153, 1832.0 0.025                                        t2
5                                                                            t1
0.0 1.82d-5, 500.0 3.41d-5, 1000.0 4.68d-5, 1500.0 5.75d-5                   t2
6                                                                            t1
0.0 5.0d-5, 500.0 2.39d-5, 1000.0 1.57d-5, 1500.0 1.17d-5                    t2
printout times
30.0 60.0                                                                    o
nodes monitored
10 5 1 18 32 36 76                                                           s
transient
2 60.0                                                                       tr
                                                                             Tr2
0.1 0 5 0 0 2.5                                                              tr3
steady state
2 20                                                                         ss
%
```

Figure F10.C.13 (continued)

```
      subroutine heatgn(rvalue,r,th,z,tim,tsn,value,number,n,arg,val,
     . ntbprs,ntab,hival,loval)
c **********************************************************************
c       this user-supplied subroutine calculates the heat generation
c       rate for heat generation functions 1 and 2 for sample problem
c       number 4 in the heating manual.
c **********************************************************************
      implicit double precision (a-h,o-z)
      common /iounit/  iecho , ihstry, imatlb, in    , io    , iplot ,
     .                 iplot0, icnvrg, iconn
      dimension        arg(1) , val(1)
      integer          ntbprs(1), ntab(1)
      logical          hival(1), loval(1)
      dimension        c1(3) , xlmda1(3), c2(2), xlmda2(2)
      data  c1         /0.5d0,0.3d0,0.2d0/
      data  xlmda1     /1.15525d-2,2.31049d-2,4.62098d-2/
      data  c2         /0.6d0,0.4d0/
      data  xlmda2     /1.15525d-2,4.62098d-2/
c **********************************************************************
      rvalue = 0.0d0
      if(number.eq.1) then
        do 10 i=1,3
          rvalue = rvalue+c1(i)*dexp(-xlmda1(i)*tim)
   10   continue
      elseif(number.eq.2) then
        do 20 i=1,2
          rvalue = rvalue+c2(i)*dexp(-xlmda2(i)*tim)
   20   continue
      else
        write(io,1000) number
        stop
      endif
      return
 1000 format('0****** user supplied subroutine heatgn has been called to
     . evaluate the'/' ****** heat generation rate for heat generation f
     .unction number',i5,'.'/' ****** this function is not defined here,
     . so the calculations will be'/' ****** terminated.')
      end
```

Figure F10.C.14 User-supplied subroutine HEATGN for sample problem 4

```
      subroutine condtn(rvalue,r1,th1,z1,r2,th2,z2,tim,tsn,tn1,tn2,
     . value,number,n,n2,arg,val,ntbprs,ntab,hival,loval)
c **********************************************************************
c       this user-supplied subroutine calculates the anisotropic,
c       temperature-dependent conductivity for material 1.  the
c       temperature-dependent conductivity in the x-direction is
c       given by the third tabular function in the input.  the
c       conductivity in the y-direction is twice that in the
c       x-direction.  used with sample problem 4 in heating manual.
c **********************************************************************
      implicit double precision (a-h,o-z)
      common /iounit/  iecho , ihstry, imatlb, in    , io    , iplot ,
     .                 iplot0, icnvrg, iconn
      dimension        arg(1) , val(1)
      integer          ntbprs(1), ntab(1)
      logical          hival(1), loval(1)
c **********************************************************************
      if(number.eq.1) then
        call table(arg,val,ntbprs,ntab,hival,loval,3,tsn,rvalue)
        if(th1.ne.th2) rvalue = 2.0d0*rvalue
      else
        write(io,1000) number
        stop
      endif
      return
 1000 format('0****** user supplied subroutine condtn has been called to
     . evaluate the'/' ****** conductivity for material number',i5,'.
     .the conductivity for this'/' ****** material is not defined here,
     .so the calculations will be'/' ****** terminated.')
      end
```

Figure F10.C.15 User-supplied subroutine CONDTN for sample problem 4

F10.D POSTPROCESSING

The normal printed output from a HEATING run only provides a summary of the solution since HEATING places an emphasis on postprocessing as the primary means of examining the results from an analysis. One computer program available for graphical postprocessing of HEATING analyses is TECPLOT.[1] Additionally, two computer programs are available to produce tabular output— H7MAP and H7MONITOR. All of these approaches make use of the data stored in a plot data file produced by every execution of HEATING. The user may also write their own program that accesses HEATING plot data files to produce customized output. The procedure for reading this file is explained in Sect. F10.5.3.

F10.D.1 H7TECPLOT/TECPLOT

TECPLOT is a commercially available interactive plotting program for visualizing engineering and scientific data. It can produce XY plots, mesh plots, contour plots, and vector plots. TECPLOT cannot read the unformatted HEATING plot data file directly. The user must first run the HEATING-to-TECPLOT interface code H7TECPLOT to produce formatted data files for TECPLOT. TECPLOT is available only on specific computers for which a license has been purchased from Amtec Engineering, Inc. The command line for executing H7TECPLOT is

```
h7tec
```

The user is prompted for the name of an existing HEATING plot data file, the name for a TECPLOT ASCII input file to be created, and the selection of information to be stored in the TECPLOT ASCII file (temperature-time plots, temperature-distance plots, or contour plots).

F10.D.2 H7MAP

H7MAP reads the plot data file generated by HEATING and produces node number and/or temperature maps similar to those generated during execution by earlier versions of HEATING (those versions prior to 7.0). The command line for executing H7MAP is

```
h7map
```

The user is prompted for the name of an existing HEATING plot data file, the name for the output file to be created, and the selection of information to be written to the output file (node-number maps and/or nodal-temperature maps and output planes for three-dimensional problems).

F10.D.3 H7MONITOR

H7MONITOR reads the plot data file generated by HEATING and produces a table of temperature vs time for selected nodes. This is a postprocessor version of the type of output that can be obtained during execution by using the NODES MONITORED data block although the output is formatted differently. H7MONITOR can only obtain the nodal temperatures at printout times stored in the plot file, whereas use of the NODES MONITORED data block allows more frequent monitoring. The command line for executing H7MONITOR is

```
h7mon
```

The user is prompted for the name of an existing HEATING plot data file, the name for the output file to be created, and the nodes whose temperatures are to be written to the output file.

F10.D.4 USER-CREATED TABULAR OUTPUT

Sufficient information is found in the plot data file generated by HEATING to produce various other types of tabular output. Users can, of course, produce their own customized tabular output. Sect. F10.5.3 supplies information about the HEATING plot file for this purpose.

REFERENCES

1. *TECPLOT - Version 5 Users Manual*, Amtec Engineering, Inc., Bellevue, Wash., 1992.

NUREG/CR-0200
Revision 6
Volume 2, Section F11
ORNL/NUREG/CSD-2/R6

Computational Physics and Engineering Division

KENO V.a: AN IMPROVED MONTE CARLO CRITICALITY PROGRAM WITH SUPERGROUPING

L. M. Petrie
N. F. Landers*

Date Published: March 2000

Prepared for the
Office of Nuclear Material Safety and Safeguards
U.S. Nuclear Regulatory Commission
Washington, DC 20555
Under Interagency Agreement DOE 1886-C000-9B
NRC JCN No. B0009

Prepared by the
OAK RIDGE NATIONAL LABORATORY
managed by
LOCKHEED MARTIN ENERGY RESEARCH CORP.
for the
U.S. DEPARTMENT OF ENERGY
under contract DE-AC05-96OR22464

*Formerly with Oak Ridge National Laboratory.

ABSTRACT

KENO V.a is an extension of the KENO V Monte Carlo criticality program and was developed for use in the SCALE system. In addition to the features available in KENO V, KENO V.a offers versatile new geometry capabilities and in-line printer plots of the geometry. The new geometry features include (1) the array of arrays option, (2) the holes option, and (3) variable chords for hemicylinders and hemispheres.

The array of arrays option allows arrays to be built of other arrays and nested to any depth, subject to the availability of sufficient computer memory. The holes option allows the placement of one or more geometry regions within other geometry regions. The depth of hole nesting is limited only by the availability of sufficient computer memory. The variable chords option allows a hemisphere or hemicylinder to be as small as non-existent or as large as a full sphere or cylinder, or any size in between.

The primary purpose of KENO V.a is to determine k-effective. Other calculated quantities include lifetime and generation time, energy-dependent leakages, energy- and region-dependent absorptions, fissions, fluxes, and fission densities.

KENO V.a retains the KENO V features such as flexible data input, the ability to specify origins for cylindrical and spherical geometry regions, the capability of supergrouping energy-dependent data, a P_n scattering model in the cross sections, a procedure for matching lethargy boundaries between albedos and cross sections to extend the usefulness of the albedo feature, and improved restart capabilities.

This advanced user-oriented program features simplified data input and efficient use of computer storage. This allows the user to readily solve large problems whose computer storage requirements and geometric complexity precluded solution when using older versions of KENO.

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

LIST OF FIGURES

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

LIST OF TABLES

LIST OF TABLES (continued)

ACKNOWLEDGMENTS

Many individuals have contributed significantly to the development of KENO V.a. Special recognition is given to G. E. Whitesides, formerly Director of the Computing Applications Division, who was responsible for the concept and development of the original KENO code. He has contributed significantly to some of the techniques utilized in KENO V.a. S. K. Fraley, formerly of the Oak Ridge National Laboratory (ORNL), is acknowledged for his assistance in implementing the mechanics of supergrouping. J. T. Thomas, retired from ORNL, has contributed many ideas that have been implemented in the code. R. M. Westfall has provided consultation, encouragement, and benchmarks for validating the code. The special abilities of J. R. Knight, retired from ORNL, contributed substantially to debugging the code. Appreciation is expressed to C. V. Parks and J. C. Turner for their assistance in reviewing this document. P. B. Fox provided the color plots. The contributions of L. F. Norris (ret.) and W. C. Carter in preparing this document are gratefully acknowledged.

Special appreciation is expressed to J. C. White of Y-12 and W. R. Golliher of K-25 for providing funding for the development of the "array of arrays" and "holes" options which have significantly enhanced the geometric capabilities of the code.

F11.1 INTRODUCTION TO KENO V.a

KENO V.a, a functional module in the SCALE system, is a multigroup Monte Carlo criticality program used to calculate the k-effective of a three-dimensional (3-D) system. Special features include simplified data input, supergrouping of energy-dependent data, the ability to specify origins for spherical and cylindrical geometry regions, a P_n scattering treatment, extended use of differential albedo reflection, and an improved restart capability.

The KENO V.a data input features flexibility in the order of input. The single restriction is that the title must be entered first and the parameter data, if any, must immediately follow. A large portion of the data has been assigned default values that have been found to be adequate for many problems. This feature enables the user to run a problem with a minimum of input data.

Blocks of input data are entered in the form:

READ XXXX input data END XXXX

where XXXX is the keyword for the type of data being entered. The types of data entered include parameters, geometry region data, array definition data, biasing or weighting data, albedo boundary conditions, starting distribution information, the cross-section mixing table, extra one-dimensional (1-D) (reaction rate) cross-section IDs for special applications, and printer plot information.

A block of data can be omitted unless it is needed or desired for the problem. Within the blocks of data, most of the input is activated by using keywords to override the default values.

The geometry input is very similar to that of KENO IV, except the specification of the biasing data has been rearranged to minimize storage requirements. An important improvement is the ability to specify the origin for cylinders, hemicylinders, spheres, and hemispheres. This feature allows the use of nonconcentric cylindrical and spherical shapes and provides a great deal of freedom in positioning them. Another improvement allows for hemicylinders and hemispheres whose cut surface can be placed at any distance between the radius and the origin.

An additional geometry convenience is the availability of an alternative method for specifying the array definition (mixed-box or unit-orientation) data. This method utilizes FIDO-like options for filling the array.

The most outstanding KENO V.a geometry advancement is the addition of the "array-of-arrays" and "holes" capabilities. The array-of-arrays option allows the construction of arrays from other arrays. The depth of nesting is limited only by computer space restrictions. This option greatly simplifies the setup for arrays involving different units at different spacings. The hole option allows placing a unit or an array at any desired location within a geometry region. The emplaced unit or array cannot intersect any geometry region and must be wholly contained within a region. As many holes as will snugly fit without intersecting can be placed in a region. This option is especially useful for describing shipping casks and reflectors that have gaps or other geometrical features. Any number of holes can be described in a problem and holes can be nested to any depth.

An important feature of KENO V.a is the capability of supergrouping the energy-dependent information such as cross sections and fluxes. This automatic feature is activated when the computer storage is insufficient to hold the entire problem at once. The energy-dependent data are then broken into supergroups that are written on a direct-access device and moved in and out of memory as necessary. Thus larger problems can be run on smaller computers.

Anisotropic scattering is treated by using discrete scattering angles. The angles and associated probabilities are generated in a manner that preserves the moments of the angular scattering distribution for the selected group-to-group transfer. These moments can be derived from the coefficients of a P_n Legendre polynomial expansion. All moments through the 2n - 1 moment are preserved for n discrete scattering angles. A one-to-one correspondence exists such that n Legendre coefficients yield n moments. The cases of zero and one scattering angle are treated in a special manner. KENO V.a can recognize that the distribution is isotropic even if the user specifies multiple scattering angles, and therefore selects from a continuous isotropic distribution. If the user specifies one scattering angle, the code performs semicontinuous scattering by picking scattering angle cosines uniformly over some range between -1 and +1. The probability is zero over the rest of the range.

Differential albedos are available to simulate tracking in a reflector. These albedos were generated using the Hansen-Roach 16-energy-group structure. KENO V.a can extend the use of these albedos to include other energy group structures. This step is done by matching lethargy boundaries between the albedos and cross sections so the appropriate energy transfers can be made. Lethargy boundary tables are created for both the albedo and cross-section energy group structures, and the lethargy interval corresponding to the desired transfer is determined based on a uniform distribution over the lethargy interval. Approximations must be made when the energy group boundaries of the albedos and cross sections are different; therefore the user should scrutinize the results to evaluate the effects of the approximations until an adequate information base is established.

The KENO V.a restart option is easy to activate. Certain changes can be made when a problem is restarted, including the use of a different random sequence and turning off certain print options such as fluxes or the fissions and absorptions by region.

F11.2 THEORY AND TECHNIQUES

F11.2.1 THE TRANSPORT EQUATION

The equation KENO V.a solves may be derived in the following manner, starting with the Boltzmann neutron transport equation which may be written as[1]

$$\frac{1}{v}\frac{\partial\Phi}{\partial t}(X,E,\Omega,t) + \Omega\cdot\nabla\Phi(X,E,\Omega,t) + \Sigma_t(X,E,\Omega,t)\Phi(X,E,\Omega,t) = S(X,E,\Omega,t)$$

$$+ \int_{E'}\int_{\Omega'}\Sigma_s(X,E'\rightarrow E,\Omega'\rightarrow\Omega,t)\Phi(X,E',\Omega',t)d\Omega'dE' \ , \qquad \text{(F11.2.1)}$$

where

$\Phi(X,E,\Omega,t)$ = neutron flux (neutrons/cm^2/s) per unit energy at energy E per steradian about direction Ω at position X at time t moving at speed v corresponding to E,

$\Sigma_t(X,E,\Omega,t)$ = macroscopic total cross section of the media (cm^{-1}) at position X, energy E, direction Ω and time t,

$\Sigma_s(X,E'\rightarrow E,\Omega'\rightarrow\Omega,t)$ = macroscopic differential cross section of the media (cm^{-1}) per unit energy at energy E′ per steradian about direction Ω' at position X, and time t, for scattering to energy E and direction Ω,

$S(X,E,\Omega,t)$ = neutrons/cm^3/s born at position X and time t per unit energy at energy E per steradian about direction Ω (excludes scatter source).

Defining $q(X,E,\Omega,t)$ as the total source resulting from the external source, scattering, fission and all other contributions, the following relationship can be written.

$$q(X,E,\Omega,t) = S(X,E,\Omega,t) + \int_{E'}\int_{\Omega'}\Sigma_s(X,E'\rightarrow E,\Omega'\rightarrow\Omega,t)\Phi(X,E',\Omega \qquad \text{(F11.2.2)}$$

Combining Eqs. (F11.2.1) and (F11.2.2), assuming the media to be isotropic, ignoring the time dependence of the cross sections and converting the equation to multigroup form yields

$$\frac{1}{v_g}\frac{\partial\Phi_g}{\partial t}(X,\Omega,t) + \Omega\cdot\nabla\Phi_g(X,\Omega,t) + \Sigma_{t_g}(X)\Phi_g(X,\Omega,t) = q_g(X,\Omega,t) \ , \qquad \text{(F11.2.3)}$$

where

g is the energy group of interest,

v_g is the average velocity of the neutrons in group g,

$\Phi_g(X,\Omega,t)$ is the angular flux of neutrons having their energies in group g, at position X and time t,

$\Sigma_{tg}(X)$ is the macroscopic total cross section of the media at position X for group g, corresponding to

$$\Sigma_{tg}(X) = \frac{\int_{\Delta E_g} \Sigma_t(X,E)\Phi(X,E,\Omega,t)dE}{\int_{\Delta E_g} \Phi(X,E,\Omega,t)dE} ,$$

where ΔE_g defines group g, and

$q_g(X,\Omega,t)$ is the total source contributing to energy group g at position X, and time t in direction Ω.

Utilizing the relationship $X' = X - R\Omega$, defining the problem to be time-independent, using an integrating factor[2] on both sides of Eq. (F11.2.3), and defining

$$T(R) = \int_O^R \Sigma_{t_g}(X-R'\Omega)dR' ,$$

the following equation can be written.

$$\Phi_g(X,\Omega) = \int_O^\infty q_g(X-R\Omega,\Omega)e^{-T(R)}dR . \qquad \text{(F11.2.4)}$$

At this point, the problem becomes an eigenvalue problem. If there is no external source, the source may be defined as

$$q_g(X,\Omega) = \sum_{g'} \int d\Omega' \Phi_{g'}(X,\Omega')\Sigma_s(X,g' \rightarrow g,\Omega' \cdot \Omega) + \frac{1}{k}Q'_g(X,\Omega) , \qquad \text{(F11.2.5)}$$

where

k is the largest eigenvalue of the integral equation,

$Q'_g(X,\Omega)$ is the fission source at position X for energy group g and direction Ω (all fission contributions to group g from all energy groups in the previous generation),

$\Sigma_s(X,g' \rightarrow g,\Omega' \cdot \Omega)$ is the scattering cross section for scattering at position X from group g′ and direction Ω' to group g and direction Ω.

In terms of energy, the scatter can be defined as

$$\Sigma_s(X,g' \rightarrow g,\Omega' \cdot \Omega) = \frac{\int_{\Delta E_g} \int_{\Delta E_{g'}} \Sigma_s(X,E' \rightarrow E,\Omega' \cdot \Omega)\Phi(X,E',\Omega')dE'}{\int_{\Delta E_{g'}} \Phi(X,E',\Omega')dE'} \qquad \text{(F11.2.6)}$$

where

ΔE_g is the energy-range-defining energy group g, and

$\Delta E_{g'}$ is the energy-range-defining energy group g′.

Assuming the fission neutrons to be isotropic, the fission source $Q_{g'}(X,\Omega)$ can be written as

$$Q'_g(X,\Omega) = \frac{1}{4\pi}\sum_{g'}\int_{\Omega'} d\Omega'\Phi_{g'}(X,\Omega')\chi(X,g'\to g)\upsilon_{g'}(X)\Sigma_{fg'}(X) \ , \tag{F11.2.7}$$

where

$\chi(X,g'\to g)$ is the fraction of neutrons born in energy group g from fission in energy group g′ in the media at position X,

$\upsilon_{g'}(X)$ is the number of neutrons resulting from a fission in group g′ at position X,

$\Sigma_{fg'}(X)$ is the macroscopic fission cross section of the material at position X for a neutron in energy group g′.

Substituting Eq. (F11.2.5) into Eq. (F11.2.4) yields the following equation:

$$\Phi_g(X,\Omega) = \int_0^\infty dRe^{-T(R)}$$

$$\left\{\frac{1}{k}Q'_g(X-R\Omega,\Omega) + \sum_{g'}\left[\int_{\Omega'} d\Omega'\Phi_{g'}(X-R\Omega,\Omega')\Sigma_s(X-R\Omega,g'\to g,\Omega'\cdot\Omega)\right]\right\}. \tag{F11.2.8}$$

The definition of k may be given as the ratio of the number of neutrons in the (n + 1)*th* generation to the number of neutrons in the n*th* generation or the largest eigenvalue of the integral equation. Using Eq. (F11.2.7), Eq. (F11.2.8) can be written as

$$\Phi_g(X,\Omega) = \int_0^\infty dRe^{-T(R)} \ \sum_{g'}\frac{1}{k}\int_{\Omega'}\upsilon_{g'}(X-R\Omega)\Sigma_{fg'}(X-R\Omega)\chi(X-R\Omega,g'\to g)\Phi_{g'}(X-R\Omega,\Omega')\frac{d\Omega'}{4\pi}$$

$$+ \sum_{g'}\int_{\Omega'} d\Omega'\Sigma_{tg'}(X-R\Omega,\Omega')\frac{\Sigma_s(X-R\Omega,g'\to g,\Omega'\cdot\Omega)}{\Sigma_{tg'}(X-R\Omega)} \quad . \tag{F11.2.9}$$

Writing Eq. (F11.2.9) in "generation notation," multiplying and dividing certain terms by $\Sigma_{t_g}(X)$ and multiplying both sides of the equation by $\upsilon_g(X)\Sigma_{f_g}(X)$, yield the following equation, which is solved by KENO V.a:

$$\frac{\upsilon_g(X)\Sigma_{fg}(X)}{\Sigma_{tg}(X)}\Sigma_{tg}(X)\Phi_{g,n}(X,\Omega) = \frac{\upsilon_g(X)\Sigma_{fg}(X)}{\Sigma_{tg}(X)}\Sigma_{tg}(X)\int_O^\infty dRe^{-T(R)}$$

$$\left\{\frac{1}{k}\sum_{g'}\int_{\Omega'}\frac{\upsilon_{g'}(X-R\Omega)\Sigma_{fg'}(X-R\Omega)}{\Sigma_{tg'}(X-R\Omega)}\chi(X-R\Omega,g'\rightarrow g)\Sigma_{tg'}(X-R\Omega)\phi_{g',n-1}(X-R\Omega,\Omega')\frac{d\Omega'}{4\pi}\right.$$

$$\left.+\sum_{g'}\int_{\Omega'}\frac{\Sigma_s(X-R\Omega,g'\rightarrow g,\Omega'\cdot\Omega)}{\Sigma_{tg'}(X-R\Omega)}\Sigma_{tg'}(X-R\Omega)\Phi_{g',n}(X-R\Omega,\Omega')d\Omega'\right\} \quad , \qquad \text{(F11.2.10)}$$

where n indicates the n*th* generation and n - 1 is the (n - 1)*th* generation. Note that the left-hand side of the equation, $\upsilon_g(X)\ \Sigma_{fg}(X)\Phi_{g,n}(X,\Omega)$ is the fission production for the n*th* generation.

The solution strategy utilized by KENO V.a solves Eq. (F11.2.10) by using an iterative procedure. The fission production at point X in energy group g due to neutrons in the (n - 1)*th* generation, normalized to the system multiplication, is

$$\frac{1}{k}\sum_{g'}\int_{\Omega'}\frac{\upsilon_{g'}(X)\Sigma_{fg'}(X)}{\Sigma_{tg'}(X)}\chi(X,g'\rightarrow g)\Sigma_{tg'}(X)\Phi_{g',n-1}(X,\Omega')\frac{d\Omega'}{4\pi} \ .$$

The collision points used in KENO V.a are chosen by selecting path lengths from the distribution

$$e^{-T(R)}$$

which is the probability of transport from any position X – RΩ to position X.

The first collision density of neutrons in group g per unit solid angle about Ω resulting from the fission source produced by the (n - 1) generation, normalized to the system multiplication, is

$$\Sigma_{tg}(X)\int_0^\infty dRe^{-T(R)}\frac{1}{k}\int_{\Omega'}\sum_{g'}\frac{\upsilon_{g'}(X-R\Omega)\Sigma_{fg'}(X-R\Omega)}{\Sigma_{tg'}(X-R\Omega)}$$

$$\chi(X-R\Omega,g'\rightarrow g)\Sigma_{t_{g'}}(X-R\Omega)\Phi_{g',n-1}(X-R\Omega,\Omega')\frac{d\Omega'}{4\pi} \ .$$

The scattering source at position X emerging in group g and direction Ω resulting from previous collisions in the same generation, is

$$\sum_{g'}\int_{\Omega'}\frac{\Sigma_s(X,g'\rightarrow g,\Omega'\cdot\Omega)}{\Sigma_{tg'}(X)}\Sigma_{t_{g'}}(X)\,\Phi_{g',n}(X,\Omega')d\Omega' \ .$$

The collision density in group g, per solid angle about Ω is

$$\Sigma_{tg}\int_0^\infty dRe^{-T(R)}\sum_{g'}\int_{\Omega'}\frac{\Sigma_s(X-R\Omega,g'\rightarrow g,\Omega'\cdot\Omega)}{\Sigma_{tg'}(X-R\Omega)}\Sigma_{tg'}(X-R\Omega)\Phi_{g',n}(X-R\Omega,\Omega')d\Omega' \ .$$

The total collision density times $\frac{\upsilon_g(X)\Sigma_{fg}(X)}{\Sigma_{tg}(X)}$ is the relationship from which KENO V.a picks the source points for the next generation.

F11.2.2 COLLISION TREATMENT

A collision occurs in a geometrical region when a history exhausts its mean-free-path length within the boundaries of the region. For each collision, the absorbed weight and the fission weight are tabulated, then the weight is modified by the nonabsorption probability. This new weight is checked for splitting and Russian roulette, and if it survives, the history is scattered. A new energy group is selected from the cumulative transfer probability distribution. This group-to-group transfer determines an angular scattering distribution, usually expressed as a Legendre expansion of the cross-section transfer array. A set of discrete angles and probabilities are generated by a generalized Gaussian quadrature procedure,[3] preserving the moments of the Legendre expansion of the angular scattering distribution. KENO V.a treats P_0 and P_1 Legendre expansions as special cases. If the scattering distribution is isotropic, a flag is set to randomly select new direction cosines from an isotropic distribution, instead of using discrete scattering angles. If the distribution is a P_1 expansion, KENO V.a randomly selects the cosine of the scattering angle according to

(1) $|\bar{\mu}| < \frac{10^{-10}}{3}$: scattering distribution is isotropic,

(2) $|\bar{\mu}| \leq 1/3: \quad \mu = (\sqrt{1 + 6\zeta\bar{\mu} + (3\bar{\mu})^2} - 1)/3\bar{\mu},$

or

(3) $|\bar{\mu}| > 1/3: \quad \mu = \zeta(1 - |\bar{\mu}|) + \bar{\mu},$

where ζ is a uniform random variable between -1 and +1
and $\bar{\mu}$ is the mean cosine of the scattering angle.

Otherwise, KENO V.a randomly selects one of the discrete scattering angles (μ). New direction cosines are then calculated according to the following relationships where u, v, and w are the initial direction cosines and u′, v′, and w′ are the direction cosines after the collision:

$$u' = u\cos\psi - \sqrt{v^2+w^2}\sin\psi\cos\eta$$

$$v' = v\cos\psi + \frac{uv}{\sqrt{v^2+w^2}}\cos\eta\sin\psi - \frac{w}{\sqrt{v^2+w^2}}\sin\psi\sin\eta \ ,$$

$$w' = w\cos\psi + \frac{uw}{\sqrt{v^2+w^2}}\cos\eta\sin\psi + \frac{v}{\sqrt{v^2+w^2}}\sin\psi\sin\eta \ ,$$

where

$\sin\psi = \sqrt{1-\mu^2}$,

$\cos\psi = \mu$ = cosine of the scattering angle,

η = a random azimuthal angle between 0 and 2π.

F11.2.3 FISSION POINT SELECTION

In order for a fission to occur, a neutron must first have a collision. The fission weight, FISW, is defined as the neutron weight, WT, times the ν-fission probability, FNFP(KR,IG) of the material in which the collision occurred, at the energy of the incident neutron, IG.

FISW = WT*FNFP(KR,IG)

Two important variables that are used in the processing of fission points are FWR, which is defined as the fission weight, FISW, divided by a random number, and RAKBAR, which is defined as a factor times the running average value of k-effective, AKBAR. This factor is a function of the square root of the number of neutrons per generation and was chosen because it usually produces an adequate number of independent fission points and does not produce so many that an excessive amount of time is spent choosing from the fission points that are produced.

The following procedure for generating fission points is repeated until FWR is less than RAKBAR. A fission point is generated only if FWR is greater than RAKBAR. Multiple fissions at the same point are allowed only if FISW is greater than RAKBAR. If FISW is greater than RAKBAR, a fission point is stored with FWR set equal to RAKBAR divided by a random number and FISW is decremented by RAKBAR. Then the energy group of fission is chosen randomly from the fission spectrum of the mixture in which the fission occurred. The energy group of fission, the x, y, and z position, the location of the unit within the array, the region number, the value of FWR, the region number of the array boundary, and the nesting data for holes and/or arrays are stored in the fission bank. Only the number of positions in the fission bank (input parameter NFB=) fission points are kept to be used as fission positions for the next generation. Typically NFB is equal to the input parameter NPG, the number of neutrons per generation. If a fission occurs and the fission counter is less than NFB, the fission point information is stored in the fission bank. If a fission occurs and the fission counter is greater than or equal to the number per generation, a search is made to find the smallest stored value of FWR. If FWR of the newly fissioned neutron is less than the smallest FWR in the table, it is discarded. Otherwise, the information from the newly fissioned neutron replaces that associated with the smallest value of FWR found in the table.

When the next generation is ready to be processed, data are transferred from the fission bank into the neutron bank to be used as starting positions for the fission neutrons. If more than NPG neutrons are saved in the fission bank, NPG of those having the highest values of FWR will be used. If too few fission positions were stored (less than the number per generation), a warning message to that effect (K5-132) is printed, and additional fission points are randomly chosen from those that were stored until the number of fission points available to start the next generation is equal to the number of neutrons per generation.

F11.2.4 BIASING OR WEIGHTING

In order to minimize the statistical deviation of k-effective per unit computer time, KENO V.a utilizes weighted tracking rather than analog tracking. Weighted tracking accounts for absorption by reducing the neutron weight, rather than allowing the neutron history to be terminated by absorption. To prevent expending excessive computer time tracking low-weight neutrons, Russian roulette is played when the weight of the neutron drops below a preset weight, WTLOW. Neutrons that survive Russian roulette are assigned a weight, WTAVG. The value of WTLOW and WTAVG can be assigned as a function of position and energy. The values used by KENO V.a are

DWTAV = 0.5	is the default value of WTAVG,
WTAVG = DWTAV	is the weight given a neutron that survives Russian roulette, and
WTLOW = WTAVG/3.0	is the value of weight at which Russian roulette is played.

A study[4] by Hoffman shows these default values to be reasonable for bare critical assemblies. Figure F11.2.1, from this study, shows the analytic relationship between the variance and WTLOW when WTAVG is 0.5. Note that the default value of 0.167 for WTLOW is very close to the minimum point on the curve. Experimental results of actual Monte Carlo calculations[5] provide further assurance that 0.167 is an optimum choice for WTLOW when WTAVG is 0.5.

Figure F11.2.2, also from the Hoffman study, shows the analytic relationship between the variance and the value chosen for WTAVG for a value of WTLOW = 0.167. Although the KENO V.a default value for WTAVG is not the optimum, a close examination of the data shows the variance to be changing relatively slow as a function of WTAVG. Even though this study shows a value near 0.26 to be optimum for this system, further studies of other systems are needed before changing the default value of WTAVG from the 0.5 that has been used in previous versions of KENO.

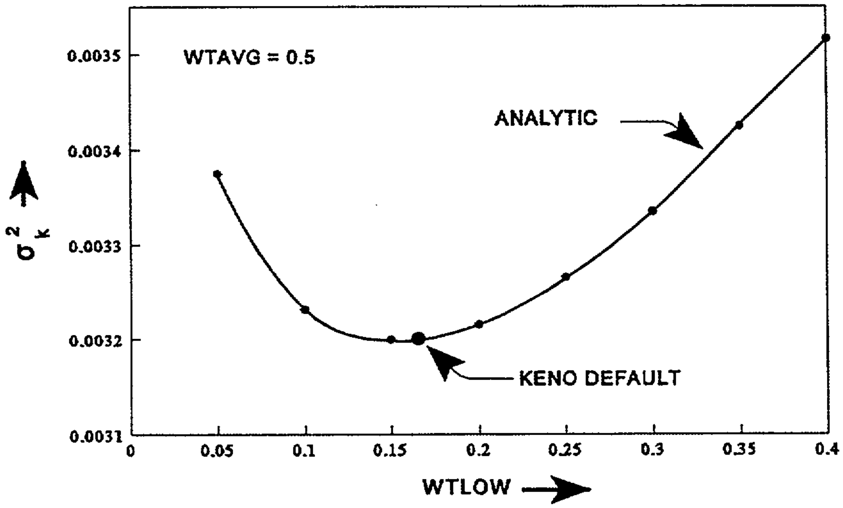


Figure F11.2.1 Analytic estimate of the relationship between WTLOW and the variance, σ_k^2, when WTAVG is 0.5

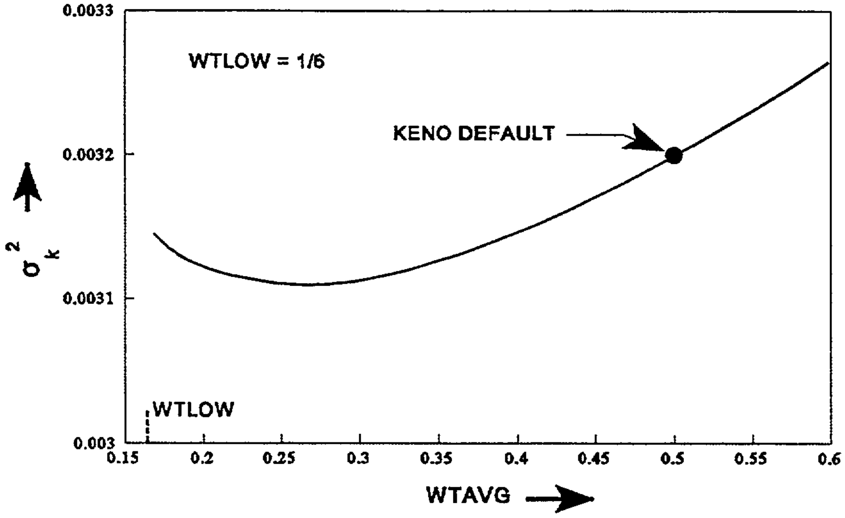


Figure F11.2.2 Analytic estimate of the relationship between WTAVG and the variance, σ_κ^2, when WTLOW is 1/6.

Inside a fissile core, the importance of a neutron is a slowly varying function in terms of energy and position. Hence, for many systems, the standard defaults for WTLOW and WTAVG are good values to use. For reflectors, however, the worth of a neutron varies both as a function of distance from the fissile material and as a function of energy. As a neutron in the reflector becomes less important relative to a neutron in the fissile region, it becomes desirable to spend less time tracking it. Therefore a space- and energy-dependent weighting or biasing function is used in KENO V.a to allow the user to minimize the variance in k-effective per unit tracking time. When a biasing function is used in a reflector, it becomes possible for a neutron to move from one importance region into another whose WTLOW is greater than the weight of the neutron. When this occurs, Russian roulette is played to reduce the number of neutrons tracked. When the reverse occurs, that is, the neutron moves to a region of higher importance, its weight may be much higher than WTAVG for that region. When the weight of the neutron is greater than a preset value, WTHI, the neutron is split into two neutrons, each having a weight equal to one-half the weight of the original neutron. This procedure is repeated until the weight of the split neutron is less than WTHI. The default value for WTHI is WTAVG*3.0. WTHI is the weight at which splitting occurs.

The weighting or biasing function for a given core material and reflector material can be obtained by using the adjoint solution from S_n type programs for a similar (usually simplified) problem. This adjoint flux gives the relative contribution of a neutron at a given energy and position to the total fissions in the system. The weighting function for KENO V.a is thus proportional to the reciprocal of the adjoint flux. Although such a function can be difficult to obtain, the savings gained makes the effort worthwhile for many of the materials that are frequently used as reflectors. Biasing functions[6] have been prepared for several reflector materials commonly used in KENO V.a calculations. The use of biasing to minimize the variance in k-effective per unit computer time will usually increase the variance in other parameters such as leakage or absorption in the reflector.

F11.2.5 DIFFERENTIAL ALBEDOS

Arrays reflected by thick layers of material having a small absorption to scattering ratio may require large amounts of computer time to determine k-effective[7] because of the relatively long time a history may spend in the reflector. A differential albedo technique was developed for use with the KENO codes to eliminate tracking in the reflector. This involves returning a history at the point it impinges on the reflector and selecting an emergent energy and polar angle from a joint density function dependent upon the incident energy and polar angle. The weight of the history is adjusted by the functional return from the reflector, which is also based on the incident energy and angle.

The characteristics of a differential albedo emulate the attributes of the reflector material and are independent of the material or materials adjacent to the reflector. Thus, a differential albedo that is generated for a given reflector material can be used with any array, regardless of the type of fuel or fissile material contained within the array.

For many calculations involving reflected arrays of fissile material, the differential albedo treatment is a powerful tool that can significantly reduce the computing time required to determine k-effective. The savings will vary, depending on the importance of the reflector to the system. A substantial effort is required to generate a differential albedo, but the savings gained are well worth the effort for commonly used reflector materials.

To generate the differential albedo information for a material, a fixed-source calculation must be made for each incident energy and angle. The data presently available for use with KENO V.a were generated by 1-D discrete-ordinates calculations for slab geometry, representing infinite slabs. Consequently, for a finite reflector, these data will not correctly treat histories that enter the reflector near an edge. Past experience with differential albedo reflectors indicates that k-effective appears to be conservative for small faces and will tend toward the correct result as the face becomes large relative to the area near the corners. Care must be taken to ensure that any surface to which a differential albedo is applied is large enough that the errors at the edges can be ignored.

Because differential albedos are expensive and time-consuming to generate, those corresponding to the Hansen-Roach 16-energy-group structure are the only differential albedos currently available for use with KENO V.a. In the past, their use was limited to problems utilizing cross sections having the Hansen-Roach 16-energy-group structure. KENO V.a extends the use of differential albedos to other energy-group structures by allowing appropriate energy transfers. This is accomplished by creating lethargy boundary tables for the albedo group structure and the cross-section group structure and determining the lethargy interval corresponding to the desired transfer (cross-section group structure to albedo group structure or vice versa) based on a uniform lethargy distribution over the interval. When the energy-group boundaries of the cross sections and albedos are different, the results should be scrutinized by the user to evaluate the effects of the approximations.

F11.2.6 SUPERGROUPING

An important feature of KENO V.a is the capability of supergrouping energy-dependent information. This includes the cross sections, albedos, pointer arrays, weights, leakages, absorptions, fissions and fluxes. If the available computer memory is too small to hold all the problem data at once, KENO V.a determines the number of supergroups necessary to allow execution of the problem. A problem cannot be supergrouped if the energy-dependent data associated with any individual energy group are too large to fit in the available memory. If enough memory is available to accommodate all the energy-dependent data at once, only one supergroup is created. Once the number of supergroups has been determined, the energy-dependent data are arranged in supergroups and are written on the direct-access supergroup file. During execution of a problem, the supergrouped data are moved in and out of memory as necessary.

The advantage of supergrouping is that larger problems can be run on smaller computers. This capability is gained at the expense of running time and increased I/Os. The more supergroups, the more I/Os are used and the slower the problem will run because of the banking, sorting, and use of direct-access devices in the solution of the problem.

In order to reduce the amount of data movement between memory and the direct-access supergroup file, KENO V.a maintains a bank of histories (the neutron bank) and follows all those histories that fall within the current supergroup before going to the next supergroup. Histories that are scattered out of the current supergroup are placed back in the bank. When all the histories in the current supergroup have been processed, the bank is sorted, placing the histories for the most populous supergroup at the top of the bank. All other histories are placed at the bottom of the bank. The data for the most populous supergroup are then brought into memory and tracking proceeds.

F11.2.7 RESTART

KENO V.a incorporates a versatile and convenient restart capability. The decision to write a restart file requires the user to specify only the number of generations between writing restart data and the unit number where the restart file is to be written. A file definition card must be included in the job control language for the restart data file. The input data are the first data written on the restart data file. The group-dependent input data are written a group at a time, which includes the cross sections, albedos, pointer arrays and weights. The number of records of input data is automatically determined by the code and written on the restart data file. After the input data have been written on the restart data file, the calculated data are written at the end of each specified generation. These data include the generation number, random number, number of histories per generation, number of energy groups, bank lengths, common information, the k-effectives by generation, the neutron bank, the fission densities, matrix arrays, and the calculated group-dependent data. These group-dependent data are written a group at a time and include leakages, absorptions, fissions and fluxes.

The KENO V.a restart capability allows a problem to be restarted at the first generation with different input because all data input supersedes data from the restart data file.

If a problem is to be restarted at a generation greater than 1, the only data that can be changed are certain parameter data. Changes in the parameter data that are not allowed include (1) requesting fissions and absorptions by region if they were not requested by the parent case, (2) requesting fission densities and fluxes if they were not requested by the parent case, (3) requesting matrix information that was not requested in the parent case, and (4) changing the configuration of the neutron bank to be different from that of the parent case.

Because restart data are written a group at a time, a problem may be restarted with an entirely different supergroup structure.

If a problem is to be restarted following a generation for which restart data were not written, the code will write a message and restart with the next available generation for which restart data exist. If no such generation is found, the problem is terminated.

F11.2.8 GEOMETRY

KENO V.a geometry is restricted to the use of specific shapes. These shapes are called geometry regions or regions. Allowed shapes are cubes, cuboids (rectangular parallelepipeds), spheres, cylinders, hemispheres, and hemicylinders. These shapes must be oriented along orthogonal axes and cannot be rotated. They can be translated. Hemispheres and hemicylinders are not limited to half-spheres and half-cylinders; the definitive plane can be positioned by entering a chord. The value of this chord can range from the positive magnitude of the radius (giving a complete sphere or cylinder) to the negative magnitude of the radius (giving a zero volume, nonexistent sphere, or cylinder).

A major restriction applied to KENO V.a geometry is that intersections are not allowed. Furthermore, each successive geometry region must completely enclose the preceding region. Tangency and shares faces are allowed. The volume of a region is the volume of the specified shape minus the volume of the preceding region shape. To alleviate the complete enclosure restriction, KENO V.a allows multiple sets of geometry regions with each set independently governed by this restriction. Each set of these multiple geometry regions are called units or box types. Units can be stacked together in a 3-D rectangular parallelepiped called an array or lattice just as children's blocks can be stacked. Units that are to be stacked together in this manner must have a rectangular parallelepiped outer region and the adjacent faces of adjacent units must be the same size and shape. An array can be treated as a building block and be used as a unit within another array.

The use of holes in KENO V.a allows a unit to be emplaced within another unit, thereby alleviating the restriction that each region within a unit must completely enclose all preceding regions within that unit. However, a hole is not allowed to intersect other holes or regions. A unit that is to be used as a hole need not have a rectangular parallelepiped as its outer boundary.

Multiple arrays can be described in KENO V.a. The global array in an unreflected problem is the outermost array in the problem geometry description. The global array in a reflected problem is the array referenced by surrounding geometry regions following the last array placement description that does not immediately follow a unit number description. See Sect. F11.4.5.

Consistent with past versions of KENO V.a, KENO V.a retains the capability of running a single unit problem. A single unit problem is one that has no array description.

F11.2.9 REFERENCES

1. Robert V. Meghreblian and David K. Holmes, *Reactor Analysis*, McGraw-Hill, 1960.

2. J. E. Powell and C. P. Wells, *Differential Equations*, Ginn Company, 1950.

3. E. A. Straker, P. N. Stevens, D. C. Irving, and V. R. Cain, *The MORSE Code – A Multigroup Neutron and Gamma-Ray Monte Carlo Transport Code, Appendix B*, ORNL-4585, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., 1970.

4. T. J. Hoffman, *The Optimization of Russian Roulette Parameters for KENO V.a*, ORNL/TM-7539, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., 1982.

5. L. M. Petrie and N. F. Cross, *KENO IV - An Improved Monte Carlo Criticality Program*, ORNL-4938, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., 1975. Also see Sect. F5 of the SCALE manual.

6. J. R. Knight and L. M. Petrie, *16 and 123 Group Weighting Functions for KENO V.a*, ORNL/TM-4660, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., 1975.

7. G. E. Whitesides and J. T. Thomas, "The Use of Differential Current Albedos in Monte Carlo Criticality Calculations," *Trans. Am. Nucl. Soc.* **12**, 889 (November 1969).

F11.3 LOGICAL PROGRAM FLOW

The general flow of the KENO V.a program during the solution of a problem is given in this section. A formal flow chart is not included because of the voluminous nature of the program. The logical program flow is broken up into small sections. The format of each section includes an abbreviated flow chart, a brief explanation of the purpose of that section of the program, and a brief description of each subroutine involved. The abbreviated flow charts are drawn with KENO V.a subroutine names contained in boxes and library routines as bare names. An arrow in the flow chart indicates that the subroutine associated with the arrow will be treated in detail later in the section.

F11.3.1 PROGRAM INITIATION

The function of this portion of the program (Fig. F11.3.1) is to initialize information, print a header page, call the parameter reading subroutine PARAM, access subroutine MASTER with the storage space allocated by subroutine ALOCAT, and close out the direct-access files when the problem is completed or terminated.

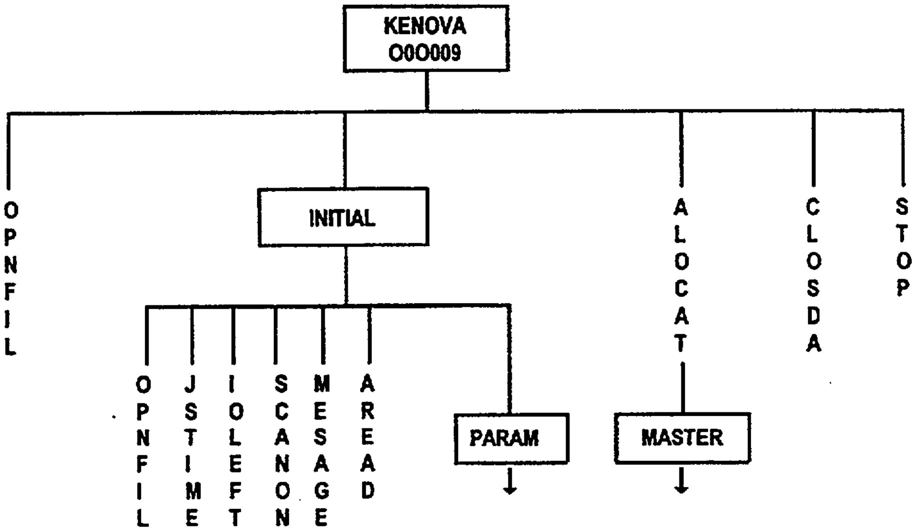


Figure F11.3.1 Flow chart of program initiation

KENOVA - O0O009 - These main programs set flags to specify the proper mode of data reading for the stand-alone KENO V.a program. INITAL is called to perform some initialization. MASTER is then called from ALOCAT with the requested storage allocation. The direct access files are closed out by CLOSDA when a problem is completed or terminated.

OPNFIL - This library routine is called to initialize the input and output logical units.

INITAL - This subroutine calls library routines to perform initialization and print a header page. It then reads the problem's title card and parameter read flag and calls subroutine PARAM to do the actual reading of the parameter data.

JSTIME - This call to the library routine JSTIME is for the purpose of storing the initial time in COMMON/FINAL/ for timing purposes.

IOLEFT - This call to the library routine IOLEFT is for the purpose of storing the initial I/O count in COMMON/FINAL/ for future calculations that indicate the number of I/Os used in certain parts of the program.

SCANON - This library routine is called to activate the feature that allows scanning for the word END when reading data.

MESAGE - This library routine is called with two arguments, an eight-character hollerith argument and an output unit. Additional library routines are called from MESAGE to print a header page in block letters. The header page includes the eight-character hollerith argument (KENO V.a-V), the date, the time execution was begun, and the job name. The library routine LISTQA is called to provide a program verification page for quality assurance purposes.

AREAD - This library routine is used to read alphanumeric data. It is used here to read the title and the parameter reading flag.

PARAM - This subroutine is called to set default values for the parameter input data and to read parameter data. See Sect. F11.3.2 for a more detailed description of PARAM.

ALOCAT - This assembly language library routine is called with three arguments. The first argument is a subroutine name and the second argument is the maximum number of words of storage to be allocated. ALOCAT calls subroutine MASTER with two arguments, an array name and the length of the array. The third argument is prefixed by an asterisk and specifies the statement number to return to if there isn't enough storage to run the problem.

MASTER - This subroutine is called by subroutine ALOCAT. It is the controlling subroutine for the bulk of the KENO V.a program flow. See Sect. F11.3.3 for a more detailed description of subroutine MASTER.

CLOSDA - This library routine is called to close out each direct-access file at the normal completion or normal termination of a problem.

STOP - This library routine is called to write a message and terminate execution when there is insufficient storage to run the problem.

F11.3.2 PARAMETER DATA

The function of this section of the program is to set default values for the parameters and to read the parameter input data (Fig. F11.3.2).

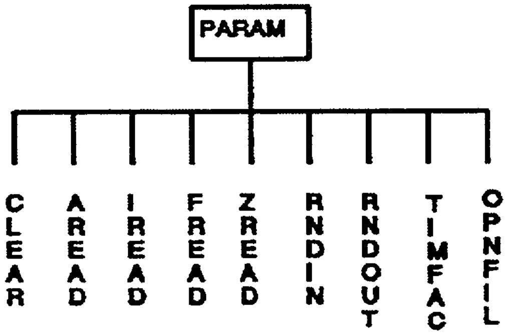


Figure F11.3.2 Flow chart of parameter reading

PARAM - This subroutine is responsible for reading the parameter input data and setting default values for the parameters. The library routine CLEAR is used to zero parameters that are defaulted to zero. The parameter data block is read using AREAD, IREAD, FREAD and ZREAD. Each entry in the parameter data block uses a keyword so the code can store the parameter data in the correct location. If the problem is a restart problem, restart information including the title of the original case, parameter data and some common information is read from the restart data file. The defaulted and input parameter tables are printed by PARAM. Some preliminary data checking is done, and appropriate warning and error messages may be printed.

CLEAR - This library routine is called to zero the parameters that are defaulted to zero.

AREAD - This library routine can be called many times from subroutine PARAM. It is used to read parameter names and alphanumeric parameter data.

IREAD - This library routine can be called many times from subroutine PARAM. It is used to read integer parameter data.

FREAD - This library routine can be called many times from subroutine PARAM. It is used to read floating-point data.

ZREAD - This library routine can be called from subroutine PARAM to read a hexadecimal random number that will be used as a kernel for the random-number package.

RNDIN - This library routine is called from subroutine PARAM to transfer the random number read by ZREAD to the random-number package. It is called only if a random number was entered as parameter data.

RNDOUT - This library routine is called from subroutine PARAM to preserve the current random number so it can be written on the restart data set and printed in the parameter data.

TIMFAC - This library routine is called from subroutine PARAM to provide the proper adjustment factor by which the allotted time is multiplied. This factor adjusts the execution time for execution on different computers.

OPNFIL - This library routine is called to initialize the logical unit for the restart data file.

F11.3.3 OVERALL PROGRAM FLOW

The purpose of this section of the program is to direct the primary flow (Fig. F11.3.3) of KENO V.a. This primary flow covers the complete scope of the program from data reading to editing and printing the results.

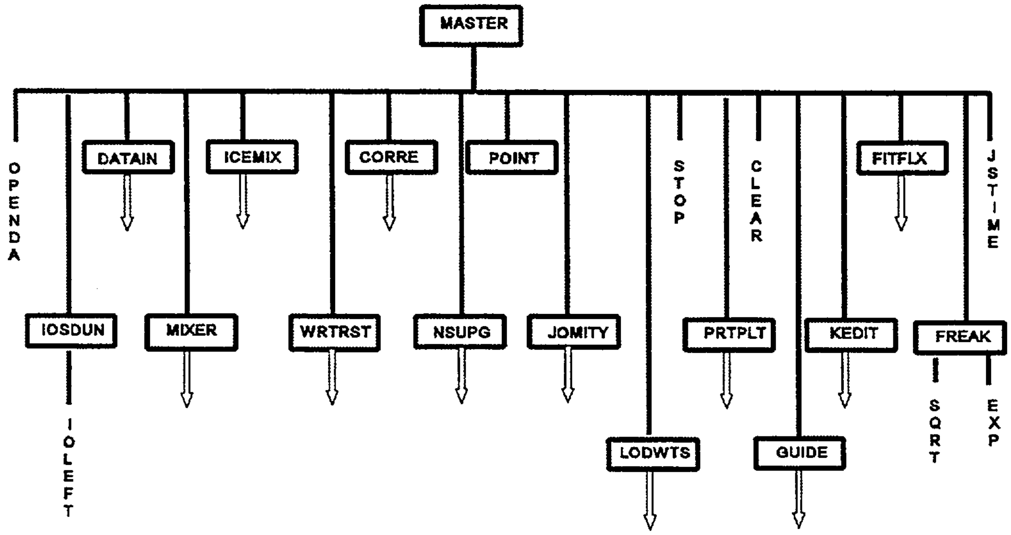


Figure F11.3.3 Flow chart of overall program flow

MASTER - This subroutine controls the primary flow of KENO V.a. It initializes the direct-access files and calls subroutines to read, check and print the input data, calls the tracking routines and prints the calculated results. The number of I/Os used during various operations is calculated and printed. The following subroutines are called from MASTER as indicated.

SUBROUTINE	FUNCTION	CONDITION
OPENDA	Initialize direct access	Always
IOSDUN	Initialize I/Os	Always
DATAIN	Read input data	Always
MIXER	Mix cross sections	If mixing table is read
ICEMIX	Read ICE mixed library	If cross sections are not to be used from the restart data file
WRTRST	Write restart information	If a unit is defined for writing restart information
CORRE	Generate albedo-cross-section information	If differential albedo data are used
NSUPG	Create supergrouped data	Always
POINT	Calculate pointers	Always
JOMITY	Primarily controls geometry processing	Always
PRTPLT	Prints specified plots	If a plot data block is entered and the plot option is not turned off
CLEAR	Initializes arrays where calculated data are stored	Always
LODWTS	Load biasing data from direct access into memory	Always
GUIDE	Control tracking	Always
KEDIT	Edit calculated results	Always
FITFLX	Load fluxes for printing	If fluxes are calculated
FREAK	Print frequency distribution	Always
JSTIME	Timing	Always

OPENDA - This library routine initializes the direct-access files.

IOSDUN - This subroutine is called several times to indicate the number of I/Os used for various operations.

DATAIN - This subroutine controls the reading of all input data except the title and parameters. It is explained in more detail in Sect. F11.3.4.

MIXER - This subroutine is called only if mixing table information is to be entered as data. It controls the mixing of cross-section information and writes a Monte Carlo formatted mixed cross-section library for use later in the program. More details are contained in Sect. F11.3.5.

ICEMIX - This subroutine reads the Monte Carlo formatted mixed cross-section library and manipulates the cross sections to obtain the cross-section information used by KENO V.a. This information is then written on the direct-access data file. See Sect. F11.3.6 for additional information.

WRTRST - This subroutine is called if a unit has been defined on which to write restart information. The function of WRTRST is to write restart information as explained in Sect. F11.3.7.

CORRE - This subroutine is called only if the albedo data block specifies differential albedo data. It creates lethargy boundary tables and generates albedo cross-section energy group correspondence information. See Sect. F11.3.8 for additional details.

NSUPG - The purpose of this subroutine is to create supergrouped information and write it on the direct-access file as described in Sect. F11.3.9.

POINT - This subroutine calculates pointers to access data in memory.

JOMITY - This subroutine is primarily responsible for generating additional geometry data, checking the geometry data, writing appropriate geometry error messages, and printing the geometry that is used in the problem. Section F11.3.10 contains additional details.

LODWTS - This subroutine reads biasing information data from the direct-access data file, loads it into memory and prints it as described in Sect. F11.3.11.

PRTPLT - This subroutine is called to generate and print 2-D printer plots representing slices through the geometrical representation of the physical problem. See Sect. F11.3.12 for additional details.

CLEAR - This library routine initializes arrays where the calculated data are stored.

GUIDE - This subroutine guides the flow of the program through the actual tracking of each history. See Sect. F11.3.13 for a more detailed explanation.

KEDIT - This subroutine is responsible for editing the k-effectives and printing the various information calculated by KENO V.a. Section F11.3.14 contains additional details.

FITFLX - This subroutine is called only if fluxes are calculated. Its purpose is to determine the maximum number of regions for which fluxes can be contained in memory and to print the fluxes. A more detailed description is contained in Sect. F11.3.15.

FREAK - This subroutine is called to generate and print the frequency distribution of the k-effective calculated for each generation. The library routines SQRT and EXP are utilized in these calculations.

JSTIME - This library routine is called at the completion of a problem to compute the total amount of time used.

F11.3.4 PROBLEM DESCRIPTION

This section of the program controls the reading of the input data, excluding the title card and parameter data (Fig. F11.3.4). After the data have been read, they are written on the direct-access data file.

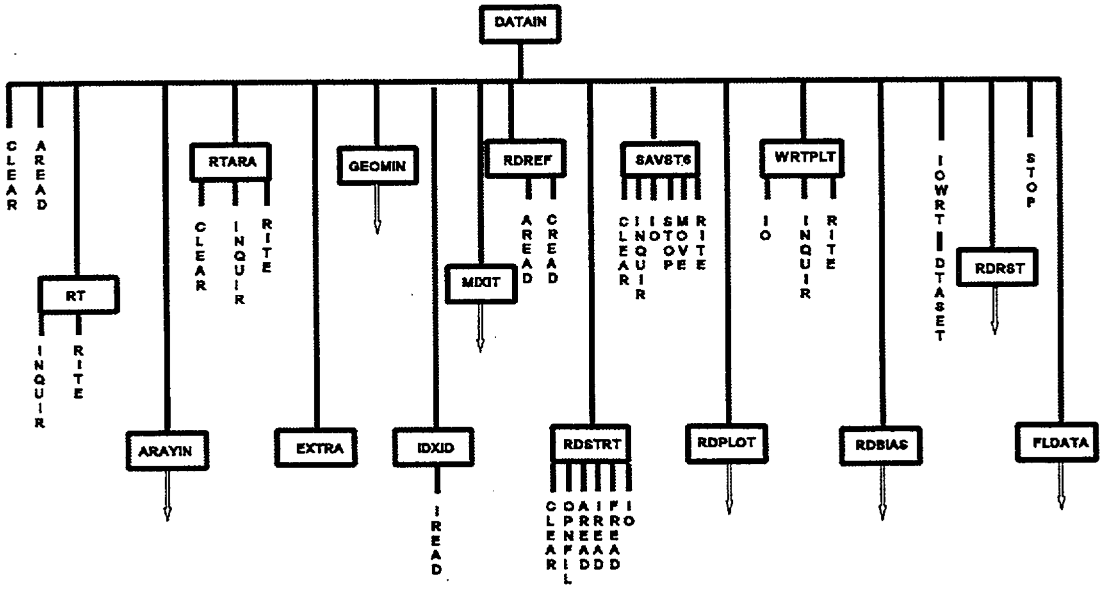


Figure F11.3.4 Flow chart of input data reading

DATAIN - This subroutine controls all the data reading with the exception of the title and parameter data. It initializes COMMON /STDATA/, the common that contains the start data, and initializes the MT array that contains the IDs of the 1-D cross sections that are to be utilized in the problem. The data reading is accomplished by reading blocks of data as described in Sect. F11.4. A keyword precedes the data, indicating the type of data to be read. After reading the keyword, the appropriate subroutine is called to read the accompanying data block. After the data block has been read, it is written on the direct-access data file. IOWRT is called several times to generate a table that lists the unit numbers used by KENO V.a, their names, data set names and the volume containing the data set. This table can be valuable for quality assurance applications. The library routine STOP is called to write error messages. Subroutine FLDATA is then called to supply information that was not entered as data.

CLEAR - This library routine is called to initialize COMMON /STDATA/. If biasing data are entered, it is called to clear the space that will contain the biasing data. CLEAR is called with two arguments, a beginning location and a length. It initializes all included locations to zero.

AREAD - This library routine is used to read the READ flag, the keyword for the type of data to be read, the END flag and the keyword for the type of data just completed. It can be called many times from DATAIN.

RT - This subroutine is called to write data on the direct-access data file. It passes information between DATAIN and RITE. INQUIR is called to return the value of the next direct-access record after the geometry data and/or the extra 1-D cross sections are written on a direct-access device.

RITE - This library routine is called from RT to write an array of data on the direct-access data file.

ARAYIN - This subroutine is called to read data defining the array size. It also reads the unit orientation data if any are entered as data. ARAYIN is not called for a single unit problem. See Sect. F11.3.4.1 for additional information. Data input for the array data block is described in detail in Sect. F11.4.5.

RTARA - This subroutine is called only if ARAYIN is called. CLEAR is called to zero the array before RTARA reads the array data from the scratch unit and writes them on the direct-access device. INQUIR is called to return the value of the next direct-access record after the array data are written on the direct-access device.

EXTRA - This dummy subroutine is provided to allow the user to input extra data that are not normally processed by KENO V.a. The user must provide the programming to read and utilize the data.

GEOMIN - This subroutine is called to read the geometry region data. See Sect. F11.3.4.2 for additional information. The geometry region data block is described in detail in Sect. F11.4.4.

IDX1D - This subroutine is called if the number of extra 1-D cross sections is greater than zero and an extra 1-D data block is entered. It reads the extra 1-D IDs and loads them into the MT array. The data reading is accomplished using the library routine IREAD. Section F11.4.9 describes the data input for defining extra 1-D data.

MIXIT - This subroutine is called to read the mixing table data block that defines the mixtures that are to be created. Section F11.1.4.3 explains the mixing procedure in more detail.

RDREF - This subroutine is called to read the boundary conditions (or albedo options) that are to be applied at the outer boundaries of the system described by the geometry data and the unit orientation data. The boundary condition data block is read using the library routine AREAD. The library routine, CREAD, is used to read the albedo names. Some preliminary data checks are made to detect invalid face code names and incompatible boundary conditions. Section F11.4.6 describes the data input for defining the boundary conditions.

RDSTRT - This subroutine is called to read the start data block that is used to define the spatial distribution of the initial generation. The library routine, CLEAR, is called to zero the array that will contain the start data. The library routine AREAD is used to read the keywords associated with the start data. IREAD and FREAD are used to read the integer and floating-point start data, respectively. The library routine IO is used to write the start data associated with start type 6 on the scratch data file. The library routine, INQUIR, returns the value of the next direct-access record after writing the start data on the direct-access device. Data input for defining the initial source distribution is described in Sect. F11.4.8.

SAVST6 - This subroutine is called to save the data associated with start type 6. SAVST6 is called only if start type 6 was specified in the start type data. See Sect. F11.4.8 for start type information. The library routine CLEAR is called to initialize the array that will contain the start data arrays. The library routine IO is used to read the start data array from the scratch data file and load it into memory. The library routine MOVE is used to move the start data array into the neutron bank. The library routine RITE is called to write the neutron bank data on the direct-access data file. The library routine, INQUIR, returns the value of the next direct-access record after writing the start type 6 data on the direct-access device.

RDPLOT - This subroutine is called to read the plot data block that is used to generate printer plots. Section F11.3.4.4 explains the processing of the plot data in more detail. Section F11.4.11 describes the plot input data in detail.

WRTPLT - This subroutine reads the plot data block from the scratch data file and loads it on direct access. IO reads the data from the scratch data file and loads them in memory. RITE writes the data on the direct-access data file. INQUIR returns the value of the next direct-access record.

RDBIAS - This subroutine is called to read the biasing or weighting data to be used in the problem. See Sect. F11.3.4.5 for more information. The biasing input data block is described in detail in Sect. F11.4.7.

IOWRT - This library routine is called with five arguments. They are, in order: (1) the output unit, (2) a four-character hollerith name representing a unit name, (3) the unit number represented by the second argument, (4) the number of words of hollerith information contained in the fifth argument, and (5) the hollerith information to be printed. IOWRT is called several times to generate a table of the unit numbers, their names, the data set names, and the volumes on which each resides.

DTASET - This library routine is called from IOWRT to provide the data set name of the requested I/O unit and the volume on which it resides.

RDRST - After the data reading is complete, this subroutine is called if a unit containing data for restarting the problem has been defined. It loads data from the restart data file as described in Sect. F11.3.4.6.

FLDATA - This subroutine is called to supply default data for arrays that were not entered as input. Section F11.3.4.7 contains a more detailed account of the exact procedure.

F11.3.4.1 *Array Data*

This section explains the procedure involved in reading the array data used in the problem (Fig. F11.3.5).

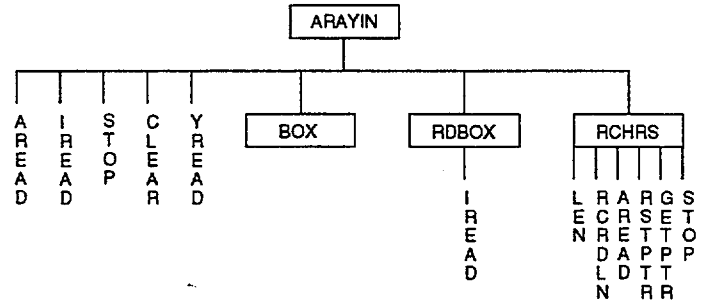


Figure F11.3.5 Flow chart for reading array data

ARAYIN - This subroutine is called from DATAIN when the words READ ARRA are encountered. It is responsible for reading the data parameters that define the size of each unit orientation array. The unit orientation array data block for each array is read by HLFWRD for the FILL option and by RDBOX for the LOOP option. BOX then writes the array data on the scratch data file. CLEAR is called to zero the unit orientation array and the data reading is done using the library routines AREAD, IREAD, and YREAD. Section F11.4.5 describes the data read by this subroutine.

AREAD - This library routine is used to read the keywords associated with the array data.

IREAD - This library routine is used to read the integer data associated with the array data.

STOP - This library routine is called to write an error message and stop if insufficient memory is available to accommodate the unit orientation array.

CLEAR - This library routine is called to zero the unit orientation array before it is loaded.

YREAD - This library routine is called to read the unit orientation data for the FILL option.

BOX - This subroutine is called to write the array data on the scratch data file.

RDBOX - This subroutine is called only if the LOOP option is used for entering the unit orientation data. It uses the library routine **IREAD** to read the unit orientation data. Some data consistency checks are made, and appropriate error messages are written if errors are encountered. If the input geometry is to be printed, RDBOX prints the unit orientation for each array.

RCHRS - This subroutine is used to read the comment associated with an array. The intrinsic function LEN determines the length of the comment. GETPTR is used to return the current pointer in the input buffer. RSTPTR resets the pointer. AREAD is used to read the input data and RCRDLN sets the length of the input buffer. STOP is called to write an error message and stop if the array comment is too long (i.e., the ending delimiter is missing).

F11.3.4.2 *Geometry Data*

This section of the program reads the geometry data (Fig. F11.3.6).

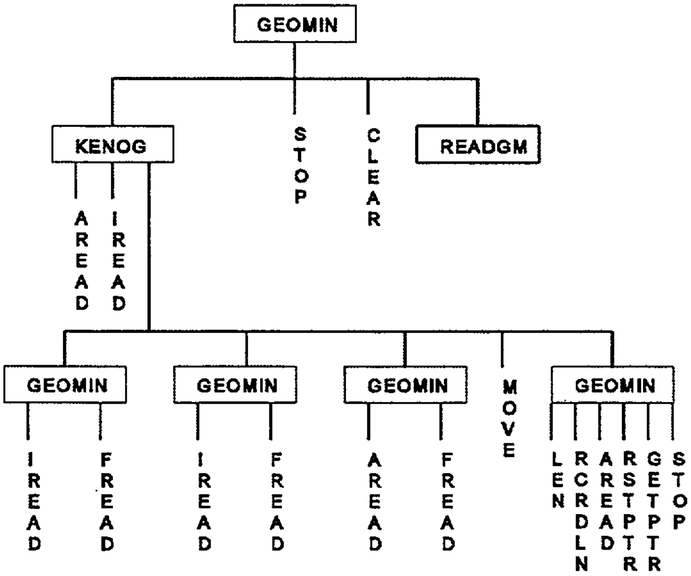


Figure F11.3.6 Flow chart for reading geometry data

GEOMIN - This subroutine controls the reading of the geometry data. KENOG is called to read each geometry region specification and write it on the scratch data unit. Pointers for the data arrays are then calculated and CLEAR is called to initialize the data arrays. READGM is called to read the data from the scratch data unit and load them into the appropriate data arrays. STOP is called to write an error message if more storage is needed. The data block read in this portion of the program is described in Sect. F11.4.4.

KENOG - This subroutine uses AREAD to read the geometry word. IREAD is called to read the unit number for a unit or box type. It is also used to read the number of replicated regions. XXIN is called to read the mixture number, the bias ID number and the geometry dimensions. The origin specification, if any, associated with a sphere, hemisphere, cylinder or hemicylinder is read by RDORGN. The chord specification, if any, associated with a hemisphere or hemicylinder is also read by RDORGN. The necessary geometry data block is written on the scratch data unit. If the input geometry data block is to be printed, KENOG does this as the data are read.

XXIN - This subroutine is called to read the mixture number, the bias ID number and the geometry dimensions. IREAD is used to read the mixture number and bias ID number. FREAD is used to read the geometry region dimensions.

XXINA - XXINA is an entry point in XXIN. It is called when the geometry word ARRA or ARRAY is encountered. IREAD is used to read the mixture number. FREAD is used to read the geometry region dimensions.

RDORGN - This subroutine uses AREAD to read a word. If the word is ORIGIN, it uses FREAD to read the points defining the origin. If the word is CHORD, it uses FREAD to read the offset of the plane with respect to the origin. If the word is not ORIGIN or CHORD, a flag is set to prevent KENOG from attempting to read another geometry word.

RCHRS - This subroutine is used to read the comment associated with a unit in the geometry region data. LEN is used to return the length of the comment. GETPTR is used to return the current pointer in the input buffer. RSTPTR resets the pointer. AREAD is used to read the input data and RCRDLN sets the length of the input buffer. STOP is called to write a message and terminate the problem if the comment is too long.

READGM - This subroutine reads the geometry data from the scratch data unit and loads them into the proper arrays.

F11.3.4.3 *Mixing Table Data*

This section deals with reading the mixing table data.

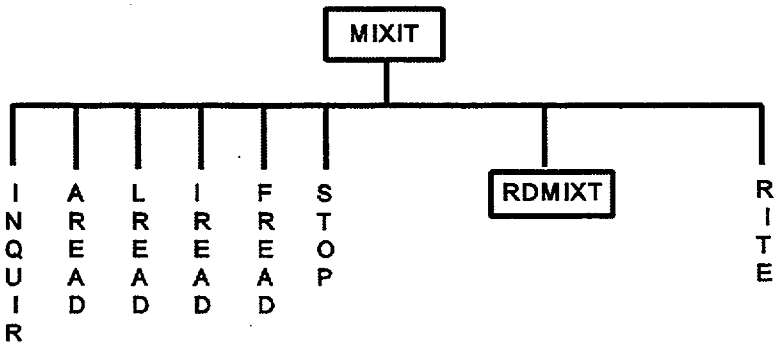


Figure F11.3.7 Flow chart for reading mixing table data

MIXIT - This subroutine uses AREAD to read the mixture keywords and the scattering keyword. IREAD is used to read the mixture numbers and the number of scattering angles as well as the nuclide IDs. LREAD is used to determine if the next digit is a numeric digit. FREAD reads the number densities. The necessary data arrays are written on the scratch data file. Pointers are calculated for the necessary storage arrays and RDMIXT is called to load the data from the scratch file into the storage arrays. RITE is called to write the mixing table information on the direct-access device and INQUIR is called to return the value of the next direct-access record.

LREAD - This library routine returns a value of "true" if the next character is a numeric digit. Otherwise, a value of "false" is returned.

STOP - This library routine is called from MIXIT if the storage space is insufficient to hold the mixing table arrays.

RDMIXT - This subroutine reads the mixing table data arrays from the scratch data file and loads them into the storage arrays.

F11.3.4.4 *Plot Data*

This section of the program reads the plot data used to generate character or color plot maps of the mixtures, units and/or bias IDs used in the problem (Fig. F11.3.8). The plot input data block is described in Sect. F11.4.11.

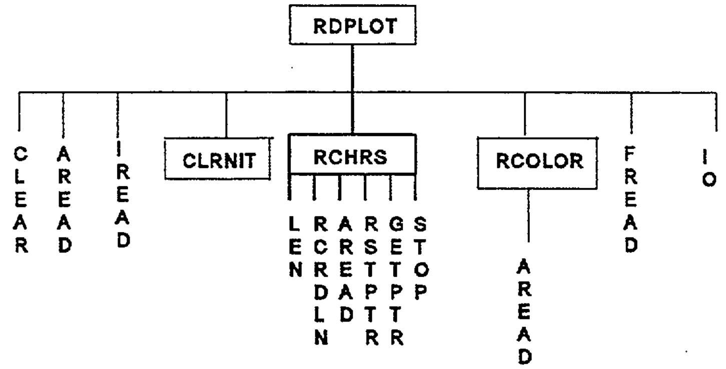


Figure F11.3.8 Flowchart for reading plot data

RDPLOT - This subroutine uses CLEAR to initialize the data arrays. RCHRS is used to read the plot title and the character string of symbols to be used in the plot. AREAD, IREAD and FREAD are used to read the plot input data. SQRT is used to determine the normalization factor for the direction cosines and IO is called to load the plot data on the scratch data file.

RCHRS - This subroutine is used to read the plot title and the character string that defines the symbols to be used in the plot. The intrinsic function LEN is used to determine the length of a character variable. GETPTR is used to return the current pointer in the input buffer. RSTPTR resets the pointer. AREAD is used to read the input data and RCRDLN sets the length of the input buffer. STOP is called to write an error message if the plot title is too long and terminate the problem.

CLRNIT - This subroutine initializes the color tables used in creating color plots.

RCOLOR - This subroutine is used to redefine the red, green, and blue components of the plot color tables. AREAD is used to read the color table position and the new red, green, and blue values of that position.

F11.3.4.5 *Biasing Data*

This section of the program reads the biasing data used in the problem. The biasing input data block is described in Sect. F11.4.7.

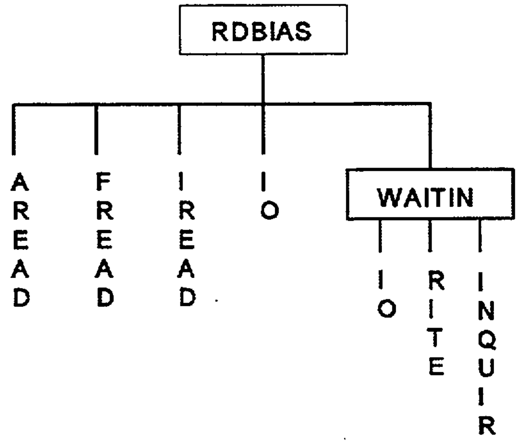


Figure F11.3.9 Flow chart for reading biasing data

RDBIAS - This subroutine is responsible for reading the biasing data block and writing it on the scratch data file. AREAD is used to read the keywords used in the biasing data and a title for the biasing material if the energy and space-dependent values of the biasing function are entered by the user. IREAD and FREAD are used to read the numerical data. Pointers for the storage arrays needed to process the biasing data are determined, and WAITIN is called to load the data from the scratch data file into the storage arrays and write them on the direct-access data file.

WAITIN - This subroutine reads the biasing data block from the scratch data file and loads it into the storage arrays. IO is used to load the energy and space-dependent biasing function (*wtavg*) into the storage arrays. RITE is used to write the biasing data on the direct-access data file. INQUIR is then called to return the value of the next direct-access record.

F11.3.4.6 *Restart Data*

This section of the program reads restart information from the restart data file (Fig. F11.3.10).

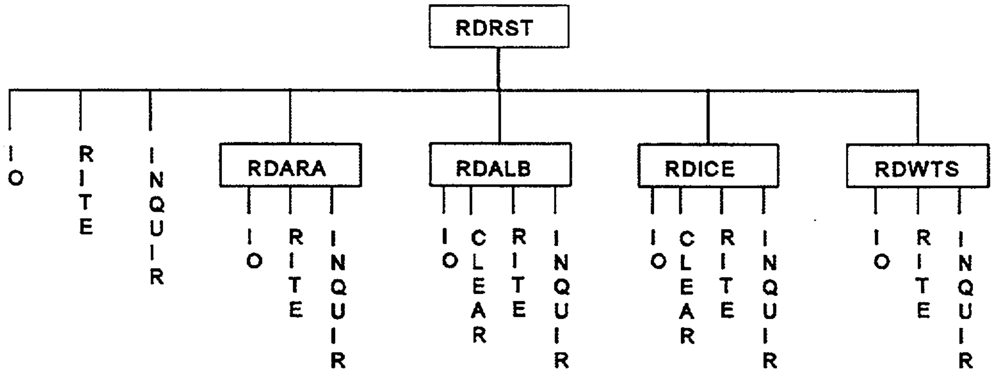


Figure F11.3.10 Flow chart for reading restart data

RDRST - This subroutine is called only if the problem is to use data from the restart data file. The program recognizes that restart data will be read if the restart unit is defined as a number greater than zero. IO is used to load the array that contains the 1-D IDs from the restart data file. Each type of data is loaded from the restart data file using IO and is written on the direct-access data file by RITE. All restart data except the mixed cross-section data, the differential albedo data, the array data, and the biasing data are processed directly in RDRST. RDICE is called to load the cross-section data on the direct-access data file, RDALB is called to load the differential albedo data on the direct-access data file, RDARA is called to load the array data on the direct-access data file and RDWTS is called to load the biasing data on the direct-access data file. The library routine INQUIR is called to return the value of the next direct-access record.

RDALB - This subroutine is called from RDRST to read the albedo data block from the restart data file and write it on the direct-access data file. IO is used to load the albedo pointer and length arrays into memory from the restart data file. CLEAR is called to zero the albedo pointer and length arrays and RITE writes them on the direct-access data file. Each record of albedo data is read from the restart data file, loaded into memory using IO, and written on the direct-access data file using RITE. When all the records of albedo data have been processed, the updated pointer and length arrays are rewritten over the initial ones using RITE. INQUIR returns the value of the next direct-access record.

RDICE - This subroutine is called from RDRST to read the cross-section data block from the restart data file and write it on the direct-access data file. CLEAR is called to zero the pointer and length arrays. The length array is then read from the restart data file. RITE is used to write the pointer array and the length array on the direct-access data file. Then IO and RITE are used to load the cross-section data from the restart data file and write them on the direct-access data file. This procedure is repeated for every record of each mixture. The updated pointer and length arrays are then rewritten over the initial ones using RITE. INQUIR returns the value of the next direct-access record.

RDARA - This subroutine is called from RDRST to read the array data block from the restart data file and write it on the direct-access data file. IO is used to load the data from the restart data file and RITE is used to write them on the direct-access data file. INQUIR returns the value of the next direct-access record.

RDWTS - This subroutine is called from RDRST to read the biasing data block from the restart data file and write it on the direct-access data file. IO is used to load the data from the restart data file and RITE is used to write them on the direct-access data file. INQUIR returns the value of the next direct-access record.

F11.3.4.7 *Generate Remaining Data*

This section of the program is responsible for generating data blocks that are required to solve a problem but are not entered directly as data.

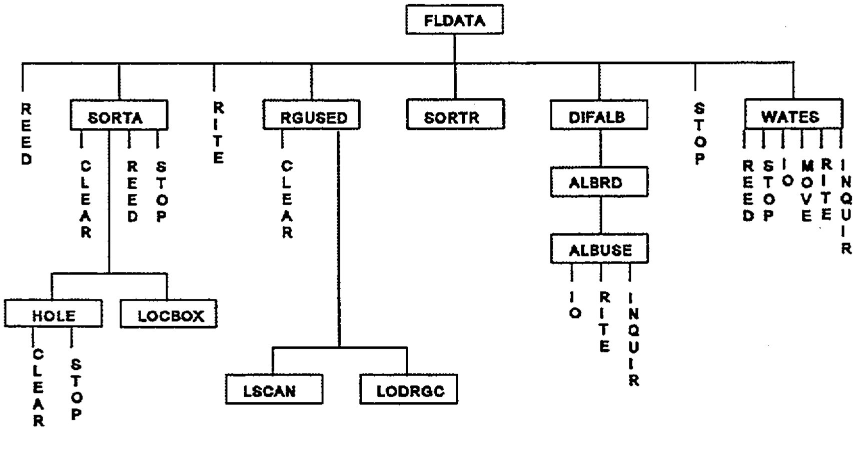


Figure F11.3.11 Flow chart for generating remaining data

FLDATA - The library routine REED is used to load data arrays from the beginning of the mixtures used in the geometry through the geometry data. The library routine, STOP, may be called to print an error message. SORTA is called to determine which arrays and holes are used as well as the array and hole nesting levels. RITE is called to write the geometry data on the direct-access data file. RGUSED is called to determine which geometry regions are used in the problem. SORTR is called to generate the mixture correspondence array. It is called again to generate the bias region correspondence array. These correspondence arrays are used to avoid storing mixture cross sections and biasing data that were entered as data but are not actually used in the problem. If boundary conditions specify differential albedo data and they are not available from the restart data file, DIFALB is called to read the albedo data block and load the requested data on the direct-access file. If the requested biasing data were unavailable from the restart data file, WATES is called to load the energy and position biasing function (*wtavg* array) on the direct-access data file.

SORTA - This subroutine checks to see that the global array is properly defined. It determines the array correspondence array and the nesting levels for holes and arrays. SORTA uses CLEAR to initialize arrays. REED is used to load the array data and STOP is used to write an error message and terminate if more computer storage is needed for the problem.

HOLE - This subroutine is called from SORTA to determine what holes occur at the next nesting level and to adjust the array nesting level for arrays that occur in holes. It also checks to be sure holes are not recursively nested.

LOCBOX - This function subprogram is called from SORTA to determine the unit or box type at a given position in the unit orientation array.

RGUSED - This subroutine determines which geometry regions are used in the problem. The library routine CLEAR is called to zero the space for the region correspondence array. LSCAN determines if a particular unit or box type has been used in the unit orientation array and if it has, LODRGC is called to load the region number into the region correspondence array.

LSCAN - This is a logical function that returns a value of true if the specified unit or box type is used in the unit orientation array. A value of false is returned if the unit or box type was not used in the unit orientation array.

LODRGC - This subroutine loads the region number in the region correspondence array.

SORTR - This subroutine is called twice from FLDATA to create a mixture correspondence array and a biasing correspondence array. These correspondence arrays are used to avoid storing mixture cross sections and biasing information that were defined in the input data but were not referenced in the geometry data utilized in the problem. They are also used throughout the code for accessing the proper mixture cross sections and biasing information.

DIFALB - This subroutine is called if differential albedos are specified as a boundary condition but are not available from the restart data file. It rewinds the albedo data file, reads the header record and calculates pointers. ALBRD is called to load the albedo data.

ALBRD - This subroutine searches through the albedo data file to locate the requested albedo name or boundary condition. If it is not found, an error message is written. If it is found, the number of different differential albedos that were requested are tabulated and ALBUSE is called.

ALBUSE - This subroutine writes the pointer array on the direct-access data file. IO is used to load data from the albedo data file and RITE is used to write the data on the direct-access data file. Then the pointer and length arrays are rewritten on the direct-access data file. The library routine INQUIR is called after each call to RITE to return the value of the next direct-access record.

WATES - This subroutine reads the biasing input data block from the direct-access data file and reads the KENO V.a weights library. STOP is called if the computer storage space is too small to contain the energy- and position-dependent biasing function (*wtavg*). IO is used to load the biasing function into a temporary storage array. If a specific biasing function is to be used, MOVE is called to load it into the *wtavg* array. If biasing or weighting data are entered from cards, REED is used to load the data into a temporary storage array. If a specific biasing function that was loaded from cards is to be used in the problem, MOVE is called to load it into the *wtavg* array. When all the data have been processed, RITE is called to load the biasing input data on the direct-access data file. RITE is called again to load the *wtavg* array on the direct-access data file. INQUIR returns the value of the next direct-access record after each call to RITE.

F11.3.5 CREATE A MIXED CROSS-SECTION DATA FILE

The function of this portion of the program is to utilize the mixing table data and AMPX working format library from the cross-section data file to create a Monte Carlo formatted mixed cross-section data file (Fig. F11.3.12). This data file has the same format as an ICE mixed cross-section MORSE/KENO V.a library.[1]

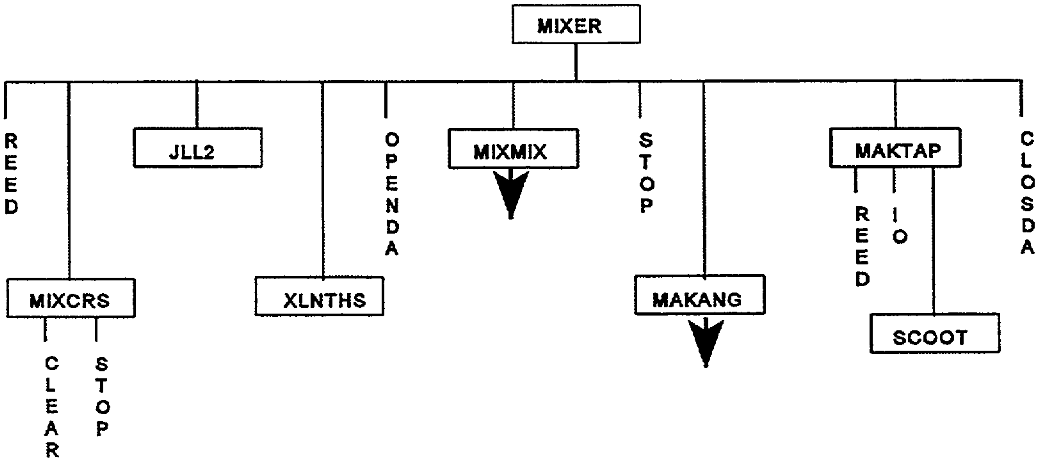


Figure F11.3.12 Flow chart for mixing cross sections

MIXER - This subroutine controls the mixing of cross sections and the generation of the angles and probabilities to create a Monte Carlo formatted mixed cross-section library. First the mixing table is read from the direct-access data file, MIXCRS is called to generate the mixture correspondence array, and JLL2 is called to generate the array that points to the beginning of each group in the triangularized mixture array. XLNTHS is called to calculate the number of direct-access blocks required for the mixtures and MIXMIX is called to do the mixing. MAKANG is called to generate the angles and probabilities from the mixed cross sections, and MAKTAP is called to write the Monte Carlo formatted mixture tape. Calls to STOP are made to print error messages if more space is needed to mix the cross sections or to process the angles.

MIXCRS - The mixture correspondence array is generated by this subroutine. This array relates the mixture number in the mixing table to the mixture index.

JLL2 - This routine generates an array of pointers that point to the beginning location of each energy group in a triangularized mixture transfer array.

XLNTHS - This routine computes the number of direct-access blocks necessary to hold the mixtures.

MIXMIX - This subroutine controls the actual mixing of the cross sections. See Sect. F11.3.5.1 for a more detailed description of subroutine MIXMIX.

MAKANG - This subroutine controls the generation of angles and probabilities. See Sect. F11.3.5.2 for a more detailed description of subroutine MAKANG.

MAKTAP - Subroutine MAKTAP is used to write a mixed cross-section library in a format similar to the ICE Monte Carlo library format. MAKTAP calls SCOOT to compress out zero 1-D cross sections.

SCOOT - This subroutine eliminates 1-D cross sections that are zero in all groups.

F11.3.5.1 *Cross-Section Mixing*

This section of the program produces mixed cross sections (Fig. F11.3.13).

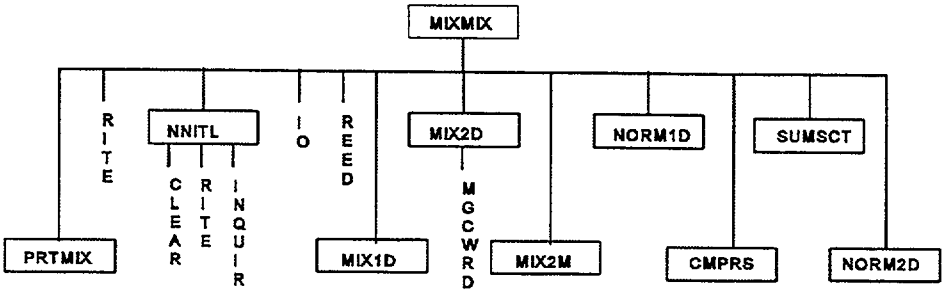


Figure F11.3.13 Flow chart of mixing operations

MIXMIX - This subroutine controls the actual mixing of the cross sections. It calls PRTMIX to print the mixing table and calls NNITL to initialize the mixtures. It then reads the input AMPX working library, calls MIX1D to mix the 1-D cross sections, and calls MIX2D to mix the 2-D arrays. After all the nuclides selected from the working library have been mixed, the already mixed cross sections may be used as input for further mixing operations using MIX1D to mix the 1-D cross sections and MIX2M to mix the 2-D cross sections. NORM1D is then called to normalize the fission spectrum and to prepare the adjoint production cross sections if necessary. CMPRS is called to generate the magic word array and to compress the 2-D arrays. SUMSCT is called to sum the transfer array by group and NORM2D is called to convert the transfer array to a probability density function.

PRTMIX - This subroutine prints the requested number of scattering angles and the mixing table.

NNITL - This subroutine initializes the mixture cross sections to zero on the direct-access storage. CLEAR is called to zero the array, RITE writes the array on the direct-access device, and INQUIR returns the value of the next direct-access record.

MIX1D - This subroutine mixes the 1-D cross sections. It mixes most 1-D cross sections using number densities, but the fission spectrum is mixed based on $\upsilon\Sigma_f\phi dE$ for the nuclide being mixed. The flux is the flux used to generate the multigroup cross sections, if it is available. Otherwise a flat flux is used. If the problem is an adjoint case, MIX1D prepares the cross sections in adjoint form as they are mixed.

MIX2D - This subroutine mixes the 2-D cross sections from a working library. The mixture cross sections are stored in a triangularized array. The library routine, MGCWRD, is called to create the magic word for accessing the mixture cross sections. If the problem is an adjoint case, MIX2D prepares the cross sections in adjoint form as they are mixed.

MIX2M - This subroutine mixes previously mixed 2-D cross sections into new mixtures using new number densities or volume fractions.

NORM1D - This subroutine is used to normalize the fission spectrum vector. If the problem is an adjoint problem, it also interchanges $\upsilon\Sigma_f$ and the fission spectrum.

CMPRS - This subroutine compresses out zeros in the 2-D mixture arrays and generates the pointer array used to access data in these arrays.

SUMSCT - This subroutine is used to sum the transfers for each group.

NORM2D - This subroutine normalizes the transfer arrays and divides the P_ℓ arrays by $(2_\ell + 1)$.

F11.3.5.2 *Generate Angles and Probabilities*

This portion of the program produces the angles and probabilities for each mixture (Fig. F11.3.14).

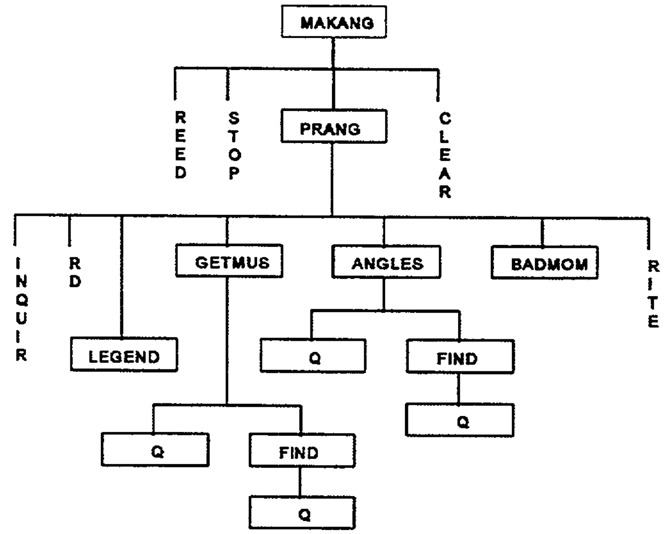


Figure F11.3.14 Flow chart of angle and probability generation

MAKANG - This subroutine controls the generation of angles and probabilities. If the available storage is insufficient to generate all the angles and probabilities for one mixture at a time, MAKANG supergroups the data. Subroutine PRANG is called to actually generate the angles and probabilities for a supergroup.

PRANG - This subroutine reads the mixture data for a supergroup and loops through each transfer in the supergroup. It calls LEGEND to convert the Legendre expansion to moments. GETMUS is called to generate the orthogonal polynomials and ANGLES is called to generate the angles and probabilities. If there is an error, BADMOM is called to print the data input and the data corresponding to the angles and probabilities actually used.

LEGEND - This subroutine determines the moments of a function from a Legendre expansion of the function.

GETMUS - This subroutine calculates the μ_i's and σ_i's that determine the orthogonal polynomials, the Q_i's.[2]

ANGLES - This subroutine determines the angles and probabilities from the Q_i polynomials.[2] It calls FIND to determine the roots of Q_i, and calls the function Q to evaluate Q_i.

FIND - This subroutine finds the roots of a polynomial Q_i by using an interval-halving technique.

Q - This function evaluates a polynomial Q_i by using the recursion formulas.

$$Q_{i+1}(x) = (x - \mu_{i+1})Q_i(x) - \sigma_i^2 Q_{i-1}(x) \ ,$$

$Q_0(x) = 1.0$, and

$Q_1(x) = x - \mu_1$.

BADMOM - This routine is called when an error is detected in generating the angles and probabilities. BADMOM prints the Legendre coefficients that were input to the calculation and the moments corresponding to these coefficients. It then computes the moments corresponding to the angles and probabilities actually generated and the Legendre coefficients corresponding to these moments. BADMOM then prints these moments and coefficients so they may be compared with the original moments and coefficients.

F11.3.6 WRITE CROSS SECTIONS ON DIRECT-ACCESS FILE

This section of the program is responsible for reading the Monte Carlo mixed cross-section library, selecting the desired information, and writing it on the direct-access data file (Fig. F11.3.15).

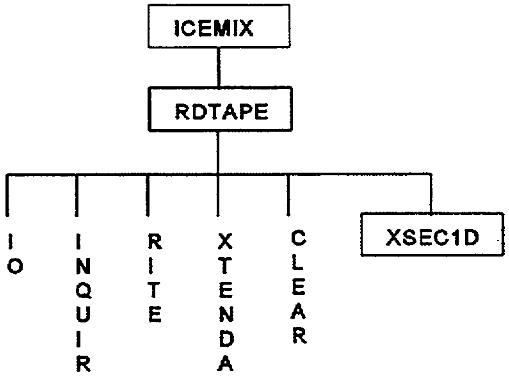


Figure F11.3.15 Flow chart of mixed cross-section processing

ICEMIX - This subroutine reads information from the Monte Carlo mixed cross-section library, performs preliminary checks for data consistency, calculates pointers for the desired data arrays and calls RDTAPE to actually read the library and write the desired information on the direct-access data file.

RDTAPE - This subroutine reads the Monte Carlo formatted mixed cross-section library, sorts out the data needed by KENO V.a, and writes the data on the direct-access data file. The library routine SQRT is used in calculating the inverse velocities used by KENO V.a. The library routine IO is used repeatedly to read and load data from the Monte Carlo formatted mixed cross-section library. RITE is used to write data required by KENO V.a on the direct-access data file. INQUIR returns the value of the next direct-access record after each call to RITE.

XTENDA is a library routine that is called to extend the number of blocks for a direct-access device if too few blocks were initialized. XSEC1D is called to process the 1-D cross sections. RDTAPE manipulates the pointer array and writes it on the direct-access data file. The P_0 cross sections, angles and probabilities are also processed by RDTAPE.

XSEC1D - This subroutine sorts the 1-D cross sections and loads them in the following order: total cross section, scattering cross section, production cross section, absorption cross section, extra cross sections for special purposes, and the fission spectrum. The extra cross sections require the user to provide programming to utilize them. If one of the required cross sections is not found, it is padded with zeros. The cross sections are normalized by the total cross section and the fission spectrum is summed and normalized to 1.0. The data are transferred back to RDTAPE where they are written on the direct-access data file.

F11.3.7 WRITE INPUT DATA ON RESTART FILE

This section of the program is responsible for writing all data except the calculated results on the restart data file (Fig. F11.3.16). This section of the program is omitted if a unit number has not been assigned for the restart data file. This information is entered as parameter data, WRS=, as described in Sect. F11.4.3. The calculated results are written on the restart data file later in the program. The restart data file is used for restarting a problem.

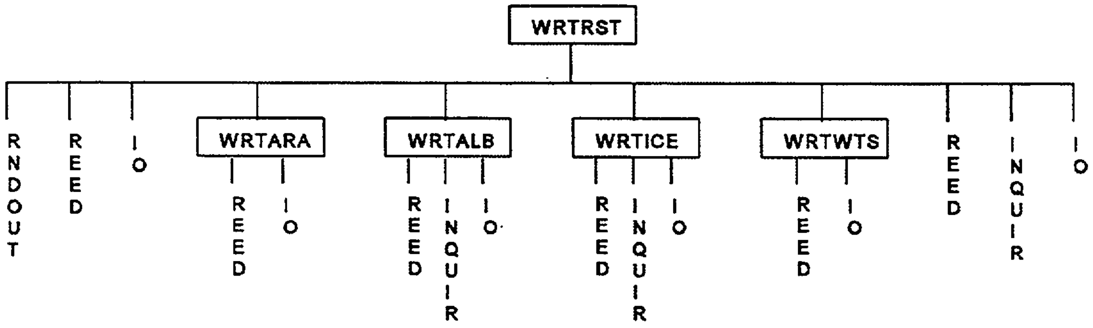


Figure F11.3.16 Flow chart for writing data on the restart data file

WRTRST - This subroutine writes the input data on the restart data file. RNDOUT is called to return the seed of the next random number. The problem title and parameter data are in the first record written on the restart data file. The array containing the IDs of the 1-D cross sections is then written. The geometry region data, mixing table data, extra 1-D data, biasing data, start data, plot data, and energy and inverse velocities are written. The unit orientation header record is written, and WRTARA is called to write the unit orientation data. The albedo header record is written, and WRTALB is called to write the albedo data. The cross-section header record is written, and WRTICE is called to write the cross sections. The header record for user-supplied weighting data is written, and WRTWTS is called to write the user-supplied weighting or biasing data on the restart data file.

WRTARA - This subroutine is called from WRTRST to write the array number, array size and corresponding unit orientation array on the restart data file for each array that is entered in the problem. The library routine REED is called to load the direct-access pointers for the albedo data blocks. INQUIR is called to return the value of the next direct-access record. REED is used to read the data from the direct-access data file and IO is used to write the data on the restart data file.

WRTALB - This subroutine is called from WRTRST to write the albedo data on the restart data file. The library routine REED is used to read the albedo data from the direct-access data file and IO is used to write the data on the restart data file.

WRTICE - This subroutine is called from WRTRST to write the cross-section data block on the restart data file. REED is called to load the direct-access pointers for the cross-section data blocks. INQUIR is called to return the value of the next direct-access record. The library routine REED is used to read the rest of the cross-section information from the direct-access data file and IO is used to write it on the restart data file.

WRTWTS - This subroutine is called from WRTRST to write the biasing input data block on the restart data file. The library routine REED is used to read the data block from the direct-access data file, and IO is used to write it on the restart data file.

F11.3.8 GENERATE ALBEDO CROSS-SECTION CORRESPONDENCE TABLES

This section of the program generates the correspondence tables that are necessary to correlate the energy group structures of the cross sections and albedos (Fig. F11.3.17). It is invoked only if differential albedos are used in the problem.

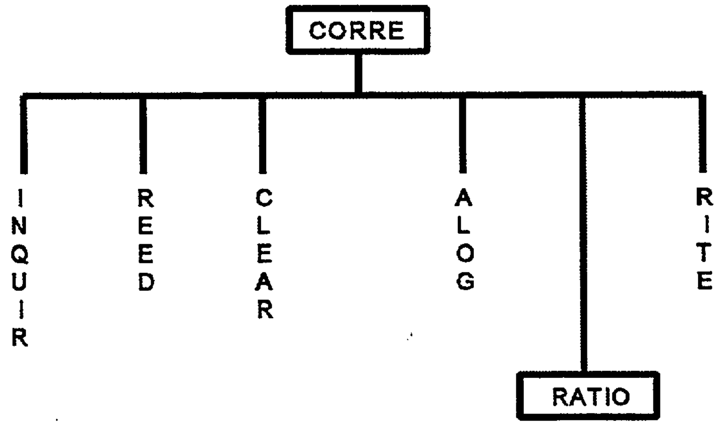


Figure F11.3.17 Flow chart for generating correspondence arrays

CORRE - This subroutine is called only if differential albedos are used. The library routine REED is called to read arrays. The library routine INQUIR is called to return the value of the next direct-access record. The library routine CLEAR is used to initialize arrays. CORRE reads the energy bounds for the albedos and the cross sections and converts them to lethargies by using the library routine ALOG. RATIO is called to create a probability array relating the cross-section energy bounds to those of the albedos. A second call is made to RATIO to create and print the probability array relating the albedo energy bounds to those of the cross sections. RITE is used to write the correspondence tables on the direct-access data file.

RATIO - This subroutine generates the pointer array that is used to access the albedo data. It also generates probability arrays that are used to correlate the cross-section energy group structure with the albedo energy group structure.

F11.3.9 GENERATE SUPERGROUPED DATA

The function of this section of the program is to create supergrouped data from the energy-dependent input data (Fig. F11.3.18). It determines which energy group has the largest amount of data associated with it, and determines how many supergroups must be created to be able to fit the data into the available computer memory. The energy groups associated with each supergroup are tabulated and supergrouped data are written on the direct-access supergroup data file, one supergroup at a time.

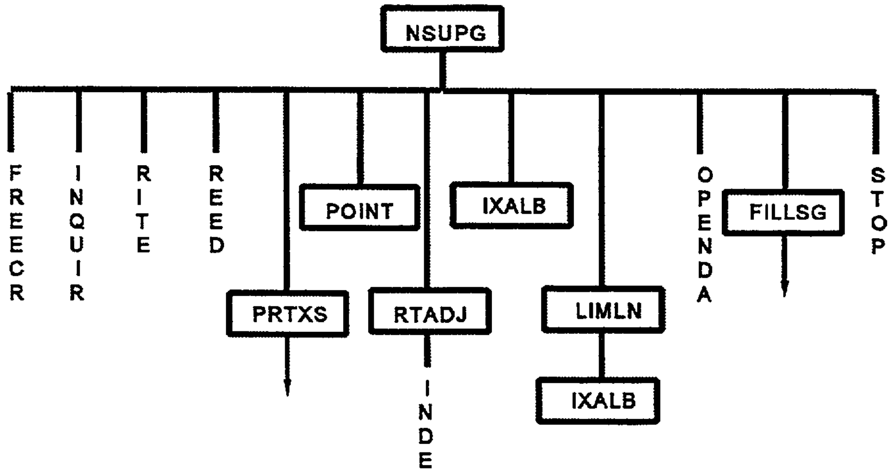


Figure F11.3.18 Flow chart of supergroup creation

NSUPG - This subroutine controls the creation of the supergrouped data and writes them on the direct-access supergroup data file. The library routines RD and REED are used to load the cross-section data and albedo data from the direct-access data file. The library routine INQUIR is used to return the value of the next direct-access record. PRTXS is called to print the cross-section information as described in Sect. F11.3.9.1. POINT is called to determine the storage requirements of the nonsupergrouped data and create pointers to access these data. The first portion of the additional information table is printed in the computer output. RTADJ is then called to right adjust the albedo boundary condition names. These albedo boundary condition names are then printed in the additional information table, thus completing the table. If insufficient space is available, STOP is called to write a message and terminate the problem. A rough estimate of the number of supergroups is made, and an implied loop iterative procedure is used to determine the number of supergroups that must be created to fit the problem in the available space. IXALB is called to determine the amount of space required for the albedos corresponding to a cross-section energy group. A check is made to be sure the largest amount of data for a single energy group will fit in the available memory. If they will, supergrouping is possible. Otherwise a message is written and a stop is executed. Once the number of supergroups is determined, LIMLN is called to calculate and print information in the space and supergroup information table. The printed information includes the supergroup and corresponding energy groups, the length of the cross section and albedo data for the supergroups and the total length of each supergroup. FILLSG is then called to construct the supergroups as described in Sect. F11.3.9.2. After the supergrouped data have been written on the direct-access device, the library routine FREECR is called to release the space that is not needed for the problem.

PRTXS - This subroutine prints the cross-section information as described in Sect. F11.3.9.1.

POINT - This subroutine determines the storage requirements of the nonsupergrouped data and creates pointers to access data within the nonsupergrouped storage array.

RTADJ - This subroutine right adjusts the albedo boundary condition names which are read in as left-adjusted data. RTADJ utilizes the FORTRAN-supplied intrinsic function INDEX to locate the first blank character.

IXALB - This function determines the amount of space necessary to contain the albedo data corresponding to a cross-section energy group.

LIMLN - This subroutine is called to calculate and print the supergroup number, the energy groups contained in the supergroup, the length of the cross section and albedo data in the supergroup, and the total length of the supergroup. IXALB is utilized to obtain the amount of space required to contain the albedo data corresponding to a cross-section energy group.

FILLSG - This subroutine constructs supergroups as described in Sect. F11.3.9.2.

F11.3.9.1 *Print Macroscopic Cross Sections*

This portion of the program prints the macroscopic cross-section data for materials used in the problem description (Fig. F11.3.19).

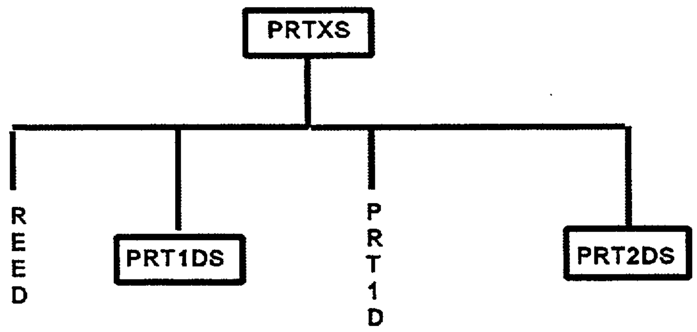


Figure F11.3.19 Flow chart for printing macroscopic cross sections

PRTXS - If macroscopic cross sections are not to be printed, a return is executed. PRTXS loops over the number of mixtures that are used in the problem. REED is used to load the data, then pointers into the data arrays are calculated. If the space is insufficient to contain the data, a message is printed. If 1-D mixture cross sections are to be printed, PRT1DS is called to print them. If the extra 1-D cross sections are to be printed, REED is used to load the data and the library routine PRT1DS is called to print them. If the transfer arrays are to be printed, REED is called to load the data and PRT2DS is called to print them. If the number of scattering angles is greater than zero and the mixture probabilities and angles are to be printed, REED is used to load probability data and PRT2DS prints them. Then REED loads angle data and PRT2DS prints them.

PRT1DS - This subroutine prints the macroscopic 1-D cross sections, one energy group at a time.

PRT2DS - This subroutine prints a 2-D variable length array in a compact manner.

F11.3.9.2 *Write Supergroup Data File*

This portion of the program collects the group-dependent data by supergroup and writes them on the direct-access supergroup data file (Fig. F11.3.20).

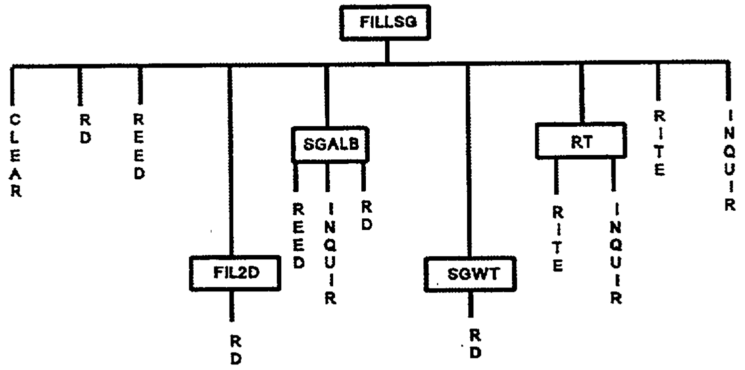


Figure F11.3.20 Flow chart for loading supergrouped data

FILLSG - This subroutine is responsible for filling the supergroups and writing them on the direct-access supergroup data file. For each supergroup, CLEAR is called to initialize the working space. Pointers are calculated for the data arrays, and space is allotted for the calculated data such as fluxes that will be supergrouped. The library routines REED and RD are used to load data from the direct-access data file into the proper arrays as follows: For each mixture the inverse velocities are loaded, followed by the pointer array and all the 1-D cross-section arrays. RD and REED are used to load the 2-D arrays from the direct-access data file. FIL2D is called to load them into the supergroup. Pointers are calculated for the albedo data and SGALB is called to load the albedo data into the supergroup. SGWT loads the average weight array (biasing data) into the supergroup. FILLSG then calls RT to write the calculated data for the supergroup on the direct-access supergroup data file. RITE writes the group-dependent data on the direct-access supergroup data file. INQUIR is called as necessary to return the value of the next direct-access record.

FIL2D - This subroutine loops over the number of mixtures. It uses RD to load the 2-D cross-section data for the supergroup from the direct-access data file; first the P_0 data, then for each scattering angle, the angles and probabilities. The pointer array is redefined so the supergrouped data can be accessed.

SGALB - This subroutine uses REED and RD to load the albedo data for the supergroup from the direct-access data file. INQUIR is used to provide the value of the next direct-access record. It loops over the number of differential albedos used in the problem. Then the pointer array is redefined so the supergrouped data can be accessed.

SGWT - This subroutine loops over the number of biasing regions used in the problem. RD is used to load the average weight array from the direct-access data file. If any average weight entry remains undefined or zero, it is set to the default value of weight average.

RT - This subroutine is called to write the calculated data (fluxes, etc.) on the direct-access supergroup data file using the library routine, RITE. INQUIR is used to return the value of the next direct-access record, after the call to RITE.

RITE - This library routine is used to write the group-dependent data for the supergroup on the direct-access supergroup data file.

F11.3.10 PROCESS GEOMETRY

This portion of the program is primarily responsible for loading the geometry data, generating additional geometry data, checking the geometry for consistency, writing error messages related to the geometry, and printing the geometry that is used in the problem (Fig. F11.3.21).

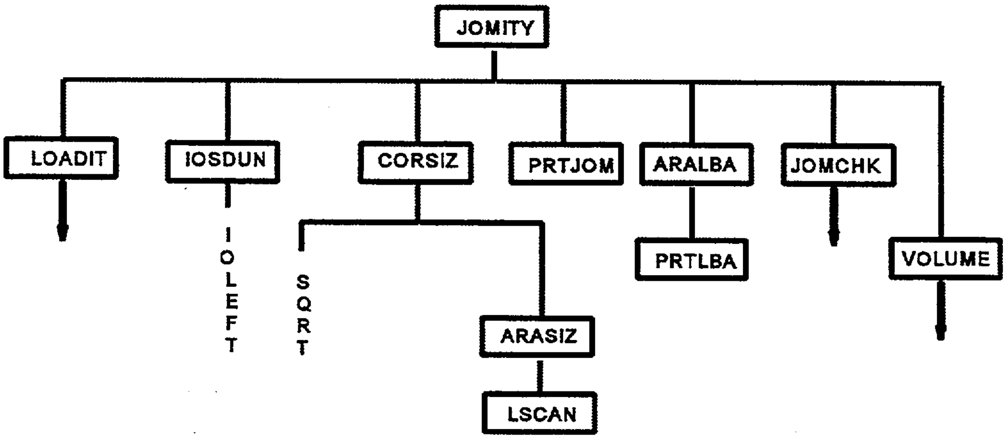


Figure F11.3.21 Flow chart for processing geometry

JOMITY - This subroutine is responsible for generating additional geometry data, checking the geometry data, writing geometry error messages, and printing the geometry.

LOADIT - This subroutine loads the geometry data and the non-supergrouped portion of the albedo data. Section F11.3.10.1 contains a more detailed description of the procedure.

IOSDUN - This subroutine returns the number of IOs used. The library routine IOLEFT returns the number of input/output requests left before the system cancels the job.

CORSIZ - This subroutine sends the appropriate lattice or array information to ARASIZ for each lattice that is used in the problem. Using this information, it calculates the overall positive dimensions of the global array. The library routine, SQRT, is utilized to calculate the maximum chord length of an unreflected array, a reflected array or a single unit problem.

ARASIZ - This subroutine uses the array unit orientation data to calculate the positive dimensions of the core boundary for that array or lattice. The function LSCAN is called to determine if a specified unit or box type has been used in the array. ARASIZ also checks to ensure that the faces of adjacent units are the same size and shape. Several error messages are written if errors are encountered.

LSCAN - This is a logical function that returns a value of true if the specified unit or box type is used in the unit orientation array. A value of false is returned if the unit or box type was not used in the unit orientation array.

PRTJOM - This subroutine prints the geometry data used in the problem.

ARALBA - This subroutine calls PRTLBA for each array that is used in the problem.

PRTLBA - This subroutine is called to print the unit orientation data for each lattice or array that is used in the problem. It may print a warning message associated with one or more lattices.

JOMCHK - The purpose of this subroutine is to perform consistency checks on the geometry data and write the appropriate error messages. See Sect. F11.3.10.2 for additional details.

VOLUME - This subroutine is responsible for calculating the volume of each geometry region and the cumulative volumes for each unit that is used in the problem. See Sect. F11.3.10.3 for additional details.

F11.3.10.1 *Load Data From the Direct-Access File*

This portion of the program loads data from the direct-access data file into permanent memory (Fig. F11.3.22).

LOADIT - This subroutine calls the library routine REED to load the geometry data. If the problem is an array problem (lattice geometry), LODARA is called to load the lattices that are used in the problem and recompute and readjust the array nesting level array and hole nesting level array. If multiple boxes are used in the problem, PRTARA is called to print the unit orientation array for each lattice used in the problem. BOXC is called to load the box correspondence array and LODALB is called to load the non-supergrouped portion of the albedo data.

LODARA - This subroutine is responsible for loading the lattices (unit orientation arrays) that are used in the problem, computing the hole nesting level array, and computing and adjusting the array nesting level array. CLEAR is used to initialize the arrays and REED is used to load the unit orientation arrays. HOLE and LOCBOX are both called by LODARA.

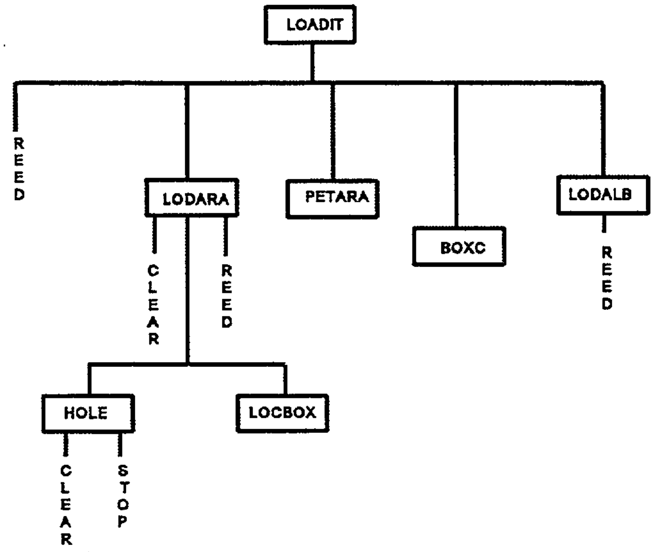


Figure F11.3.22 Flow chart for loading data from direct access

HOLE - This subroutine is called from LODARA to determine which holes occur at the next nesting level and to adjust the array nesting level for arrays that occur in holes. It also checks to ensure that holes are not recursively nested. CLEAR is used for initialization purposes and STOP is called if holes are recursively nested.

LOCBOX - This function is called from LODARA to return the unit or box type at a given position in the unit orientation array.

PRTARA - This subroutine prints a table of the arrays used in the problem. The array number, the number of units in the x, y, and z directions, and the nesting level is printed for each array.

BOXC - This subroutine uses the number of units or box types and the geometry region number corresponding to the first and last geometry region of each unit to generate the box correspondence array which contains the unit or box type number for each geometry region. This is loaded in the appropriate position as it is generated.

LODALB - This subroutine calls the library routine REED to load the pointer and length arrays for the albedo data from the direct-access data file. REED is used to load the nonsupergrouped albedo data, for each albedo that is used, into a temporary array. A loop over the number of angles is then used to load these data into the appropriate arrays. When all of the albedos used in the problem have been processed, REED is called to load the pointer arrays for the cross sections and albedos as well as the arrays defining the albedo to cross-section energy group correlation.

F11.3.10.2 *Check the Geometry Data*

This portion of the program checks the geometry for inconsistencies, surface intersections, and other errors (Fig. F11.3.23).

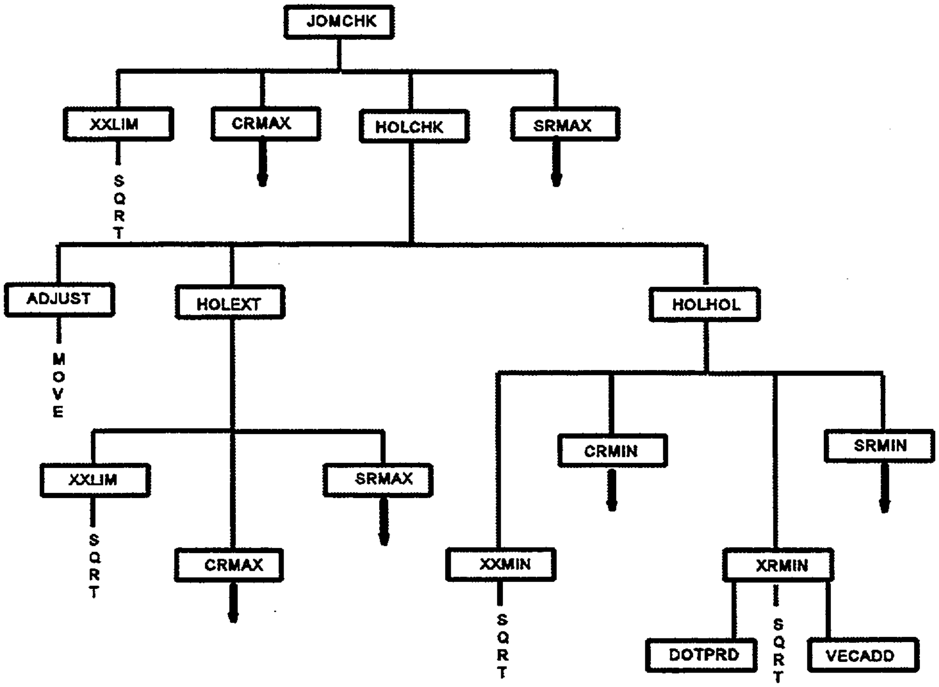


Figure F11.3.23 Flow chart of geometry checking procedure

JOMCHK - This subroutine checks each geometry region to ensure that it does not intersect the next outer region. If an intersection occurs, an error flag is set and an error message is written. JOMCHK checks each surface of the outer region for an intersection. If the surface is a planar surface, the function XXLIM is used to return the farthermost point of the inner region in the direction of the plane specified in the call (i.e., for the -x face, XXLIM returns the most negative x value of the inner region). For a spherical surface, the function SRMAX is used to return the length of the maximum radius vector of the inner region with respect to the origin of the outer region. For a cylindrical surface, the function CRMAX is used to return the length of the maximum radius vector of the inner region with respect to the axis of the outer region. Subroutine HOLCHK is called to check for intersections between holes and other geometry regions, and holes with other holes.

XXLIM - This routine returns the maximum coordinate of an interior region corresponding to a particular face direction (for negative face directions, maximum means most negative).

CRMAX - This function determines the maximum cylindrical radius vector of a geometry region with respect to a given axis, and then returns the magnitude of that vector. See Sect. F11.3.10.2.1 for additional details.

SRMAX - This function determines the maximum radius vector of a geometry region with respect to a given origin, and then returns the magnitude of that vector. See Sect. F11.3.10.2.1 for additional details.

HOLCHK - This subroutine is responsible for ensuring that a hole doesn't intersect any other geometry region. ADJUST is called to adjust the dimensions of the hole with respect to the origin of the unit that contains the hole. HOLEXT is called to check for a hole intersecting the region external to it. HOLHOL is used to check for a hole intersecting the region internal to the region that contains the hole. It is also used for checking for the intersection of two holes.

ADJUST - This subroutine corrects the dimensions of a region represented as a hole, with respect to the origin of the unit that contains the hole.

HOLEXT - This subroutine checks for a hole intersecting the region external to it. The function XXLIM is used to return the farthermost point of the hole in the direction of the plane specified in the call. The function CRMAX is used for a cylindrical surface to return the length of the maximum radius vector of the hole with respect to the axis of the outer region. The function SRMAX is used for a spherical surface to return the length of the maximum radius vector of the hole with respect to the outer region.

HOLHOL - This subroutine is responsible for ensuring that a hole doesn't intersect the geometry region internal to the region that contains the hole. It also checks for holes intersecting each other. The function XXMIN is used to return the nearest point of the hole in the direction of the plane specified in the call. The function CRMIN is used for a cylindrical surface to return the length of the minimum radius vector of the hole with respect to the axis of the surface. The function XRMIN determines the minimum coordinate of a hole with respect to a flat circular face. The function SRMIN is used for a spherical surface to return the length of the minimum radius vector of the hole with respect to the origin of the surface.

XXMIN - This function returns the minimum coordinate of the hole with respect to the face of the cuboid specified in the call. SQRT is utilized in determining the minimum coordinate of spherical or cylindrical holes.

CRMIN - This function returns the magnitude of the minimum cylindrical radius vector of a hole with respect to a given axis. See Sect. F11.3.10.2.1 for additional details.

XRMIN - This function determines the minimum coordinate of a hole with respect to a flat circular face. SQRT is used in processing spherical and cylindrical holes. The functions DOTPRD and VECADD are called to perform vector arithmetic.

DOTPRD - This function returns the dot product of two vectors.

VECADD - This routine returns the vector sum of two vectors.

SRMIN - This function returns the magnitude of the minimum radius vector from the center of a sphere to another geometry region. See Sect. F11.3.10.2.1 for additional details.

F11.3.10.2.1 *Determine Distances.* This section of the geometry checking procedure is utilized in checking for geometry surface intersections (Fig. F11.3.24).

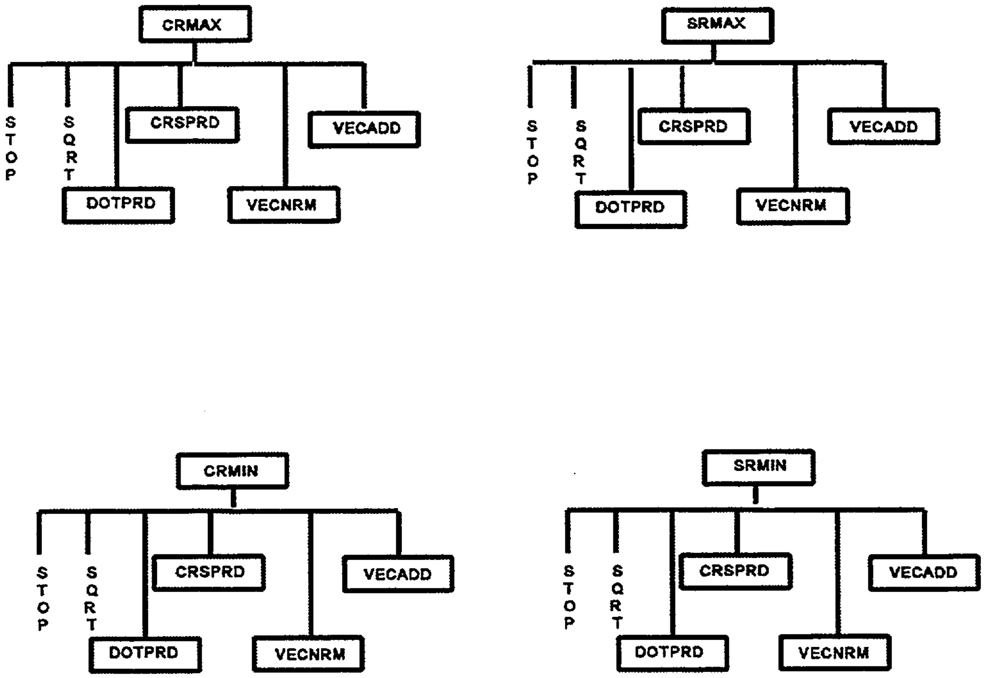


Figure F11.3.24 Flow chart for distance to intersection

CRMAX - This function determines the maximum cylindrical radius vector of a geometry region with respect to a given axis, and then returns the magnitude of that vector. Depending on the geometry type, CRMAX may call CRSPRD to generate the cross product of two vectors, DOTPRD to generate the dot product of two vectors, VECADD to add two vectors together and VECNRM to multiply a vector by a scalar. STOP is called if geometry inconsistencies are encountered.

SRMAX - This function determines the maximum radius vector of a geometry region with respect to a given origin, and then returns the magnitude of that vector. Depending on the geometry type, SRMAX may call CRSPRD to take the cross product of two vectors, DOTPRD to take the dot product of two vectors, VECADD to add two vectors together, and VECNRM to multiply a vector by a scalar. STOP is called if geometry inconsistencies are encountered.

CRMIN - This function determines the minimum cylindrical radius vector of a geometry region with respect to a given axis and returns the magnitude of that vector. The geometry type determines which functions will be used to calculate the minimum radius vector. DOTPRD is used to generate the dot product of two vectors, CRSPRD generates the cross product of two vectors, VECNRM multiplies a vector by a scalar, and VECADD adds two vectors together. STOP is called if geometry inconsistencies are encountered.

SRMIN - This function determines the minimum radius vector of a geometry region with respect to a given origin (center of a sphere) and returns the magnitude of that vector. Depending on the geometry type, SRMIN may call DOTPRD to take the dot product of two vectors, CRSPRD to take the cross product of two vectors, VECNRM to multiply a vector by a scalar, and VECADD to add two vectors. STOP is called if geometry inconsistencies are encountered.

CRSPRD - This routine is a utility routine to generate the cross product of two vectors.

DOTPRD - This function returns the dot product of two vectors.

VECADD - This subroutine returns the result of adding two vectors.

VECNRM - This routine scales a vector by a constant.

F11.3.10.3 *Calculate Volumes*

This portion of the program is responsible for calculating the volume of each region used in the problem, the cumulative volumes for each unit used in the problem, the number of times each unit was used in the problem, and the total volume of each region summed over all occurrences.

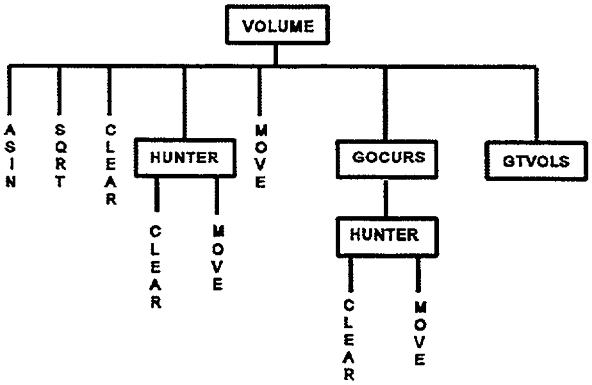


Figure F11.3.25 Flow chart for calculating volumes

VOLUME - This subroutine calculates the volume of each region for every unit that is used in the problem. It then calculates the cumulative volumes for each unit. The intrinsic functions, ASIN, which returns the arc sign of a variable, and SQRT, which determines the square root of an expression, are used when calculating the volume of a hemicylinder. CLEAR is used to initialize arrays. If an external reflector is present, HUNTER is called to determine the number of times each array and/or hole is used in the reflector. GOCURS is used to determine the number of times each unit, array and hole is used in the problem. GTVOLS is called to calculate and print the number of occurrences for each unit and the corresponding total volumes for the entire system. CLEAR is called to initialize the array for the total volume of each mixture used in the problem prior to calculating and printing those totals.

HUNTER - This subroutine determines the number of times each unit, array and hole is used. CLEAR is used to initialize storage arrays for the present hole level and the next hole level. MOVE is used to move the storage arrays.

GOCURS - This subroutine loops over the array size and calls HUNTER to determine the number of times each unit, lattice or array, and hole is used in the problem.

GTVOLS - This subroutine calculates the total volume of each region for the entire problem by multiplying the volume of the region by the number of times the region is used in the problem.

F11.3.11 LOAD BIASING OR WEIGHTING DATA

This portion of the program is executed only if the input parameter data contain PWT=YES, as described in Sect. F11.4.3. If biasing data are to be printed, the program loads and prints the average weight array (Fig. F11.3.26).

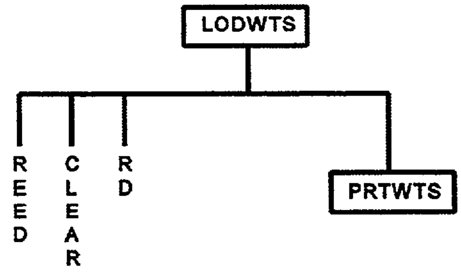


Figure F11.3.26 Flow chart for loading biasing data

LODWTS - This subroutine is responsible for printing the bias IDs versus the material IDs used in the problem and for loading and printing the biasing or weighting data. The library routine REED is called to load the bias IDs used in the problem from the direct-access device. The number of sets of data that will fit in the available memory is calculated. Then the library routine CLEAR is used to initialize that space. RD is used to load the data that will fit and PRTWTS is called to print them. Then the entire process is repeated until all the biasing or weighting data used in the problem have been loaded and printed.

PRTWTS - This subroutine prints the group-dependent weight average array for each biasing region in a compact fashion.

F11.3.12 GENERATE PLOT

This portion of the program generates character and color plots of a 2-D slice through the geometry (Fig. F11.3.27). As many plots as are desired can be plotted.

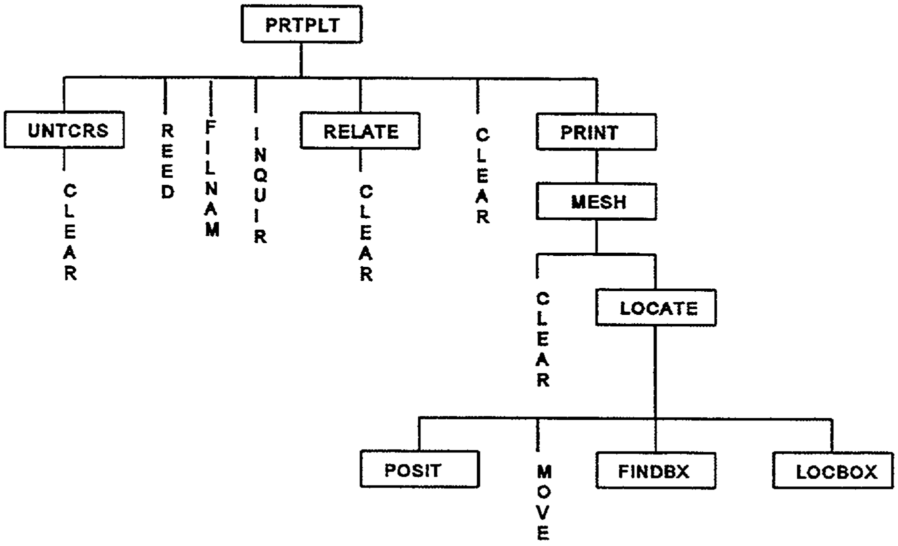


Figure F11.3.27 Flow chart for plots

PRTPLT - This subroutine controls the generation of the character or color plots. The library routine CLEAR is called to zero the necessary space. Subroutine UNTCRS is called to generate a unit correspondence array. REED is used to load the plot data from direct access. INQUIR is called to return the value of the next direct-access record. The plot title is printed and subroutine RELATE is called to print a heading for the symbol map and print the symbol map. The plot coordinates, direction cosines and number of symbols across and down the plot as well as the step intervals are printed. Then subroutine PRINT is called to generate the actual plot.

UNTCRS - This subroutine calls CLEAR to zero the unit correspondence array. It then loads the appropriate unit number in the appropriate position of the array.

RELATE - This subroutine calls CLEAR to zero the reverse correspondence array. It then prints the plot header telling whether a mixture, unit, or bias ID map is to be printed. It loads the array that correlates the symbols to be used in the plot with the mixture numbers, bias ID numbers, or unit numbers that were used in the problem. The array is then printed to facilitate user interpretation of the plot.

PRINT - This subroutine determines the number of pages that will be needed to output the plot. Subroutine MESH is called to load the appropriate mixture numbers, unit numbers or bias ID numbers for each line of the plot. Then PRINT outputs the line of symbols/colors corresponding to them.

MESH - This subroutine loads an array that contains the appropriate mixture number, unit number or bias ID number for each character in a line. CLEAR is called to initialize arrays if nested holes or nested arrays are present in the problem. LOCATE is called to determine the geometry region and unit or box type that contains each mesh point. MESH then loads the mixture number, unit number, or bias ID number.

LOCATE - This subroutine is responsible for determining the geometry region for each mesh point in the plot. Subroutine POSIT is called to determine the region that contains the specified mesh point. If array data are used, MOVE is called to load the current array data into the array stack. If the mesh point is within an array, FINDBX is called to determine the position within the lattice or array that contains the mesh point. LOCBOX is then called to determine the unit or box type that contains the mesh point. POSIT is used to determine the geometry region that contains the mesh point. If the mesh point is in a region that contains a hole, the coordinates of the mesh point are translated to the hole and POSIT is called again.

FINDBX - This subroutine locates the position in an array that contains a specified point.

LOCBOX - This function returns the unit or box type at a given position in an array.

POSIT - This subroutine determines the region within a unit that contains a specified point.

F11.3.13 PROCESS HISTORIES BY SUPERGROUP

This section of the program is where the tracking of the individual histories is done, one supergroup at a time (Fig. F11.3.28).

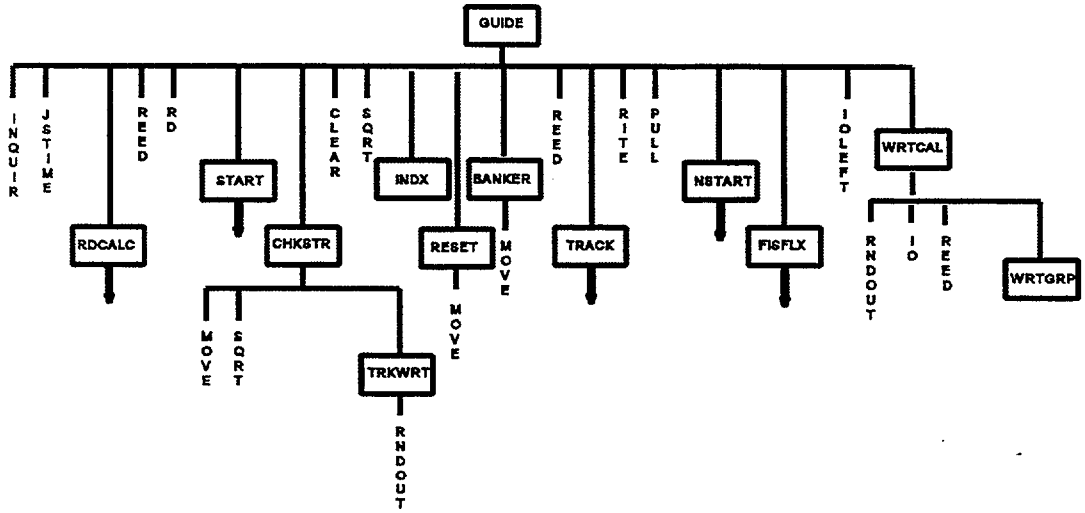


Figure F11.3.28 Flow chart of tracking routines

GUIDE - This subroutine controls the tracking procedure. It loads the calculated data for a restarted problem, calls subroutine START to obtain the initial source distribution, calls CHKSTR to load the initial starting distribution in COMMON /NUTRON/ and prints the starting points as requested by the data. The library routine CLEAR is called several times to initialize arrays. IOLEFT and JSTIME are called to initialize the I/Os and time for the tracking procedure. INQUIR is used to return the value of the next direct-access record. SQRT is used in estimating the lower limit of the 99% confidence interval of the sample distribution of k-effective.

The heart of subroutine GUIDE is a loop over generations, from the starting generation to the number of generations requested. Subroutine RESET is called to accumulate the fission source and the source vectors for matrix k-effective. It also counts the number of histories in each supergroup and determines the supergroup with the largest number of histories. Then subroutine BANKER is called to sort the histories by supergroup, loading the largest supergroup at the top of the bank. GUIDE calculates pointers for the supergrouped data and loads the supergrouped data using the library routine REED. TRACK is called to do the actual tracking. The library routine RITE writes the calculated supergrouped data on the direct-access supergroup data file. When all the supergroups have been processed, NSTART is called to provide the fission source for the next generation. Then FISFLX is called to calculate matrix information and statistics for the calculated data. GUIDE then checks to be sure sufficient time and I/Os remain to ensure that another generation can be processed. If another generation cannot be processed and restart data are to be written, WRTCAL is called to write the calculated data on the restart data file. A message is printed by GUIDE stating the reason for terminating the calculation.

GUIDE is a very important subroutine in KENO V.a. The following table is provided to assist in understanding the functions performed by the subroutines called by GUIDE:

SUBROUTINE	FUNCTION	CONDITION
INQUIR	Set direct-access pointer	Always
JSTIME	Monitor time usage	Always
RDCALC	Load calculated data for restarting a problem	If the problem is restarted
REED	Load data from direct-access supergroup data file	Always
RD	Load data from direct-access	Always
START	Provide initial source	If the problem is started with the first generation
CHKSTR	Print starting points	Always
INDX	Locate cross-section index	Always
RESET	Count histories/supergroup	Always
BANKER	Sort the neutron bank	Always
TRACK	Track individual histories	Always
RITE	Write data on direct access	Always
NSTART	Provide fission source for the next generation	Always
FISFLX	Calculate statistics	Always
IOLEFT	Monitor I/O usage	Always
PULL	Terminate problem	If excessive time is used
WRTCAL	Write data on restart file	If a restart data file is to be created

INQUIR - This library routine returns the value of the next direct-access record. It is called to obtain the pointer for the cross-section data.

JSTIME - This library routine is called several times from GUIDE for timing purposes.

RDCALC - This subroutine is called only if the problem is being restarted at a generation greater than 1. Its purpose is to read data from the restart data file and write the supergrouped data on the direct-access data file. See Sect. F11.3.13.1 for additional information.

REED - This library routine is called early in GUIDE if a problem is not being restarted in order to load data that are needed to create the initial source distribution. REED is called to load supergrouped data for the supergroup being processed from the direct-access supergroup file.

RD - This library routine is called to load the fission spectrum for use in creating the initial source distribution.

START - This subroutine is responsible for creating the initial source distribution. See Sect. F11.3.13.2 for additional details.

CHKSTR - This subroutine calls the library routine MOVE to load the initial source distribution into COMMON /NUTRON/. If the starting points are to be printed, TRKWRT is called to print them using the debug tracking format.

TRKWRT - This subroutine is called from CHKSTR to print information about the current status of the current neutron in the debug tracking format. When TRKWRT is called from CHKSTR, the information of interest is the position at which the neutron was started. RNDOUT is called from TRKWRT to make the current random number available for printing.

CLEAR - This library routine is called to initialize the fission density array and the arrays that hold the matrix k-effective and associated statistics.

INDX - This subroutine is called only if the average number of neutrons per fission and the average energy at which fission occurs are to be calculated. (See Sect. F11.4.3, NUB=.) INDX determines the position of the fission cross section in the extra 1-D cross-section array. The fission cross section is required for calculating the average energy at which fission occurs.

RESET - This subroutine is called to accumulate the fission source and the source vectors for the matrix k-effective and to count the number of histories in a supergroup.

BANKER - This subroutine is responsible for sorting all the particles in the current supergroup into the top of the neutron bank and all other particles into the bottom of the bank.

TRACK - This subroutine is responsible for the actual tracking of each individual history. See Sect. F11.3.13.3 for a detailed description.

RITE - When the program returns from TRACK, RITE is called to write the calculated supergrouped data on the direct-access supergroup data file.

PULL - This library routine is called from GUIDE to set a time interval that results in a nonstandard return if that time interval is exceeded. This step is for the purpose of preventing the program from looping indefinitely. PULL is called later in GUIDE to reset the time interval as appropriate.

NSTART - This subroutine is called from GUIDE to provide the fission source for the next generation. See Sect. F11.3.13.4 for specific details.

FISFLX - This subroutine calculates statistics for k-effective, the matrix k-effective, fissions, absorptions, leakages, and fluxes. Section F11.3.13.5 contains additional details.

IOLEFT - This library routine is called to determine if sufficient I/Os remain to allow processing another generation.

WRTCAL - If a restart data file is to be created, this subroutine writes the calculated information on the restart data file with a frequency specified by the parameter data (Sect. F11.4.3, RES=). RNDOUT is called to preserve the random number. Then the generation number, random number, number of histories per generation, number of energy groups, banked information, some common information and all the k-effectives calculated to this point are written on the restart data file. The neutron bank and, if requested, the fission densities are written. If matrix k-effective information is requested, IO is used to write it on the restart data file. Then WRTCAL loops over the number of supergroups, calculating pointers, using REED to load data from the direct-access supergroup file and calling WRTGRP to write the group-dependent calculated information on the restart data file. When all the data have been written, a message is printed.

WRTGRP - This subroutine writes calculated data (leakages, absorptions, fissions, and if requested, fluxes) for each energy group on the restart data file.

F11.3.13.1 *Load Calculated Restart Data*

This section of the program loads the calculated data such as k-effectives and fluxes from the restart data file if the starting generation number is greater than 1 (Fig. F11.3.29).

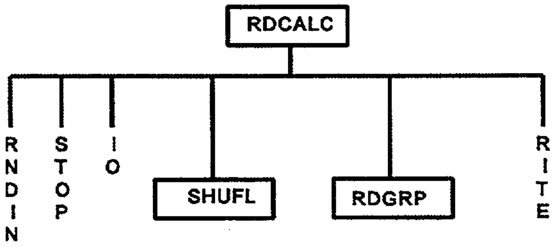


Figure F11.3.29 Flow chart for loading calculated restart data

RDCALC - This subroutine is called from GUIDE if the starting generation number is greater than 1. This indicates that a calculation is to be restarted using the starting generation number as the first generation to be processed. Thus all the results that were calculated in a previous run must be loaded from the restart data file to continue the calculation. RDCALC reads the previously calculated data from the restart data file and checks for consistency against parameters that were entered as input data. Appropriate messages are printed if inconsistencies are encountered. RNDIN may be called to load a new random number. STOP is called if more storage is needed to hold the neutron bank. IO is used to load data from the restart data file and SHUFL is called to sort and store the neutron bank. Pointers are calculated for the supergrouped restart data, and RDGRP is called to read the group-dependent data from the restart data file. RITE is used to write the restart data on the supergroup data file.

RNDIN - This library routine is called to load a new random number for use with the restarted problem if a random number was entered in the parameter input data.

STOP - This library routine is called to write an error message and stop if the available storage is not large enough to hold the neutron bank.

IO - This library routine is used to load matrix k-effective information if it is to be calculated.

SHUFL - This subroutine is called to sort and store the neutron bank.

RDGRP - This subroutine is used to load the supergrouped data from the restart data file. This includes leakages, fissions, absorptions and fluxes.

RITE - This library routine is used to write previously calculated results on the direct-access supergroup data file so the problem can be restarted properly.

F11.3.13.2 *Generate Initial Source Distribution*

This portion of the program is responsible for generating the initial source distribution (Fig. F11.3.30) in accordance with information specified in the start data (see Sect. F11.4.8).

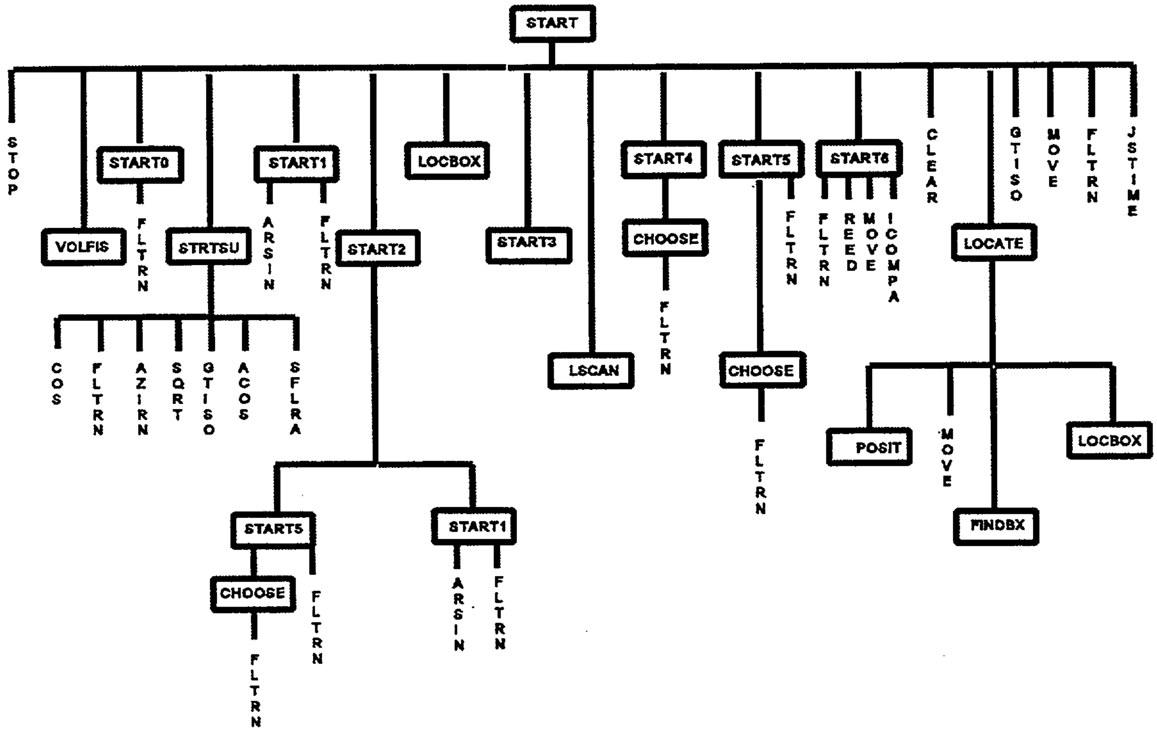


Figure F11.3.30 Flow chart for providing initial source distribution

START - This subroutine is responsible for generating the initial source distribution. If fissile material is not used in the problem, STOP is called to write a message and terminate the problem. Appropriate messages are printed if the specified start type is incompatible with the geometry configuration. Subroutine VOLFIS is called to calculate the volume fraction of fissile material. STOP is called to print error messages if the start data are geometrically invalid or the fraction started as a spike is invalid. LSCAN is used to determine if the specified unit is in the global array. Then START0, STRTSU, START1, START2, START3, START4, START5, or START6 is called to generate starting points having the characteristics specified in the start data (see Sect. F11.4.8). Subroutine LOCBOX is called after both START2 and START3 to determine the unit number at the specified position in the array. Subroutine LOCATE then determines the geometry region that contains the specified point by utilizing POSIT, FINDBX and LOCBOX. FINDBX is called to locate the position within the array that contains the starting point. LOCBOX is called to determine the unit number located at that position in the array. The starting point is translated to the coordinate system of the unit and POSIT is called to determine which region within the unit contains the starting point. A check is then made to be sure the region contains fissile material. If it does not, the point is discarded. If the region contains fissile material, GTISO is called to provide the initial direction cosines. START uses FLTRN to set the initial energy group and MOVE is called to load the initial data for the history into the neutron bank. JSTIME is called to be sure the allowed time is not exceeded. If it is, or if the required number of source neutrons have been generated, the starting is terminated. If too few initial source neutrons exist, FLTRN and MOVE are used to fill the remaining starting positions from those that were generated. A message to that effect is then printed.

STOP - This library routine is called from START to write error messages.

VOLFIS - This subroutine determines the volume fraction of fissile material in an unreflected array or an array whose reflector material is not fissile. It determines the volume fraction of fissile material in the system for single unit problems and reflected problems having fissile reflector material. If the volume fraction of fissile material is found to be zero, an error message is written and execution is terminated.

LSCAN - This logical function returns a value of true if the specified box type is used in the problem.

START0 - This subroutine is called from START to generate a uniform initial source distribution in a cuboidal volume. This is accomplished by choosing points uniformly throughout the volume by using the library routine FLTRN, and discarding points that do not occur in fissile material. See Sect. F11.4.8 and Table F11.4.6 for assistance in modifying the default boundaries over which the starting points are chosen.

STRTSU - This subroutine is called to generate a uniform initial source distribution for a single unit problem or a reflected problem for which the user has specified that the reflector be included in the starting distribution. FLTRN is used to choose points uniformly throughout a cuboidal volume. AZIRN and FLTRN are used to choose points uniformly throughout a cylindrical or hemicylindrical volume. GTISO and FLTRN are used to choose points uniformly throughout a spherical or hemispherical volume.

AZIRN - This library routine provides the sine and cosine of a random azimuthal angle.

GTISO - This library routine provides the direction cosines of an isotropically distributed random direction. It is used to generate an isotropic source distribution.

START1 - This subroutine is used to provide starting points chosen from a cosine distribution. The library routines ARSIN and FLTRN are used to provide the cosine distribution. See Table F11.4.6 for details of the various initial starting distributions.

ARSIN - This library routine is the arcsine function, y = arcsine(x), and is used in generating a cosine source distribution.

START2 - This subroutine starts a specified fraction of the initial source distribution uniformly in fissile material in the unit located at a specified position in the global array. The remainder of the initial source is chosen from a cosine distribution as described in Sect. F11.4.8. START5 is called to determine the source points located in the unit at the specified location, and START1 is called to choose points from a cosine distribution.

LOCBOX - This function is called to return the unit or box type at a given position in a given array.

START3 - This subroutine starts all the initial source neutrons at a specified point within the unit located at a specified position in the global array.

START4 - In this subroutine, a unit type is specified for starting the initial source distribution. A uniform sampling is made over the global unit orientation array to locate units of the specified type, and start the initial source neutrons at a specified positions within these units. CHOOSE is called to determine the positions within the global array that are occupied by the specified unit type.

CHOOSE - This subroutine locates the positions of a specified unit within the global unit orientation array by randomly choosing positions in the array and discarding them if the specified unit was not at that position. The library routine FLTRN is used to randomly choose positions in the unit orientation array.

START5 - This subroutine starts the initial source neutrons uniformly in fissile material within a specified unit type. A uniform sampling is made over the global unit orientation array to locate units of this type by utilizing subroutine CHOOSE.

START6 - This subroutine starts the initial source neutrons at points specified by the user. These points must be specified relative to the origin of the global array. REED is called to load the start type 6 data from the direct-access device. MOVE is called to move the start data into common. FLTRN is used to randomly supply additional starting points from existing ones. ICOMPA is used to determine if the starting point is identical to the previous one. See Table F11.4.6 for start data specifications.

LOCATE - This subroutine determines the geometry region that contains the specified starting point. FINDBX is used to determine the point's location in an array. LOCBOX determines the unit number at that location and POSIT determines the region within the unit.

POSIT - This subroutine determines the region within the unit that contains the specified position.

FINDBX - This subroutine locates the position in an array that contains a specified point.

LOCBOX - This function returns the unit or box type for a specified position in an array.

MOVE - This library routine is called from START to load data from COMMON /NUTRON/ into the neutron bank. It is also used to pad the neutron bank to provide enough starting positions if too few were initially created.

FLTRN - This library routine is called from various subroutines during creation of the initial source distribution, to return a random number between zero and 1. START calls FLTRN to aid in choosing the initial source neutrons. It is also called if too few starting positions were generated. It is used to randomly choose from the initially created starting positions to pad the neutron bank until sufficient starting points exist.

JSTIME - This library routine is used for timing purposes.

F11.3.13.3 *Track Individual Histories*

This section of the program does the actual tracking of the individual histories.

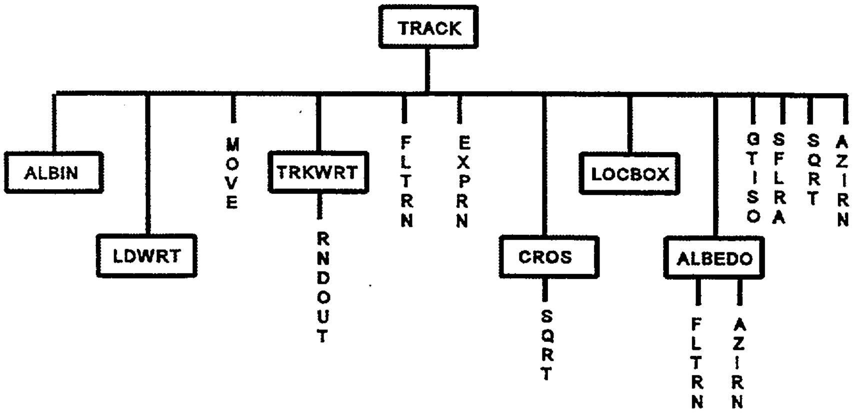


Figure F11.3.31 Flow chart for tracking individual histories

TRACK - This subroutine is called by GUIDE to accomplish the actual tracking of the individual histories. Each history is tracked, and its contributions to the various calculated results are tabulated, until it escapes from the system or is killed via Russian roulette. If a history changes supergroups as a result of a collision or an albedo reflection, it is stored in the neutron bank. In the course of tracking a history, an initialization call is made to subroutine ALBIN if differential albedo boundary conditions are utilized in the problem. LDWRT may be called to print debug information and, if history tracks are to be printed, TRKWRT is called from strategic locations throughout TRACK to provide pertinent information about the history as it moves through the tracking process. The library routine MOVE is utilized throughout the tracking procedure to move data in and out of storage arrays and commons. FLTRN is used to provide random numbers used in playing Russian roulette, processing downscatters, picking fission points, picking scattering angles and determining the fission energy group. EXPRN provides a random number, selected from an exponential distribution, to be used as the number of mean-free paths a history can traverse. CROS is called to determine if a boundary crossing has occurred. LOCBOX is called to determine the unit at a specified location in an array. ALBEDO is called to process differential albedo boundary conditions. The library routine GTISO is used to provide direction cosines from an isotropic distribution. SFLRA provides a random number between -1.0 and 1.0 for use in processing an isotropic scattering. SQRT is used in calculating the direction cosines of a history after it has a collision. AZIRN provides the sine and cosine of a random azimuthal angle for use in the anisotropic scattering treatment of a collision.

ALBIN - This subroutine contains the entry point ALBEDO. An initialization call is made to ALBIN by TRACK.

LDWRT - This subroutine prints debug information that is useful only for a programmer. It is called if BUG = YES is specified in the parameter data (see Sect. F11.4.3). In normal operation, this subroutine should never be called.

MOVE - This library routine is utilized frequently in TRACK to move data in and out of storage arrays and commons.

TRKWRT - This subroutine is called from various locations in TRACK to print information about the current neutron as it is being processed. RNDOUT is called from TRKWRT to make the current random number available for printing.

FLTRN - This library routine provides a random number between zero and 1. TRACK utilizes these random numbers for playing Russian roulette, processing downscatters, picking fission points, determining scattering angles and determining the fission energy group.

CROS - This important subroutine is responsible for processing both inward and outward crossings (i.e., it determines when a history has moved out of one geometry region into another). It determines if a crossing has actually occurred, and if it has, the coordinates of the crossing are upgraded to give the crossing point. The fraction of the path length used is also determined.

CROS consists of eight major sections. Four geometrical packages exist; one each for cuboids, cylinders, spheres and hemispheres. Hemicylinders are special adaptations of the cylinder package. Each geometrical package contains programming for inward crossings and for outward crossings. The variable N indicates the type of crossing being processed; N ≤ 0 checks for an inward crossing and N > 0 checks for an outward crossing. The variable M is the crossing indicator, indicating whether or not a real crossing occurs. M = 0 means a real crossing does not occur. M = 1 indicates a successful crossing. SQRT is the only library routine utilized in CROS. It is used in the cylinder and sphere packages to solve the quadratic equation for the fraction of the path length that is used.

LOCBOX - This function returns the unit or box type for a specified position in an array.

ALBEDO - ALBEDO is an entry point in subroutine ALBIN. It is responsible for processing a differential albedo reflection. The direction cosines for the face where the albedo reflection occurs are loaded, and the incident angle and the albedo energy group corresponding to the incident energy group are determined. The position of the albedo energy group within the supergroup, and the first cross-section energy group and the number of cross-section energy groups corresponding to it are determined. This is used to calculate the new cross-section energy group of the history. Then the returning angle and direction cosines are calculated. The history's weight is then corrected for the weight lost in the albedo portion of the problem. The weight lost in the albedo reflection is summed.

FLTRN - This library routine is called from ALBEDO to return a random number between zero and one that is used to select (1) the albedo energy group corresponding to the input energy group, (2) the returning albedo energy group, (3) the returning energy group corresponding to the returning albedo energy group and (4) the returning angle.

EXPRN - This library routine is called from TRACK to provide the number of mean-free paths to the next collision. This random number is picked from an exponential distribution.

GTISO - This library routine is called from TRACK to provide direction cosines from an isotropic distribution. These direction cosines are utilized in processing isotropic scattering.

SFLRA - This library routine is called from TRACK to provide a random number between -1.0 and 1.0 for use in processing anisotropic scattering.

SQRT - This library routine is utilized by TRACK to calculate the direction cosines of a history that has experienced a collision.

AZIRN - This library routine is called from ALBEDO to return the sine and cosine of a random azimuthal angle which are used in determining the direction cosines of the returning history.

The remainder of this section is devoted to the logical flow of subroutine TRACK, as illustrated by Fig. F11.3.32. The portions of TRACK performing specialized functions are denoted in the text by a descriptive name enclosed in quotes to distinguish them from subroutine names. These descriptive names correspond to the functions depicted in the flow chart.

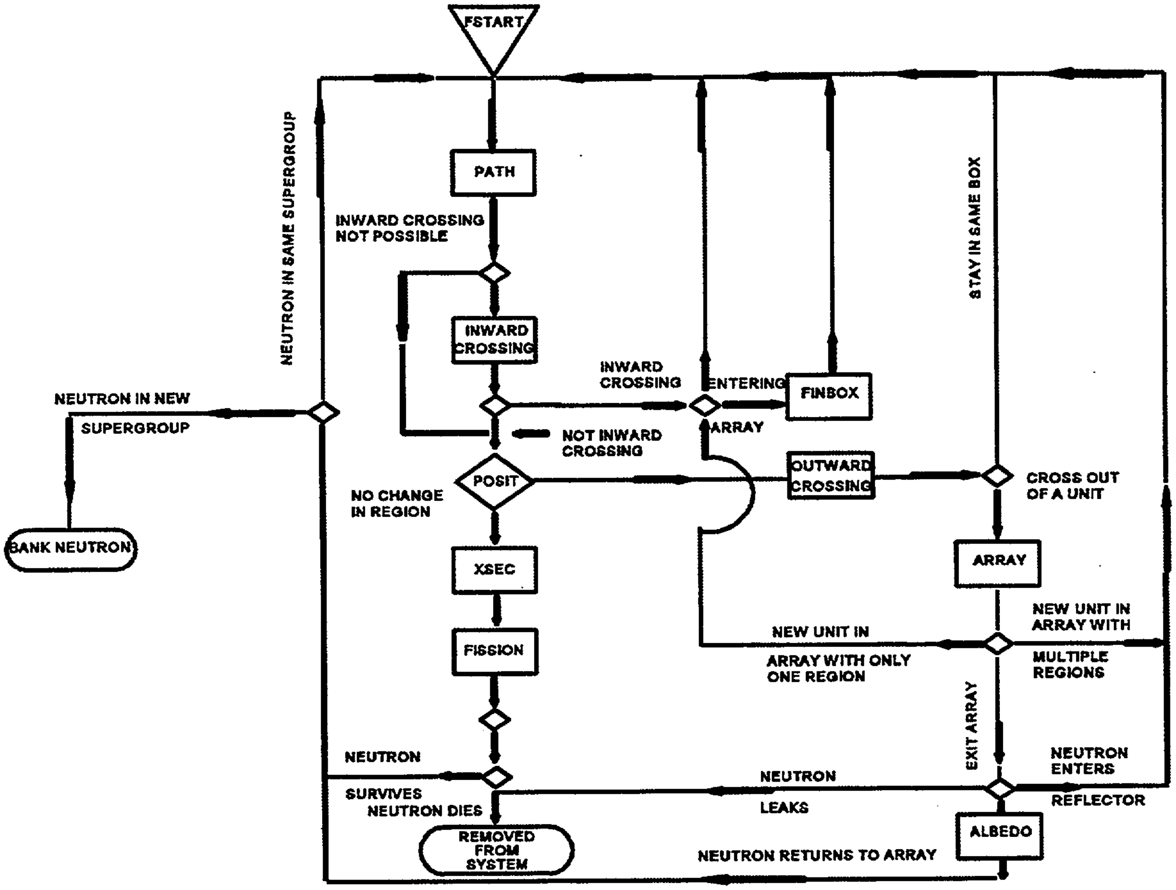


Figure F11.3.32 Logic flow chart for subroutine TRACK

A brief discussion of the logical program flow through subroutine TRACK follows. If differential albedos are used, an initialization call is made to ALBIN. If debug print was specified in the parameter data, LDWRT is called. The library routine MOVE is used to load data from the neutron bank into COMMON /NUTRON/ and various flags are set. Then the fission source portion of TRACK, denoted "FSTART," initializes and sets information necessary for processing the history. The "PATH" portion of TRACK sets the path length and the end point of the path. The "INWARD" portion of TRACK determines if an inward crossing is possible and, if it occurs, the point of the boundary crossing. If the history is entering an array from an external region, the "FINBOX" portion of the code determines the history's position relative to the origin of the unit it is entering. If an inward crossing was not possible or did not occur, the "POSIT" portion of TRACK determines if the end point of the path is in the same region. If it is, a collision occurs and is processed in the portion of TRACK denoted as "XSEC." Based on the weight of the history, it is split and banked and/or Russian roulette is played. If the history survives Russian roulette, it is scattered and fission points may be generated in the FISSION portion of TRACK, and if it now falls in a different supergroup, it is banked. Otherwise it is retained in the system and processing returns to "PATH."

If "POSIT" determines that the end point of the path is in a different region, the "OUTWARD" portion of TRACK processes an outward crossing. If the history remains within the same unit, processing returns to "PATH" to continue as before.

If, after an outward crossing, the new region indicates that the history has crossed out of the unit it was in, the "ARRAY" portion of TRACK determines if the history is exiting from the array or entering a new path. If the history enters a new unit, processing returns to "PATH" and continues as before.

If the history is exiting from the array, it can (1) enter the reflector, in which case processing returns to "PATH," (2) leak from the system, thus terminating that history or (3) undergo albedo reflection. The "ALBEDO" portion of TRACK can process a specular or mirror image reflection, a periodic reflection, or a differential albedo. For a mirror image or specular reflection, the history is returned at the point it exited with the weight and energy unchanged. The sign of the direction cosine perpendicular to the reflecting face is reversed. For a periodic reflection, the history is moved to the opposing face of the system and the weight, energy and direction cosines remain unchanged. For a differential albedo reflection the history returns at the point it exited, and subroutine ALBEDO determines the returning weight, energy and angle. The history is then returned to the PATH portion of TRACK to continue processing.

"FSTART" - This portion of TRACK calls the library routine MOVE to load information pertaining to the history to be tracked. Then logical flags are set to indicate if the history is in an array and whether or not the history is a split neutron. Variables are initialized and if the history tracks are to be printed, TRKWRT is called to print information about the history. If the history is the result of an albedo reflection that resulted in the history moving to a new supergroup, and the weight is large enough, the history proceeds to the "PATH" portion of TRACK. Otherwise Russian roulette is played. If it survives, the weight is set to the average weight and the history proceeds to the "PATH" portion of TRACK. If the history is a split neutron, variables are initialized and the history proceeds to the "XSEC" portion of TRACK to undergo the collision process.

"PATH" - This portion of TRACK determines the path length. If all the path length has been exhausted, the library routine EXPRN is used to define a new number of mean-free paths from an exponential distribution. If the region contains a void, the distance traveled is set to the maximum chord length of the system. Otherwise the distance traveled is equal to the remaining path length, divided by the macroscopic total cross section of the mixture contained in the region. The end point of the path is determined from the starting coordinates, the distance traveled and the direction cosines. If the starting and ending coordinates are identical in any given direction, the end point is changed by a very small amount in the proper direction.

"INWARD" - This portion of TRACK decides whether or not an inward boundary crossing is possible, and if it is, calls CROS to determine if a crossing actually occurs and the coordinates of the crossing. Each time CROS is called, the library routine MOVE is used to load the geometry region dimensions of the possible new region into COMMON/ NUTRON/ to be used by subroutine CROS. The crossing indicator is set to instruct CROS to check for an inward crossing and the possible new region type is set. If the history tracks are to be printed, TRKWRT is called to print information pertinent to the crossing. Subroutine CROS sets a crossing indicator that informs TRACK whether a crossing occurs as well as determining the coordinates of the crossing. If a crossing occurs, the appropriate contributions are summed into the flux and neutron age. Then the number of mean-free paths remaining for the history is determined.

When checking for an inward crossing, a check is made to determine if the history is in the innermost region of a unit and whether the innermost region contains holes. An inward crossing is not possible if the history is in the innermost region and that region does not contain holes. If holes are present in the innermost region, each hole in the region must be checked for a crossing.

If the history is not in the innermost region, an inward crossing is possible if holes are present, even if the last crossing was outward. CROS determines if an inward crossing actually occurs. If a crossing does not occur and no holes are in that region, tracking proceeds to the "POSIT" portion of TRACK. If a crossing did occur and holes are not present in that region, the flux and neutron age are updated and tracking proceeds to the "FINBOX" portion of TRACK. Tracking proceeds to the "POSIT" portion of TRACK if an inward crossing is not possible and no holes are in the region.

If holes are present in the region, after checking for an inward crossing, each of the holes is checked for an inward crossing. This involves transforming the coordinates of the history to the coordinate system of the hole and calling CROS to determine if the history crosses into the outer region of the hole. After repeating this procedure for each hole in the region, the crossing with the shortest distance is selected as the actual crossing. If the crossing is into a hole, the coordinates of the history are transformed to the coordinate system of the hole and tracking proceeds to the "FINBOX" portion of TRACK. The correct new region number is set whenever an inward crossing occurs.

"FINBOX" - This portion of subroutine TRACK is responsible for determining when a history enters a new array and for performing the transformation of coordinates whenever a history travels from a surrounding region into an array. It also determines the location within the unit orientation array of the unit containing the history. This information is used to determine the unit type, and the first and last regions of the unit.

"POSIT" - This portion of TRACK determines if the path ends in the same region in which it originated.

"OUTWARD" - An outward crossing occurs when the "POSIT" portion of TRACK determines that the history is entering a different region. The library routine MOVE loads the dimensions of the new geometry region into COMMON /NUTRON/ for use by subroutine CROS. The crossing-type indicator is then set to instruct CROS to process an outward crossing. If history tracks are to be printed, TRKWRT is called to print information pertinent to the outward crossing. CROS determines the coordinates of the boundary crossing. The contribution of the history is summed into the flux and neutron age and the number of mean-free paths remaining for the history is calculated. The region number is incremented. If the old region was not the last region in the unit, a logical flag is set to avoid some of the checking for an inward crossing for the next boundary crossing. Tracking then proceeds to the "PATH" portion of TRACK. The region number is set to the last region in the unit, if the old region was the last region in the unit. If the problem is a single unit problem or the history is exiting the external reflector, the history proceeds to the "ALBEDO" portion of TRACK to leak from the system or process an albedo reflection. If the history has exited a unit in an array, tracking proceeds to the "ARRAY" portion of TRACK to continue the tracking process.

"COLLISION" - When a history has a collision, the processing is done in this portion of TRACK. If fluxes are to be calculated, the new contribution is summed in. The age of the history is summed, the remaining path length is set to zero, the absorption weight, fission weight and the contribution to the average number of neutrons produced per fission and the self-multiplication of the unit are calculated, based on the macroscopic cross-section data and the weight of the history. The weight of the history is then redefined to be the weight times the macroscopic nonabsorption probability. If the history tracks are to be printed, TRKWRT is called to print information pertinent to the collision process. If matrix k-effectives are to be calculated, the fission weight is summed into the proper arrays. If the weight of the history exceeds the weight at which splitting occurs, a check is made to assure the neutron bank has adequate space for another history. If it is full, message K5-128 is written. A maximum of 50 of these messages are printed in a generation before an error flag is set, and execution is terminated. If the bank can hold another history, the weight of the history is halved and the neutron counter is incremented. If the history tracks are to be printed, TRKWRT is called to print information pertinent to the split neutron. The library routine MOVE is used to store the split neutron in the neutron bank. The history cycles through the checking and splitting process until its weight is less than the weight at which splitting occurs. Then the weight is checked to see if Russian roulette should be played. If it is played and the history survives, the weight is set to the average weight. If Russian roulette is not played, the weight remains unchanged. In both cases, the new energy group is computed and a check is made to determine if the history undergoes anisotropic scattering. If it does, the azimuthal angle is chosen using AZIRN and the sine and cosine of that angle are returned to be used for calculating new direction cosines. If the history does not undergo anisotropic scattering, the new direction cosines are chosen from an isotropic distribution using GTISO. This completes the "COLLISION" portion of TRACK.

"FISSION" - This portion of TRACK is responsible for generating and storing the fission source resulting from a collision. In the "COLLISION" portion of TRACK, the fission weight is defined as the weight of a history times the macroscopic production probability. If the fission weight of a history is greater than zero, the "FISSION" portion of TRACK is executed. To ensure generating enough fission source points to maintain an adequate representation of the true distribution, a minimum production factor is defined at the beginning of each generation to be

$$\frac{3.0\bar{k}}{\sqrt{FG}},$$

where $\bar{k}$ is the running average value of the k-effective through the current generation and FG is the number of histories per generation. This represents an estimate of the lower limit of the 99% confidence interval for the distribution of the generation k-effective. Experience indicates that using this factor to generate fission source points leads to enough new fission points to fill the neutron bank for most generations.

When the "FISSION" portion of TRACK is entered, the library routine FLTRN is used to provide a random number that is saved. A pseudo fission weight is defined as the fission weight divided by the random number. If the result is less than the production factor, the history proceeds to "PATH" or is stored in the neutron bank, depending on whether or not it remains in the same supergroup. If the history remained in the "FISSION" section, and its fission weight is greater than the production factor, the pseudo fission weight is redefined to be the production factor divided by the random number. If the fission bank is not full, the fission energy group is determined using FLTRN, the fission point is stored in the bank, and the number of fission points is incremented. The library routine MOVE is used to load information pertaining to the fission point from COMMON /NUTRON/ into the fission bank and the pseudo fission weight is loaded directly into the fission bank. If the fission bank is full when a new fission source point is generated, the bank is searched for the smallest pseudo fission weight. This is compared with the pseudo fission weight of the new fission point and the point having the larger pseudo fission weight is stored in the bank. After the fission point has been banked, the fission weight of the history is decremented by the production factor. If the remaining fission weight is greater than zero, the history returns to the beginning of "FISSION" to continue processing.

"ARRAY" - This portion of TRACK is responsible for processing a history that crosses a unit or box type boundary. This occurs only when the history is moving in an outward direction. A check is made to determine if the history is exiting from a unit in an array or a unit used as a hole. If the exited unit was used as a hole, the coordinates are transformed to the coordinate system of the region containing the hole. The history then proceeds to the "PATH" portion of TRACK. However, if the unit exited by the history was part of an array, a check is made to see if tracks are to be printed. If so, TRKWRT is called to print pertinent information about the history. Each face of the unit is checked in sequence to force proper positioning for face, edge and corner crossings. As each face is processed, the position of the unit within the unit orientation array is updated as appropriate (this is done by incrementing the X, Y and/or Z indices of the unit position). A logical flag is set for each face to indicate if the history is exiting from the array. After all faces have been processed or bypassed, a check is made to determine if the history remains within the array. If it does, the new unit type is determined from the position indices of the location within the unit orientation array, and a transformation of coordinates is done to correct for crossing into the new unit. If the new unit consists of only one region, the history proceeds to "FINBOX" to determine if the history is entering a new array. The history then proceeds to "PATH" to continue processing. If the history exited the global array, a check is made to determine if a reflector is external to it.

If not, the history leaks from the system or proceeds through the "ALBEDO" portion of TRACK as appropriate. If surrounding geometry exists where the history exits an array, a transformation of coordinates provides the history's position relative to the surrounding geometry. The history then proceeds to the "PATH" portion of TRACK and tracking continues.

"ALBEDO" - If an albedo boundary condition is specified on any face of the problem, this portion of TRACK is utilized to provide the proper treatment. Each face is checked in sequence to determine if the history (1) leaks from the system, (2) undergoes specular or mirror image reflection, (3) undergoes periodic reflection, or (4) proceeds through the differential albedo treatment.

If the history undergoes specular or mirror image reflection, it is returned at the point it exited the face with its energy unchanged and the sign of the direction cosine normal to that face reversed. If a periodic reflection occurs, the history is moved to the opposing face with its energy and direction cosines unchanged. If the history enters a differential albedo reflector, subroutine ALBEDO is called to determine the new weight of the history and its returning angle and energy. If the history enters a new supergroup, TRKWRT may be called to print the history track information and MOVE is called to bank the history in the neutron bank. When a history remains in the same supergroup, it is returned at the point it exited and Russian roulette is played if the returning weight is low enough to warrant that action. If the weight is sufficiently high to avoid playing Russian roulette or if the history survives Russian roulette, the history proceeds to the beginning of "PATH" and tracking continues.

F11.3.13.4 *Provide the Next-Generation Source*

This section of the program is responsible for providing the source for the next generation from the fission source generated during the tracking procedure.

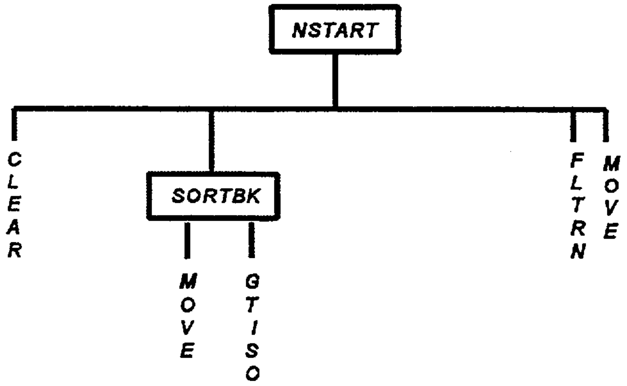


Figure F11.3.33 Flow chart for providing the next generation source

NSTART - This subroutine calls CLEAR to initialize the neutron bank and writes an error message if no fission points were generated. Subroutine SORTBK is called to move information from the fission bank containing the fission source generated by the last generation into the neutron bank to be used as the source for the next generation. NSTART then checks to be sure enough source points exist to start the next generation. If too few starting points exist, the library routines MOVE and FLTRN are used to fill the required number of starting positions from the existing fission points.

SORTBK - This subroutine sorts the fission bank so the fission points are loaded in the order of their probability of being picked in a random selection process. For each source history, the library routine MOVE is used to move the fission source generated by the last generation into the neutron bank, and the library routine GTISO is used to provide direction cosines from an isotropic distribution. Then the neutron number, weight and age are initialized.

F11.3.13.5 *End-of-Generation Processing*

This portion of the program is responsible for processing data at the end of each generation (Fig. F11.3.34).

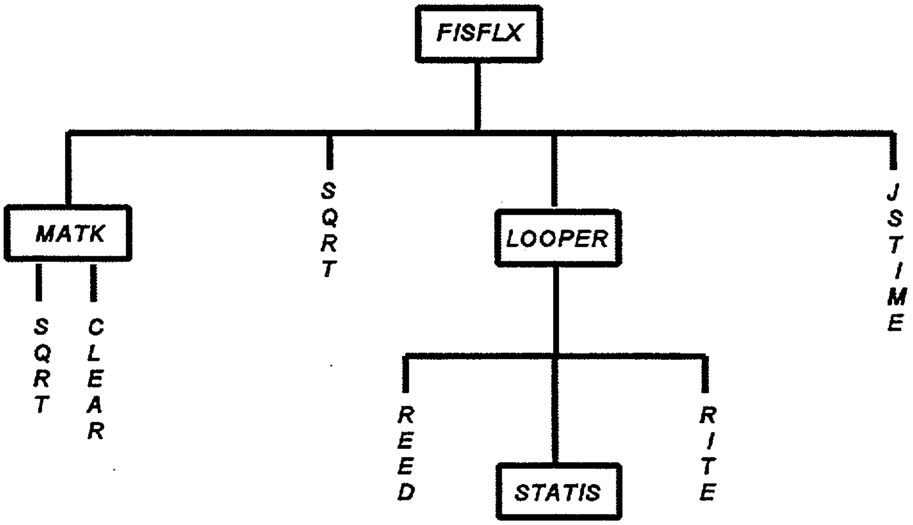


Figure F11.3.34 Flow chart for end of generation processing

FISFLX - This subroutine is called at the end of each generation to process data collected for that generation. The generation k-effective, the running average value of k-effective and its deviation, and the matrix k-effective and its deviation are processed and printed. MATK is called to process the matrix k-effective(s) and associated information. LOOPER is called and, in turn, calls STATIS to collect and process the contribution and statistics for the flux, fissions, and absorptions, as well as the contribution to the leakage.

MATK - This subroutine is called to calculate the matrix k-effective by solving for the principal eigenvalue and eigenvector of a matrix using an iterative technique. The library routine CLEAR is used to initialize arrays and SQRT is used in calculating the deviation of the eigenvalue. MATK may be called to calculate the matrix k-effective by array position, unit type, array number, and/or hole number.

LOOPER - This subroutine is called from FISFLX to load arrays in preparation for calling STATIS. A loop is made over the number of supergroups, within which pointers are calculated, REED is called to load the leakage, absorption, fission and flux arrays from the direct-access supergroup file, STATIS is called to process the data and RITE writes the processed data on the direct-access supergroup file. This procedure is repeated until all supergroups have been processed.

STATIS - This subroutine collects the sum of the contributions and the sum of the square of the contributions for the fluxes, fissions, absorptions and leakages to be used at the end of the problem in calculating their deviations.

F11.3.14 END-OF-PROBLEM PROCESSING

This portion of the program is responsible for processing data and printing all the results except the fluxes at the completion or termination of a problem (Fig. F11.3.35).

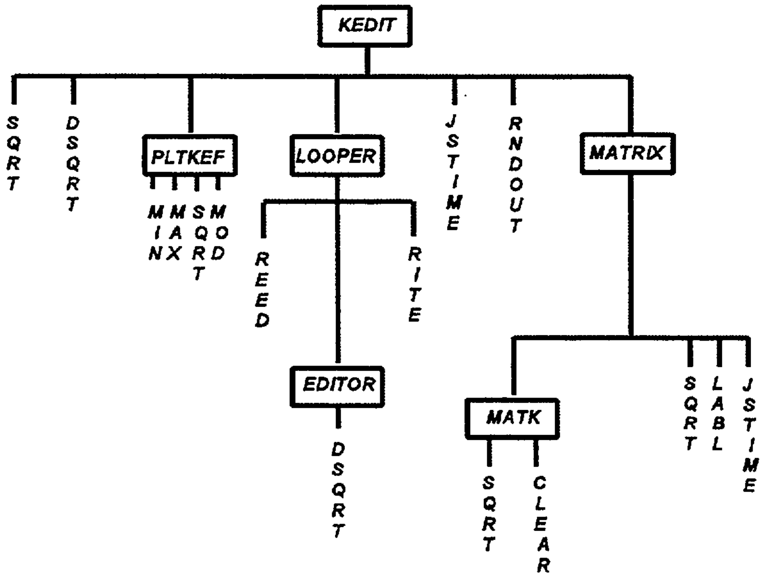


Figure F11.3.35 Flow chart for end-of-problem processing

KEDIT - This subroutine controls the processing and printing of results at the end of a problem. The life time and generation time are printed and, if the average number of neutrons per fission was calculated, it and the average fission group are printed. KEDIT then calculates and prints the average k-effective and its associated deviation for the 67, 95, and 99% confidence intervals and the number of histories involved. This is done repeatedly, skipping more generations each time. PLTKEF is called to print a plot of the average value of k-effective as a function of the number of generations. It also prints a plot of the average value of k-effective as a function of the number of generations skipped. LOOPER is called to prepare data for EDITOR which in turn calculates and prints the group-dependent fissions, absorptions and leakages and their deviations. If requested, the fissions and absorptions may also be printed by region. KEDIT then prints the total fissions, absorptions and leakages for the system, and their associated deviations. RNDOUT is called to provide the current random number to be printed. If matrix k-effective data were requested, MATRIX is called to calculate and print matrix information. KEDIT then processes and prints the fission densities.

PLTKEF - This subroutine is called from KEDIT to print a plot of the average value of k-effective versus generation and a plot of the average value of k-effective versus generations skipped. The library routines MIN, MAX, and SQRT are used to generate the k-effective axis of the plots. The MOD function is used in labeling the generation axis.

LOOPER - This subroutine is called from KEDIT to load arrays in preparation for calling EDITOR. A loop is made over the number of supergroups. Within the loop, pointers are calculated, REED is called to load the leakage, absorption, fission and flux arrays from the direct-access supergroup file and EDITOR is called to process the data. RITE then writes the processed data for the supergroup on the direct-access supergroup file.

EDITOR - This subroutine is called from LOOPER to calculate and print the energy-dependent fissions, absorptions and leakages and their deviations. The fissions and absorptions may be region-dependent as well as energy-dependent.

MATRIX - This subroutine is called from KEDIT to calculate and print various information related to the matrix k-effective. It is called if one or more matrix options were specified in the parameter data (see Sect. F11.4.3). If matrix information was collected, MATK is called to calculate cofactor k-effectives. MATRIX then prints them as they are calculated. MATRIX also prints the fission production matrix if that was specified in the parameter data (see Sect. F11.4.3). The library routine LABL is called to print the source vector. MATRIX then prints the average self-multiplication calculated on the basis of collected data. This procedure is repeated for each type of matrix information specified in the parameter data. JSTIME is called to determine the amount of time used in the problem, which is then printed in MATRIX.

MATK - This subroutine calculates the principal eigenvalue and eigenvector of a matrix using an iterative technique. It also calculates the deviation associated with the eigenvalue, using CLEAR to initialize arrays and SQRT in calculating the deviation. MATK may be called from MATRIX to calculate cofactor k-effectives.

F11.3.15 PRINT FLUXES

This portion of the program is responsible for printing the fluxes at the completion of a problem (Fig. F11.3.36).

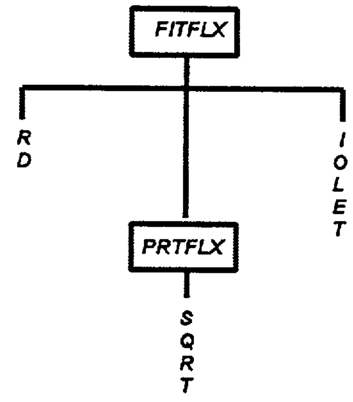


Figure F11.3.36 Flow chart for printing fluxes

FITFLX - This subroutine determines the maximum number of regions for which flux data will fit in memory and loads and prints them as they will fit. The library routine IOLEFT is called to determine the number of input/output requests remaining before the system cancels the job. Pointers are calculated and the library routine RD is used to load the fluxes for those regions from the direct-access supergroup file. PRTFLX is called to calculate the deviations and print the region- and energy-dependent fluxes and their associated deviations. If more fluxes remain to be printed, the process is repeated until all have been printed.

PRTFLX - This subroutine normalizes the fluxes, calculates their deviations and prints them, one supergroup at a time. The library routine SQRT is used when calculating the deviations.

F11.3.16 REFERENCES

1. S. K. Fraley, *Users Guide for ICE-II*, ORNL/CSD/TM-9/R1, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., July 1977. Also see Sect. F8.

2. "Appendix B, Generalized Gaussian Quadrature," *The MORSE Code - A Multigroup Neutron and Gamma-Ray Monte Carlo Transport Code*, ORNL-4585, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., 1970. Also see Sect. F9.

F11.4 KENO V.a DATA GUIDE

KENO V.a may be run "stand alone" or as a part of a SCALE criticality safety sequence. If KENO V.a is run "stand alone," cross-section data can be utilized from an AMPX[1] working format library or from an ICE (Sect. F8) mixed cross-section MORSE/KENO V.a format library, also called a Monte Carlo format cross-section library. If KENO V.a uses an AMPX working format library, a mixing table data block must be entered. If an ICE mixed cross-section MORSE/KENO V.a library is used, a mixing table data block is not entered and the mixtures specified in the KENO V.a geometry description must be consistent with the mixtures created in ICE. These are the entries in the 11$[MUD] array in the ICE input data.

If KENO V.a is run as part of a SCALE criticality safety sequence, the mixtures are defined in the CSAS data (Sect. C4.4) and a mixing table data block cannot be entered in KENO V.a. Furthermore, the mixture numbers used in the KENO V.a geometry description must correspond to those defined in the standard composition data of the CSAS input. A mixture number of 500 **must** be used in the KENO V.a geometry description in order to use a cell-weighted mixture. A cell-weighted mixture is available only in SCALE sequences that use XSDRN to perform a cell-weighting calculation.

F11.4.1 KENO V.a INPUT OUTLINE

The data input for KENO V.a is outlined below. Defaulted data for KENO V.a have been found to be adequate for many problems. These values should be carefully considered when entering data. The information in **BOLD TYPE** is entered as data.

Blocks of input data are entered in the form:

READ XXXX input data END XXXX

where **XXXX** is the keyword for the type of data being entered. The keywords that can be used are listed in Table F11.4.1. A minimum of four characters are required for a keyword. However, the keywords can be up to twelve characters long, the first four of which **must be input exactly as listed in the table**. Data input is activated by entering the words **READ XXXX** followed by one or more blanks. All input data pertinent to **XXXX** are then entered. Data for **XXXX** are terminated by entering **END XXXX** followed by two or more blanks.

Table F11.4.1 Types of input data

Type of data	First four characters
Parameters	PARA or PARM
Geometry	GEOM
Biasing	BIAS
Boundary conditions	BOUN or BNDS
Start	STAR or STRT
Array (unit orientation)	ARRA
Extra 1-D cross sections	X1DS
Cross-section Mixing Table[a]	MIXT or MIX
Plot[a]	PLOT or PLT or PICT

[a]MIX and PLT must include a trailing blank which is considered part of the keyword.

Two data records **must** be entered for every problem. The first is the problem title. The second is the **END DATA** to terminate the problem.

(1) **problem title**

Enter a problem title (limit 80 characters including blanks). A title **must be entered.** See Section F11.4.3.

(2) **READ PARA parameter data END PARA**

Enter parameter input as needed to describe a problem. Default values are assigned to all parameters. A problem **can** be run without entering any parameter data if the default values are acceptable.

Parameter data must begin with the words **READ PARA.** Parameter data may be entered in any order. If a parameter is entered more than once, the last value is used. The words **END PARA** terminate the parameter data. See Sect. F11.4.3.

(3)...(9) The following data may be entered in any order. Data not needed to describe the problem may be omitted.

(n_1) **READ GEOM all geometry region data END GEOM**

Geometry region data **must be entered** for every problem that is not a restart problem. Geometry data must begin with the words **READ GEOM.** The words **END GEOM** terminate the geometry region data. See Sect. F11.4.4.

(n_2) **READ ARRA array definition data END ARRA**

Enter array definition data as needed to describe the problem. Array definition data define the array size and position units (defined in the geometry data) in a three-dimensional (3-D) lattice that represents the physical problem being analyzed. Array data must begin with the words **READ ARRA.** The words **END ARRA** terminate the array data. See Sect. F11.4.5.

(n_3) **READ BIAS biasing information END BIAS**

Biasing information is used to define the weight that is given a neutron surviving Russian roulette. Enter biasing information as needed to describe the problem. Biasing data must begin with the words **READ BIAS.** The words **END BIAS** terminate the biasing data. See Sect. F11.4.7.

(n_4) **READ BOUN albedo boundary conditions END BOUN**

Enter albedo boundary conditions as needed to describe the problem. Albedo data must begin with the words **READ BOUN** and terminate with the words **END BOUN.** See Sect. F11.4.6.

(n_5) **READ STAR starting distribution information END STAR**

Enter starting information data for starting the initial source neutrons only if a uniform starting distribution is undesirable. Start data must begin with the words **READ STAR** and terminate with the words **END STAR**. See Sect. F11.4.8.

(n_6) **READ MIXT cross-section mixing table END MIXT**

Enter a mixing table to define all the mixtures to be used in the problem. The mixing table must begin with the words **READ MIXT** and end with the words **END MIXT**. Do not enter mixing table data if KENO V.a is run as a part of a SCALE criticality safety sequence. See Sect. F11.4.10.

(n_7) **READ X1DS extra 1-D cross-section IDs END X1DS**

Enter the IDs of any extra 1-D cross sections that are to be used in the problem. These must be available on the mixture cross-section library. Extra 1-D cross-section data must begin with the words **READ X1DS** and terminate with the words **END X1DS**. See Sect. F11.4.9.

(n_8) **READ PLOT plot data END PLOT**

Enter the data needed to provide a 2-D character or color plot of a slice through a specified portion of the 3-D geometrical representation of the problem. Plot data must begin with the words **READ PLOT** and terminate with the words **END PLOT**. See Sect. F11.4.11.

(n_9) **END DATA must be entered**

Terminate the data for the problem.

F11.4.2 PROCEDURE FOR DATA INPUT

This section is a brief list of the input data for KENO V.a. Additional information concerning KENO V.a data input may be found in Sect. F11.5. The first data record **must** be the title. The next block of data **must** be the parameters if they are to be entered. A problem can be run without entering the parameters. The remaining blocks of data can be entered in any order.

BOLD TYPE specifies keywords. A keyword is used to identify the data that follow it. When a keyword is used, it must be entered exactly as shown in the data guide. All keywords, except those ending with an equal sign, must be followed by at least one blank.

small italics correlate data with a program variable name. The actual values are entered in place of the program variable name and are terminated by a blank or a comma.

CAPITAL ITALICS identify general data items. General data items are general classes of data including

(1) geometry data such as *UNIT INITIALIZATION* and *UNIT NUMBER DEFINITION, GEOMETRY REGION DESCRIPTION, GEOMETRY WORD, MIXTURE NUMBER, BIAS ID,* and *REGION DIMENSIONS,*

(2) albedo data such as *FACE CODES* and *ALBEDO NAMES,*

(3) weighting data such as *BIAS ID NUMBERS*, etc.

F11.4.3 TITLE AND PARAMETER DATA

TITLE . . . A title must be entered.

title length is 80 characters, including blanks.

PARAMETER DATA . . . Enter only those whose values you wish to change. The commonly changed parameters are *TME, GEN,* and *NPG*. Seldom changed parameters are *NBK, NFB, XNB, XFB, WTH, WTL, TBA, BUG, TRK,* and *LNG.*

READ PARAM

Floating-point parameters

RND = *rndnum* input hexadecimal random number, a default value is provided.

TME = *tmax* execution time (in minutes) for the problem, default = 120 minutes.

TBA = *tbtch* time allotted for each generation (in minutes), default = 0.5 minutes. If *tbtch* is exceeded in any generation, the problem is assumed to be looping. Execution is terminated and final edits are performed. The problem can loop indefinitely on a computer if the system-dependent routine to interrupt the problem (PULL) is not functional.
TBA= is also used to set the amount of time available for generating the initial starting points.

WTA = *dwtav* the default average weight given a neutron that survives Russian roulette, *dwtav* default = 0.5.

WTH = *wthigh* the default value of *wthigh* is 3.0 and should be changed only if the user has a valid reason to do so. The weight at which splitting occurs is defined to be *wthigh x wtavg*, where *wtavg* is the weight given to a neutron that survives Russian roulette.

WTL = *wtlow* Russian roulette is played when the weight of a neutron is less than *wtlow x wtavg*. The *wtlow* default = 1.0/*wthigh.*

NOTE: The default values of *wthigh* and *wtlow* have been determined to minimize the deviation per unit running time for many problems.

Integer parameters

GEN = *nba* — number of generations to be run, default = 203

NPG = *npb* — number of neutrons per generation, default = 1000

NSK = *nskip* — number of generations (1 through *nskip*) to be omitted when collecting results, default = 3

RES = *nrstrt* — number of generations between writing restart data, default = 0. If **RES** is zero, restart data are not written. When restarting a problem, **RES** is defaulted to the value that was used when the restart data block was written. Thus, it must be entered as zero to terminate writing restart data for a restarted problem. (**WRS** is the logical unit number for writing restart data. See logical unit numbers in the parameter data.)

NBK = *nbank* — number of positions in the neutron bank, default = *npb* + 25

XNB = *nxnbk* — number of extra entries in the neutron bank, default = 0

NFB = *nfbnk* — number of positions in the fission bank, default = *npb*

XFB = *nxfbk* — number of extra entries in the fission bank, default = 0

X1D = *numx1d* — number of extra 1-D cross sections, default = 0

LNG = *lng* — number of words of storage to be requested by subroutine *ALOCAT*, default = 1000000. (This is reduced to fit in the space allotted to the job when the problem is run.)

BEG = *nbas* — beginning generation number, default = 1. If *BEG* is greater than 1, restart data must be available. *BEG* must be 1 greater than the number of generations retrieved from the restart file.

NB8 = *nb8* — number of blocks allocated for the first direct-access unit, default = 200

NL8 = *nl8* — length of blocks allocated for the first direct-access unit, default = 512

Alphanumeric parameter data ... enter *YES* or *NO*

FLX = *nflx* — key for collecting and printing fluxes, default = NO

FDN = *nfden* — key for collecting and printing fission densities, default = NO

ADJ = *nadj* key for running adjoint calculation, default = NO. Adjoint cross sections must be available to run an adjoint problem. If LIB= is specified, the cross sections will be adjointed by the code. If XSC= is specified, the cross sections must already be in adjoint order.

AMX = *amx* key for printing all mixture cross-section data. This is the same as activating *XAP, XS1, XS2, PKI*, and *P1D*. If any of these are entered in addition to *AMX*, that portion of *AMX* will be overridden, default = NO.

XAP = *prtap* key for printing discrete scattering angles and probabilities for the mixture cross sections, default = NO

XS1 = *prtp0* key for printing mixture 1-D cross sections, default = NO

XS2 = *prt1* key for printing mixture 2-D cross sections, default = NO

PKI = *prtchi* print input fission spectrum, default = NO

P1D = *prtex* print extra 1-D cross sections, default = NO

FAR = *lfa* key for generating region-dependent fissions and absorptions for each energy group, default = NO

GAS = *lgas* key for printing region-dependent fissions and absorptions by energy group, applicable only if FAR = YES. Default = FAR. GAS = YES prints region-dependent data by energy group. GAS = NO suppresses region-dependent data by energy group

MKP = *larpos* calculate and print matrix k-effective by unit location, default = NO.
Unit location may also be referred to as array position or position index.

CKP = *lckp* calculate and print cofactor k-effective by unit location, default = NO.
Unit location may also be referred to as array position or position index.

FMP = *pmapos* print fission production matrix by array position, default = NO

MKU = *lunit* calculate and print matrix k-effective by unit type, default = NO

CKU = *lcku* calculate and print cofactor k-effective by unit type, default = NO

FMU = *pmunit* print fission production matrix by unit type, default = NO

MKH = *lmhole* calculate and print matrix k-effective by hole number, default = NO

CKH = *lckh* calculate and print cofactor k-effective by hole number, default = NO

FMH = *pmhole* print fission production matrix by hole number, default = NO

HHL = *lhhgh* collect matrix information by hole number at the highest hole nesting level, default = NO

MKA = *lmarry* calculate and print matrix k-effective by array number, default = NO

CKA = *lcka* calculate and print cofactor k-effective by array number, default = NO

FMA = *pmarry* print fission production matrix by array number, default = NO

HAL = *langh* collect matrix information by array number at the highest array nesting level, default = NO

BUG = *ldbug* print debug information, default = NO
Enter *YES* for code debug purposes only.

TRK = *ltrk* print tracking information, default = NO
Enter *YES* for code debug purposes only.

PWT = *lpwt* print weight average array, default = NO

PGM = *lgeom* print unprocessed geometry as it is read, default = NO

SMU = *lmult* calculate the average self-multiplication of a unit, default = NO

NUB = *newbar* calculate the average number of neutrons per fission and the average energy group at which fission occurred, default = YES

PAX = *lcorsp* print the arrays defining the correspondence between the cross-section energy group structure and the albedo energy group structure, default = NO

RUN = *lrun* key for determining if the problem is to be executed when data checking is complete, default = YES. See note below.

PLT = *lplot* key for drawing specified plots of the problem geometry, default = YES. See note below.

NOTE: The parameters RUN and PLOT can also be entered in the PLOT data. See Sect. F11.4.11. It is recommended that these parameters be entered only in the parameter data in order to ensure that the data printed in the "Logical Parameters" table is what is actually performed. If RUN and/or PLT are entered in both the parameter data and plot data, the results vary, depending on whether the problem is run (1) stand-alone, (2) as a restarted problem, (3) as CSAS with parm=check, or (4) as CSAS without parm=check. These conditions are detailed below.

KENO V.a stand-alone and CSAS with PARM=CHECK — The values of RUN and/or PLT entered in the **KENO** parameter data are printed in the "Logical Parameters" table of the problem output. However, values for RUN and/or PLT entered in the **KENO plot data** will override the values entered in the parameter data.

Restarted KENO V.a — The values of RUN and/or PLT printed in the "Logical Parameters" table of the problem output are the final values from the "parent" problem unless those values are overridden by values entered in the **KENO parameter data** of the restarted problem. If the problem is restarted at generation 1, **KENO plot data** can be entered and the values for RUN and/or PLT will override the values printed in the "Logical Parameters" table.

CSAS without PARM=CHECK — The values of RUN and/or PLT entered in the KENO parameter data override values entered in the KENO plot data. The values printed in the "Logical Parameters" table control whether the problem is to be executed and whether a plot is performed.

Logical Unit Numbers

XSC = *xsecs* — logical unit number for a MORSE/KENO V.a format mixed cross-section library. When LIB≠0, default = 14. To read a mixed cross-section library from ICE or CSASI, XSC must be specified.

ALB = *albdo* — logical unit number for albedo data, default = 79

WTS = *wts* — logical unit number for weights, default = 80

LIB = *lib* — logical unit number for *AMPX* working format cross-section library, default = 0

SKT = *skrt* — logical unit number for scratch space, default = 16

RST = *rstrt* — logical unit number for reading restart data, default = 0
Enter a logical unit number to restart if *BEG* > 1

WRS = *wstrt* — logical unit number for writing restart data, default = 0
A non-zero value must be entered if *RES* > 0.

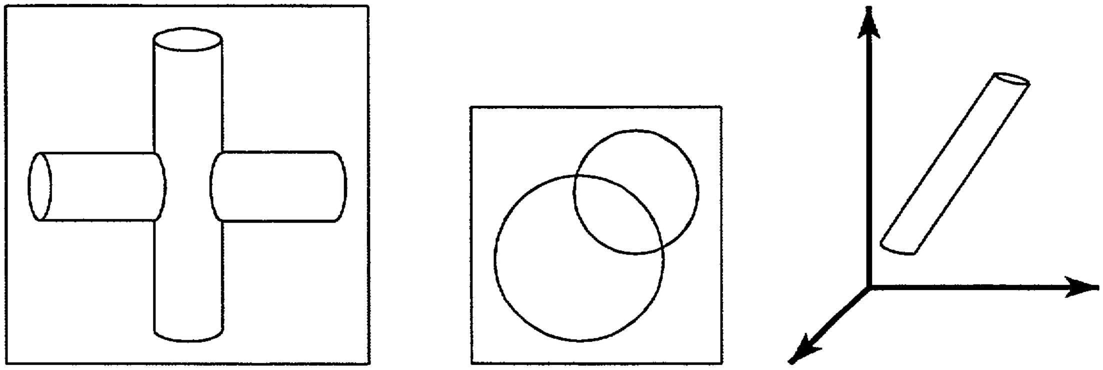
EXAMPLE: READ PARAM NPG=203 FLX=YES END PARAM

F11.4.4 GEOMETRY DATA

GEOMETRY REGION DATA . . . geometric arrangements in KENO V.a are achieved in a manner similar to using a child's building blocks. Each building block is called a UNIT or BOXTYPE. An array or lattice is constructed by stacking these units or box types. Once an array or lattice has been constructed, it can be placed in a unit by using an ARRAY or CORE specification.

Each UNIT in an array or lattice has its own coordinate system; however, all coordinate systems in all units must have the same orientation. All geometry data used in a problem are correlated to the absolute coordinate system by specifying a global unit or a global array. UNITS are constructed of combinations from several allowed shapes or geometric regions (i.e., cubes, rectangular parallelepipeds, cylinders, spheres, hemispheres, and hemicylinders). These regions can be placed anywhere within a UNIT as long as they are oriented along the coordinate system of the unit and do not intersect other regions. This means, for example, that a cylinder must have its axis parallel to one of the coordinate axes, while a rectangular parallelepiped must have its faces perpendicular to a coordinate axis. The most stringent KENO V.a geometry restriction is that none of the options allow geometry regions to intersect. Figure F11.4.1 shows some situations that aren't allowed.

Unless special options are invoked, each geometric region in a UNIT must completely enclose each interior region. However, regions can touch at points of tangency and share faces. See Fig. F11.4.2 for examples of allowable situations.



INTERSECTING REGIONS *INTERSECTING REGIONS* *ROTATED REGION*

Figure F11.4.1 Examples of geometry not allowed in KENO V.a

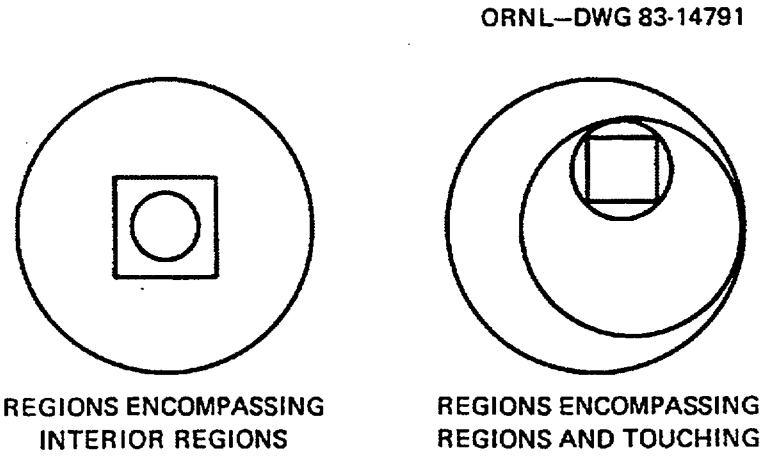

ORNL–DWG 83-14791
REGIONS ENCOMPASSING INTERIOR REGIONS
REGIONS ENCOMPASSING REGIONS AND TOUCHING

Figure F11.4.2 Examples of correct KENO V.a units

Special options are provided to circumvent the complete enclosure restriction. These options fall under the heading EXTENDED GEOMETRY DESCRIPTIONS and include ARRAY and HOLE descriptions. The HOLE option is the simplest of these and allows placing a unit anywhere within a region, as long as intersections do not occur (see Fig. F11.4.3). It is recommended that the outer boundary of a unit used as a hole should not be tangent to or share a boundary with either (1) another hole or (2) a region of the unit containing the hole because the code may find that the regions are intersecting due to precision and roundoff.

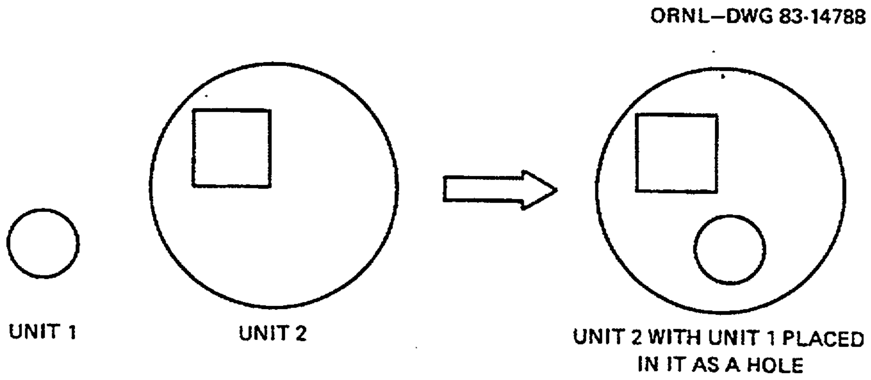


Figure F11.4.3 Example demonstrating HOLE capability in KENO V.a

An arbitrary number of HOLES can be placed in a region in combination with a series of surrounding regions.

Lattices or arrays are created by stacking UNITS that have a rectangular parallelepiped outer region. The adjacent faces of adjacent units stacked in this manner must match exactly. See Sect. F11.5.6.4 for additional clarification and Fig. F11.4.4 for a typical example.

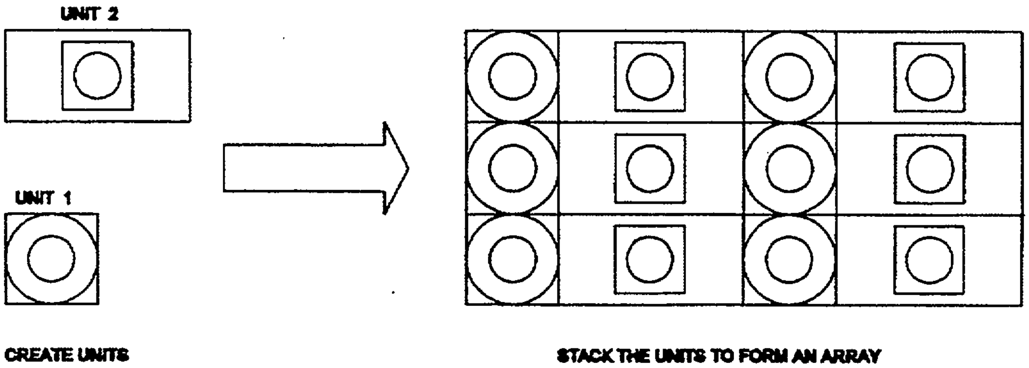


Figure F11.4.4 Example of array construction

The ARRAY option is provided to allow placing an array or lattice within a unit. Only one array can be placed directly in a UNIT. However, multiple arrays can be placed within a unit by using HOLES. When an array is placed in a unit via a HOLE, the unit that contains the array, rather than the array itself, is placed in the unit. Arrays of dissimilar arrays can be created by stacking units that contain arrays. See Fig. F11.4.5 for an example of an array composed of units containing holes and arrays.

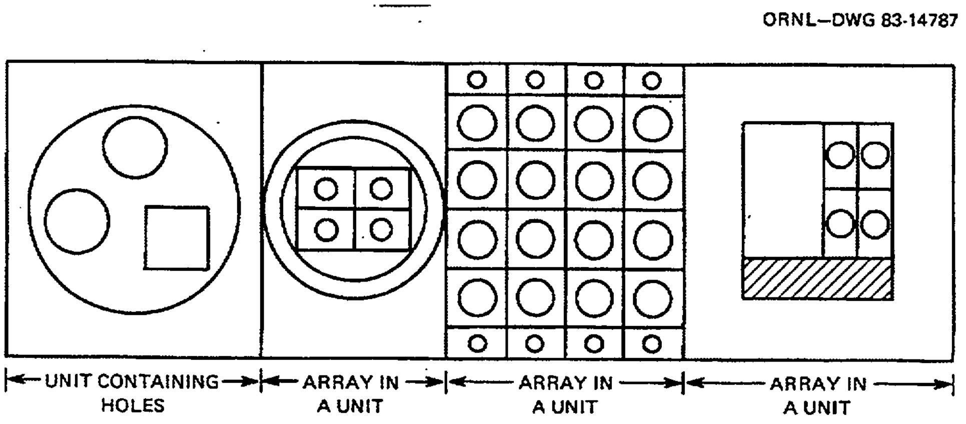


Figure F11.4.5 Example of an array composed of units containing arrays and holes

The method of entering GEOMETRY REGION DATA follows:

READ GEOM GEOMETRY REGION DATA END GEOM

GEOMETRY REGION DATA must be entered unless the problem is being restarted. A description of all units or box types the user wishes to define must be entered. See Sect. F11.5.6 for detailed examples.

The description of a unit includes all geometry data following a UNIT INITIALIZATION. A unit is terminated by encountering another UNIT INITIALIZATION or an END GEOM.

A GEOMETRY REGION DATA description consists of one or more of the following:

(a) UNIT INITIALIZATION
(b) SIMPLE GEOMETRY REGION DESCRIPTION
(c) EXTENDED GEOMETRY DESCRIPTION
(d) OPTIONAL GEOMETRY COMMENTS

(*a*) *UNIT INITIALIZATION* . . . This data sequence signals the beginning of a new geometric coordinate system and assigns the unit number to the geometry regions comprising the unit.

A UNIT INITIALIZATION is invoked when one or more of the following data items are encountered:

(a1) OPTIONAL GLOBAL SPECIFICATION
fgeom
GLOBAL

The word GLOBAL, if entered, must be followed by one or more blanks and a UNIT NUMBER DEFINITION and/or an ARRAY PLACEMENT DESCRIPTION. Enter the geometry word GLOBAL immediately prior to the UNIT or ARRAY that defines the overall geometric boundaries of the problem. The code defaults the global array to the array referenced by the last ARRAY PLACEMENT DESCRIPTION that is not immediately preceded by a UNIT NUMBER DEFINITION. Otherwise, it is the largest array number specified in the array data (Sect. F11.4.5). If the problem does not contain array data, Unit 1 is the default global unit.

(a2) UNIT NUMBER DEFINITION

KEYWORD	UNIT NUMBER
fgeom	*nbox*
UNIT	enter unit ID number (greater than zero)
or	
BOX TYPE	enter unit ID number (greater than zero)
or	
BOXTYPE	enter unit ID number (greater than zero)

The UNIT NUMBER definition assigns the specified unit number to the geometry data that define the unit.

(a3) ARRAY PLACEMENT DESCRIPTION
fgeom(*)
ARRAY
or
CORE
or
COREBDY
or
COREBNDS
or
COREBOUN

*Additional data are required as described in the EXTENDED GEOMETRY DESCRIPTION.

The array placement description is used to place a specified array in a UNIT. It positions the most negative point of the array relative to the origin of the UNIT.

(*b*) *SIMPLE GEOMETRY REGION DESCRIPTION* . . . This type of geometry data consists of simple shapes whose specification is independent of other geometry regions. Free-form input is used to enter the data. Options R, *, $, and P from Table F11.4.2 can be used. Each SIMPLE GEOMETRY REGION DESCRIPTION is entered in the form:

GEOMETRY WORD
fgeom

MIXTURE ID
mat

BIAS ID
imp

APPROPRIATE REGION DIMENSIONS (cm)
xx

OPTIONAL REGION DATA
origin data
and/or
chord data

The *GEOMETRY WORD fgeom* is followed by one or more blanks and must be one of the keywords below.

CUBE, CUBOID, SPHERE, CYLINDER, ZCYLINDER, XCYLINDER, YCYLINDER, HEMISPHERE, HEMISPHE+X, HEMISPHE-X, HEMISPHE+Y, HEMISPHE-Y, HEMISPHE+Z, HEMISPHE-Z, XHEMICYL+Y, XHEMICYL-Y, XHEMICYL+Z, XHEMICYL-Z, YHEMICYL+X, YHEMICYL-X, YHEMICYL+Z, YHEMICYL-Z, ZHEMICYL+X, ZHEMICYL-X, ZHEMICYL+Y, ZHEMICYL-Y

Note: *fgeom* may be no more than 12 characters long.

CUBE specifies a cube. It sets +X = +Y = +Z and -X = -Y = -Z. Note that the +X dimension need not equal the -X dimension of the cube (i.e., the origin need not be at the center of the cube).

CUBOID is a rectangular parallelepiped and may be described anywhere relative to the origin.

SPHERE specifies a sphere that is centered about the origin, unless otherwise specified by the optional region origin data.

CYLINDER specifies a cylinder that has its length described along the Z axis. Its centerline must lie on the Z axis, unless otherwise specified by the optional region origin data.

ZCYLINDER specifies a cylinder that has its length described along the Z axis. Its centerline must lie on the Z axis, unless otherwise specified by the optional region origin data.

XCYLINDER specifies a cylinder that has its length described along the X axis. Its centerline must lie on the X axis, unless otherwise specified by the optional region origin data.

YCYLINDER specifies a cylinder that has its length described along the Y axis. Its centerline must lie on the Y axis, unless otherwise specified by the optional region origin data.

HEMISPHERE is used to specify a spherical segment of one base whose spherical surface exists in the positive z direction. The base or flat portion of the spherical segment is centered about a point that may be specified in the optional region origin data. By default, the center of the spherical surface is the origin and the distance to the base from the center of the spherical surface is zero.

HEMISPHE*bc* is used to specify a spherical segment of one base whose spherical surface exists in the *bc* direction (b = + or -, c = x, y, or z). The base or flat portion of the spherical segment is located a distance ρ from the center of the spherical surface, and the center may be specified in the optional region origin data. **HEMISPHE+Z** is the same as the previously described **HEMISPHERE** and **HEMISPHE-Z** is the mirror image of **HEMISPHE+Z**, therefore existing only in the negative Z direction. By default the center of the spherical surface is the origin and the distance of the base from the center of the spherical surface is zero.

b*HEMICYL*cd is used to specify a cylindrical segment whose axis is in the *b* direction (b = x, y, or z) and whose cylindrical surface exists only in the *c* direction from a plane perpendicular to the *d* axis (c = + or -, d = x, y, or z). The position of the plane (cut surface) can be specified in the optional region chord data. This plane cuts the cylinder parallel to the axis at some distance, ρ, from the axis. By default, the axis passes through the origin and ρ is zero. (Examples: **ZHEMICYL+X, YHEMICYL-Z, XHEMICYL+Y**)

The *MIXTURE ID mat* specifies the mixture that is to occupy the volume defined by the region. The mixture ID number is followed by one or more blanks. A MIXTURE ID of zero indicates a void region (i.e., no material is present in the volume defined by the region).

The *BIAS ID imp* specifies the weights that are to be used in the volume defined by each geometry specification. Default weights are used for every *imp* not specified in the *BIASING INFORMATION* data. For clarification of how to use *imp* to specify nondefault weights, see Sect. F11.4.7. The *BIAS ID* number is followed by one or more blanks.

APPROPRIATE REGION DIMENSIONS. [*xx(1)* through *xx(6)*] are separated by one or more blanks and define the size of the region. Dimensions must be given in cm.

xx(1) Radius for a sphere, cylinder, hemisphere or hemicylinder,
+X dimension for a cube or cuboid.

xx(2) -X dimension for cube or cuboid,
+Z for cylinder or Z cylinder,
+X for X cylinder,
+Y for Y cylinder,
+length for hemicylinder,
omit *xx(2)* for a sphere or hemisphere.

xx(3) +Y dimension for cuboid,
-Z for cylinder or Z cylinder,
-X for X cylinder,
-Y for Y cylinder,
-length for hemicylinder,
omit *xx(3)* for a sphere, hemisphere, or cube.

xx(4) -Y dimension for cuboid,
omit for all other geometry types.

xx(5) +Z dimension for cuboid,
omit for all other geometry types.

xx(6) -Z dimension for cuboid,
omit for all other geometry types.

OPTIONAL REGION DATA specifies an origin and/or a chord that is applicable to the preceding g geometry region.

Enter origin data in the form:

fgeom enter the word **ORIG** or **ORIGIN**

xx(1) the X coordinate of the origin of a sphere or hemisphere,
the X coordinate of the centerline of a Z or Y cylinder or hemicylinder,
the Y coordinate of the centerline of an X cylinder or hemicylinder.

xx(2) the Y coordinate of the origin of a sphere or hemisphere,
the Y coordinate of a Z cylinder or hemicylinder
the Z coordinate of an X or Y cylinder of hemicylinder.

xx(3) the Z coordinate of the origin of a sphere or hemisphere; omit for all cylinders and hemicylinders.

Enter chord data in the form:

fgeom enter the word **CHORD**. Note: Chord data are applicable *only* for hemispherical and hemicylindrical shapes, *not* for SPHERE, XCYLINDER, YCYLINDER, CYLINDER, ZCYLINDER, CUBE, or CUBOID.

xx(7) The distance, ρ, from the "cut surface" to the center of a spherical surface or axis of a hemicylinder. See Figs. F11.4.6 and F11.4.7. For example, if ρ is 5 cm for both pictures, the chord data for Fig. F11.4.6 would be **CHORD -5.0** and the chord data for Fig. F11.4.7 would be **CHORD 5.0**. Entering a positive value with **CHORD** implies that more than half of the spherical or cylindrical body exists; entering a negative value with CHORD implies that less than half of the spherical or cylindrical body exists.

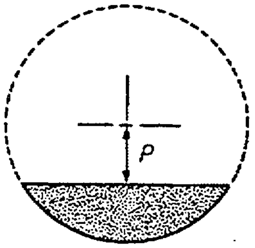


Figure F11.4.6 Partial hemisphere or hemicylinder, less than half exists
(less than half is defined by ρ < 0)

ORNL-DWG 83-9726

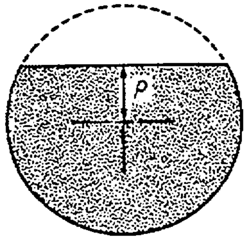


Figure F11.4.7 Partial hemisphere or hemicylinder, more than half exists
(more than half is defined by ρ > 0)

(*c*) *EXTENDED GEOMETRY DESCRIPTION* . . . This type of geometry data references geometric description data from other geometry or array descriptions. Free-form input is used to enter the data. Options R, *, $, and P from Table F11.4.2 can be used. Each EXTENDED GEOMETRY DESCRIPTION is entered in the form:

GEOMETRY WORD
fgeom

REFERENCE ID
mat

BIAS ID
imp

THICKNESS PER REGION (cm)
xx

ORIGIN COORDINATES
origin data

GENERATED REGIONS
nreg

The *GEOMETRY WORD fgeom* is followed by one or more blanks and must be one of the keywords below.

ARRAY, CORE, COREBDY, COREBNDS, COREBOUN, HOLE, REPLICATE, REFLECTOR

NOTE: *fgeom* may be no more than 12 characters long.

ARRAY, CORE, COREBDY, COREBNDS or **COREBOUN** are *ARRAY PLACEMENT* descriptions. They always start a new unit and generate a rectangular parallelepiped that fits the outer boundaries of the specified array. When an array placement description is the first geometry region in a unit, the specified array is positioned in the unit according to its origin coordinates. The origin coordinates specify the most negative point in the array with respect to the coordinate system of the surrounding unit.

HOLE is used to place a specified unit within the simple geometry region that precedes it. The position of the **HOLE** within the region is determined by the origin coordinates.

REPLICATE is used to generate additional geometry regions having the shape of the previous region. The geometry word **REFLECTOR** is a synonym for **REPLICATE**. The desired weighting functions can be applied to those regions by specifying biasing data as described in Sect. F11.4.7. The **REPLICATE** specification includes (1) the *MIXTURE ID, mat,* (2) the first *BIAS ID, imp,* for which these weighting data apply, (3) the thickness per region for each surface, *xx(1)...xx(6)*, as necessary to specify the desired thickness per region for each surface of the shape being generated, and (4) the number of regions to be generated, *nreg*. The total thickness generated for each surface is the thickness per region for that surface times *nreg*.

The replicate specification is frequently used to generate weighting regions external to an *ARRAY PLACEMENT* description. Thus an *ARRAY PLACEMENT* description followed by a **REPLICATE** description would generate regions of a cuboidal shape. A cylindrical reflector could be generated by following the *ARRAY PLACEMENT* description with a **CYLINDER** and then a **REPLICATE**. A **HOLE** cannot immediately follow a **REPLICATE**.

Extra regions using default weights can be generated by specifying the first importance region, *imp*, to be one that was not defined in the *BIASING INFORMATION*. This capability can be used to generate extra regions for collecting information such as fluxes, leakage, etc.

Multiple replicate descriptions can be used in any problem. This capability can be utilized to model different reflector materials of different thicknesses on different faces.

The number of appropriate region dimensions, *xx*, for specifying **REPLICATE** are determined by the preceding region. (For example, if the previous region was a sphere, one entry is required. If the previous region was a cylinder, the first entry is the thickness/region in the radial direction, the second entry is the thickness/region in the positive length direction, and the third entry is the thickness/region in the negative length direction, etc.) The replicate specification requirements for a cube are the same as for a cuboid.

REFERENCE ID mat specifies (1), (2), or (3) below:

(1) *numara*, the array referenced by the *ARRAY PLACEMENT* description (**ARRAY, CORE, COREBDY**, etc.).

(2) *lhole*, the unit that is to be placed within the preceding *SIMPLE GEOMETRY REGION* for a **HOLE** description.

(3) *mat*, the mixture that is to be used in the generated regions for a **REPLICATE** description.

BIAS ID imp (1) omit if *fgeom* is **ARRAY**. Enter a positive number for other *ARRAY PLACEMENT* descriptions.

(2) omit if *fgeom* is **HOLE**.

(3) For **REPLICATE**, *imp* is the *BIAS ID* number corresponding to the first generated region and is incremented by one for each generated region. If *imp* is negative it is not incremented, and all the generated regions use the absolute value of *imp* as their *BIAS ID*.

THICKNESS PER REGION xx (cm)

(1) omit for *ARRAY PLACEMENT* descriptions.

(2) omit for **HOLE**.

(3) for **REPLICATE**, [*xx(1)* through *xx(6)*] are separated by one or more blanks and define the size of the region. The thickness per region is always positive.

xx(1) First thickness per region for the generated geometry (i.e., in the radial direction for a spherical or cylindrical shape, or in the positive X direction for a cube, cuboid, or core).

xx(2) Second thickness per region for the generated geometry (i.e., in the direction of positive length for a cylinder or hemicylinder or in the negative X direction for a cube, cuboid, or core). Omit for sphere or hemisphere.

xx(3) Third thickness per region for the generated geometry (i.e., in the direction of negative length for a cylinder or hemicylinder or in the positive Y direction for a cube, cuboid, or core). Omit for sphere or hemisphere.

xx(4) Fourth thickness per region for the generated geometry (i.e., in the negative Y direction for a cube, cuboid, or core). Omit for all other geometry types.

xx(5) Fifth thickness per region for the generated geometry (i.e., in the positive Z direction for a cube, cuboid, or core). Omit for all other geometry types.

xx(6) Sixth thickness per region for the generated geometry (i.e., in the negative Z direction for a cube, cuboid, or core). Omit for all other geometry types.

ORIGIN COORDINATES xxorg [*xxorg(1)* through *xxorg(3)* are separated by one or more blanks] origin coordinates are given in cm.

xxorg(1) (1) enter the X coordinate of the most negative point of the array with respect to the coordinate system of the surrounding unit for *ARRAY PLACEMENT* descriptions.

(2) enter *xhole*, the offset in the X direction of the origin of the **HOLE UNIT** with respect to the origin of the region in which it is placed.

(3) omit for **REPLICATE.**

xxorg(2) (1) enter the Y coordinate of the most negative point of the array with respect to the coordinate system of the surrounding unit for *ARRAY PLACEMENT* descriptions.

(2) enter *yhole*, the offset in the Y direction of the origin of the **HOLE** unit with respect to the origin of the region in which it is placed.

(3) omit for **REPLICATE.**

xxorg(3) (1) enter the Z coordinate of the most negative point of the array with respect to the coordinate system of the surrounding unit for array placement descriptions.

(2) enter *zhole*, the offset in the Z direction of the origin of the **HOLE** unit with respect to the origin of the region in which it is placed.

(3) omit for **REPLICATE.**

NUMBER OF GENERATED REGIONS nreg (1) omit for array placement descriptions

(2) omit for **HOLE.**

(3) enter the number of regions to be generated for **REPLICATE.**

OPTIONAL GEOMETRY COMMENTS . . . these data allow the user to enter a comment for any unit. Data are entered in the form:

COM = *delim coment delim*

The keyword COM= signals that a comment is to be read. The first nonblank character following the keyword is the beginning delimiter, *delim*. The comment can be as long as 132 characters, including imbedded blanks. It must be terminated with the same delimiter that signaled the beginning of the comment. The optional geometry comment must follow the UNIT INITIALIZATION for a unit but can precede or follow any SIMPLE or EXTENDED GEOMETRY REGION DESCRIPTION within the unit. In the event that more than one comment is entered within a unit, the last one is used. Existing comments are printed at the beginning of each unit description in the computer printout.

EXAMPLES of geometry input are given below:

1. Initiate input data for unit number 6. UNIT 6

2. Specify a sphere of material 2, with a radius of 5.0 cm and its origin located at x = 1.0, y = 1.5, and z = 3.0. SPHERE 2 1 5.0 ORIGIN 1.0 1.5 3.0

3. Specify a hemicylinder of material 1, having a radius of 5.0 cm and a length extending from z = 2.0 cm to z = 7.0 cm. The hemicylinder has been truncated parallel to the z axis at x = -3 such that material 1 does not exist between x = -3 and x = -5. Position the origin of the truncated hemicylinder at x = 10 cm and y = 15 cm with respect to the origin of the unit.

   ZHEMICYL+X 1 1 5.0 7.0 2.0 CHORD 3.0 ORIGIN 10.0 15.0

4. Create five regions of material 4, each 3-cm thick, outside a cuboid region (a cuboid has six dimensions). The first generated region has a BIAS ID of 2.

   REPLICATE 4 2 6*3.0 5

5. Position the origin of array 6 at x = 2, y = 3, z = 4 relative to the origin of this unit.

   ARRAY 6 2 3 4 or CORE 6 1 2 3 4

6. Place unit 2 in this unit such that the origin of unit 2 is at x = 3, y = 3.5, z = 4.
   HOLE 2 3 3.5 4

F11.4.5 ARRAY DATA

ARRAY DEFINITION data . . . The array definition data block is used to define the size of an array and to position units (defined in the geometry data) in a 3-D lattice that represents the array being described. As many arrays as are necessary can be described in a problem, subject to computer storage limitations. A single unit problem is one for which an array definition data block is not entered.

An array definition data block consists of *ARRAY PARAMETERS* followed by a *UNIT ORIENTATION DESCRIPTION*. The sequence *ARRAY PARAMETERS, UNIT ORIENTATION DESCRIPTION* must be repeated for each array that is to be used in the problem. *ARRAY PARAMETERS* must be entered for all array problems and consist of parameter input defining (1) an array identification number for the lattice, (2) the number of units in each direction of the 3-D lattice, (3) the global array number, and (4) a comment to be printed at the beginning of the array in the printout. The global array is the array referenced by the global unit in a problem. If there is no global unit, the global array is the outermost array in the description of the problem. The array identification number (1), and the number of units in each direction, (2), and the associated *UNIT ORIENTATION DESCRIPTION* must be entered for each array that is used in the problem. The global array number, (3), should be entered only if a global unit does not exist. If it is entered more than once, the last value is used. The optional array comment is available for the user's convenience, but is not necessary for the problem. The *UNIT ORIENTATION DESCRIPTION* consists of a *KEYWORD, UNIT ORIENTATION DATA* and a *DELIMITER* which must be entered in order. The *KEYWORD* is used to indicate the method of entering the unit orientation data. The *UNIT ORIENTATION DATA* sequence is used to position units in 3-D lattice and the *DELIMITER* is used to terminate the unit orientation data for the array. The *UNIT ORIENTATION DESCRIPTION* does not need to be entered for a problem in which only one box type or unit is described *unless that unit number is larger than 1. It must be entered for all problems utilizing more than one unit or box type. The adjacent faces of units in contact with each other within an array must be the same size and shape.* Multiple arrays are defined by entering the sequence *ARRAY PARAMETERS, UNIT ORIENTATION DESCRIPTION* for each array that is to be described in the problem.

Enter **ARRAY DEFINITION DATA** in the form:

READ ARRAY *ARRAY PARAMETERS UNIT ORIENTATION DESCRIPTION* **END ARRAY**

A 1 × 1 × 1 array of Unit 1 can be defined by entering only the **READ ARRAY END ARRAY**. *ARRAY PARAMETERS* define the array number and the array size. They utilize the following keywords. Enter only those whose value you wish to change.

ARRAY PARAMETERS

ARA = *numa* array number for the array being input, default = 1.

GBL= *globl* array number for the global array (enter no more than once for a problem), default = largest value of *numa* for an unreflected system or the value of *numara* entered in the global unit.

NUX = *nbxmax* number of units in the X direction of the array, default = 1.

NUY = *nbymax* number of units in the Y direction of the array, default = 1.

NUZ = *nbzmax* number of units in the Z direction of the array, default = 1.

COM = *delim coment delim* allows entering a comment that will be printed with the unit orientation data. Maximum comment length is 132 characters.

The *UNIT ORIENTATION DESCRIPTION* is composed of a *KEYWORD, ORIENTATION DATA* and a *DELIMITER* as described below:

KEYWORD

type enter the word **LOOP** or **FILL**, followed by one or more blanks.

LOOP enters unit orientation data in a manner resembling *FORTRAN DO* loops. The first field contains the unit number, followed by three fields that are treated like *FORTRAN DO* loops. The arrangement of boxes may be considered as consisting of a 3-D matrix of unit numbers, with the unit position increasing in the positive X, Y, and Z directions, respectively. Each set of mixed box orientation data for the **LOOP** option consists of the following parameters, separated by one or more blanks.

ORIENTATION DATA for **LOOP**

ltype The unit or box type, *ltype* must be greater than zero.

ix1 The starting position in the X direction, *ix1* must be at least 1 and less than or equal to *nbxmax* (see *ARRAY PARAMETERS*, **NUX=**).

ix2 The ending position in the X direction, *ix2* must be at least 1 and less than or equal to *nbxmax.*

incx The number of units by which increments are made in the positive X direction. *incx* must be greater than zero and less than or equal to *nbxmax.*

iy1 The starting position in the Y direction. *iy1* must be at least 1 and less than or equal to *nbymax* (see *ARRAY PARAMETERS*, **NUY=**).

iy2 The ending position in the Y direction, *iy2* must be at least 1 and less than or equal to *nbymax.*

incy The number of units by which increments are made in the positive Y direction, *incy* must be greater than zero and less than or equal to *nbymax.*

iz1 The starting position in the Z direction, *iz1* must be at least 1 and less than or equal to *nbzmax* (see *ARRAY PARAMETERS*, **NUZ=**).

iz2 The ending position in the Z direction, *iz2* must be at least 1 and less than or equal to *nbzmax.*

incz The number of units by which increments are made in the positive Z direction. *incz* must be greater than zero and less than or equal to *nbzmax.*

The sequence *ltype* through *incz* is repeated until the entire array is described. If any portion of an array is defined in a conflicting manner, the last entry to define that portion will determine the array's configuration. To utilize this feature, fill the entire array with the most relevant unit type and superimpose the other unit types in their proper places. An example showing the use of the LOOP option is given below. This 5 × 4 × 3 array of units is a matrix of units that has 5 units stacked in the X direction, 4 units in the Y direction, and 3 units in the Z direction. X increases from left to right and Y increases from bottom to top. Each Z layer is shown separately.

Given:

1 2 1 2 1	2 1 2 1 2	1 1 1 1 1
1 1 1 1 1	2 2 2 2 2	1 3 3 3 1
1 1 1 1 1	2 2 2 2 2	1 3 3 3 1
1 2 1 2 1	2 1 2 1 2	1 1 1 1 1
Z Layer 1	Z Layer 2	Z Layer 3

The data for this array could be entered using the following entries.

(1) 1 1 5 1 1 4 1 1 3 1 This fills the entire array with 1's.

(2) 2 2 5 2 1 4 3 1 1 1 This loads the four 2's in the first Z layer.

(3) 2 1 5 1 2 3 1 2 2 1 This loads the second and third rows of 2's in the second Z layer.

(4) 2 1 5 2 1 4 3 2 2 1 This loads the desired 2's in the first and fourth rows of the second Z layer.

(5) 3 2 4 1 2 3 1 3 3 1 This loads the 3's in the third Z layer and completes the data input for thearray.

The second layer could have been defined by substituting the following data for entries (3) and (4)

(3) 2 1 5 1 1 4 1 2 2 1 This completely fills the second layer with 2's.

(4) 1 2 4 2 1 4 3 2 2 1 This loads the four 1's in the second layer.

When using the **LOOP** option, there is no single correct method of entering the data. If a unit is improperly positioned in the array or if some positions in the array are left undefined it is often easier to add additional data to correctly define it than to try to correct the existing data.

FILL enters data by stringing in unit numbers starting at X = 1, Y = 1, Z = 1, and varying X, then Y, and then Z to fill the array. *nbxmax* x *nbymax* x *nbzmax* entries are required. FIDO-like input options are also available for filling the array.

ORIENTATION DATA for **FILL**

The **FILL** option consists of entering a unit number for every position in the array by using the FIDO-like input options specified in Table F11.4.2. The orientation data for the **FILL** option may be terminated with a T.

DELIMITER

Enter the word **END** followed by a blank and the previously entered *KEYWORD* (i.e., enter **END FILL** or **END LOOP**). The delimiter need not be entered if only one set of *ARRAY DEFINITION DATA* is to be read.

To illustrate the use of the options available in Table F11.4.2, consider the following examples. The positions in an array are numbered sequentially from left to right, bottom to top. A 3 × 3 × 1 array has 9 positions and is numbered as shown below. X increases from left to right and Y increases from bottom to top.

7	8	9
4	5	6
1	2	3

EXAMPLE 1. Consider a 3 × 3 × 1 array filled with 8 Unit 1's and a Unit 2 as shown below.

1	1	1
1	2	1
1	1	1

The input data to describe this array could be entered as follows:

(1) 1 1 1 1 2 1 1 1 1 T This fills the array, one position at a time, starting at the lower left corner. The T terminates the data.

or

(2) F1 A5 2 T The F1 fills the entire array with 1's, the A5 locates the fifth position in the array, and the 2 loads a 2 in that position. The T terminates the data.

or

(3) F1 A1 4S 2 T The F1 fills the entire array with 1's, the A1 locates the first position in the array (lower left corner), the 4S skips over the next four positions in the array, and the 2 loads a 2 in the next (fifth) position. The T terminates the data.

or

Table F11.4.2 FIDO-like input for mixed box orientation fill option

Count field	Option field	Operand field	Function
		j	stores *j* at the current position in the array
i	R	*j*	stores *j* in the next *i* positions in the array
i	*	*j*	stores *j* in the next *i* positions in the array
i	$	*j*	stores *j* in the next *i* positions in the array
i	P	*j*	alternately stores *j* and -*j* in the next *i* positions of the array
	F	*j*	fills the remainder of the array with unit number *j*, starting with the current position in the array
	A	*j*	sets the current position in the array to *j*
i	S		increments the current position in the array by *i* (This allows skipping *i* positions. *i* may be positive or negative.)
i	Q	*j*	repeats the previous *j* entries *i* times. The default value of *i* is 1
i	N	*j*	repeats the previous *j* entries *i* times, inverting the sequence each time. The default value of *i* is 1
i	B	*j*	back *i* entries. From that position, repeat the previous *j* entries in reverse order. See Example 4. The default value of *i* is 1
i	I	*j k*	provides the end points, j and k, with i entries linearly interpolated between them (i.e., a total of i+2 points). At least one blank must separate j and k. When used for an integer array, the I option should only be used to generate integer steps [i.e., (k-j)/(i+1) should be a whole number]
i	L	*j k*	provides the end points, j and k, with i entries logarithmically interpolated between them (i.e., a total of i+2 points). At least one blank must separate j and k
	T		terminates the data reading for the array

Note: When entering data utilizing the options in this table, the *count field* and *option field must be adjacent with no imbedded* blanks. The operand field may be separated from the option field by one or more blanks.

(4) 4R1 2 4R1 T The first 4R1 loads 1's in the first four positions of the array, a 2 is loaded in the next (fifth) position of the array, and the last 4R1 loads 1's in the next four positions of the array. The T terminates the data.

or

(5) 4*1 2 4$1 T The 4*1 loads 1's in the first four positions of the array. A 2 is loaded in the next position of the array, and the 4$1 loads 1's in the next four positions of the array. The T terminates the data.

EXAMPLE 2. Consider a 4 × 3 × 1 array filled as shown below.

1	2	2	1
1	2	2	1
1	2	2	1

The input data to describe this array could be entered as follows.

(1) 1 2 2 1 1 2 2 1 1 2 2 1 T This fills the array, one position at a time, starting at the lower left corner. The T terminates the data.

(2) 1 2R2 2R1 Q4 2R2 1 T A 1 is loaded in the first position of the array. The 2R2 loads 2's in the next two positions of the array (positions two and three). The 2R1 loads 1's in the next two positions of the array (positions four and five). The Q4 loads the 2 2 1 1 from positions two through five in positions six through nine. The last 2R2 loads 2's in positions ten and eleven of the array. The last 1 loads a 1 in the next position (position twelve). The T terminates the data.

(3) 1 2 5N2 T The 1 is entered in the first position of the array and the 2 is entered in the second position. The 5N2 causes the previous two entries to be repeated five times reversing their order each time. The T terminates the data.

(4) 1 2R2 1 2Q4 T The first 1 is loaded in the first position of the array. The 2's are loaded in positions two and three of the array. Then a 1 is loaded in the fourth position of the array. This describes the entire bottom row of the array. The 2Q4 then repeats the previous four entries (those loaded in positions one through four of the array) two times. Thus the first row is repeated twice, filling the remainder of the array. The T terminates the data.

EXAMPLE 3. Consider a 6 × 3 × 1 array as shown below.

1	2	2	1	1	2
1	2	2	1	1	2
1	2	2	1	1	2

A simple input description for this array is

(1) 1 2 2N2 2Q6 T The first position of the array is filled with a 1. The second position of the array is filled with a 2. The 2N2 causes the previous two entries to be repeated two times, reversing their order each time. This completes loading the bottom row of the array. The 2Q6 repeats the previous six entries twice, completing the second and third rows of the array. The T terminates the data.

EXAMPLE 4. Consider a 7 × 3 × 1 array as shown below.

1	2	3	4	3	2	1
2	3	4	5	4	3	2
1	2	3	4	3	2	1

The input data to describe this array could be entered as follows:

(1) 1 2 3 4 1B3 2 3 4 5 1B3 1 2 3 4 1B3 T The 1 2 3 and 4 are loaded in the first four positions of the array. The 1B3 steps back over the 4 and repeats the 1 2 3 sequence in reverse order (i.e., 3 2 1). This yields the 1 2 3 4 3 2 1 in the first row of the array. The same procedure applies to the next two rows.

(2) 1 2 3 4 1B3 2 3 4 5 1B10 T The 1 2 3 4 1B3 yields the first row as explained above. The 2 3 4 5 enters the 2 3 4 5 at the left of the second row. The 1B10 enters the 4 3 2 at the right of the second row and the entire third row.

F11.4.6 ALBEDO DATA

ALBEDO DATA. . . Albedo boundary conditions are entered using a *FACE CODE* to define **where** albedo conditions are to be used, and an *ALBEDO NAME* to indicate **which** albedo condition is to be used on that face. The **default value for each face is vacuum.** The default values are overridden only on faces for which other albedo names are specified. Albedo boundary conditions are applied **only to the outermost region of a problem.** This geometry region must be a rectangular parallelepiped. It is possible to specify a different albedo name on each face of the outermost region. However, if a periodic boundary condition is to be used, it must be specified on opposing faces simultaneously.

Enter **ALBEDO DATA** in the form:

READ BOUNDS

FACE CODE
face

ALBEDO NAME
aname

END BOUNDS

The sequence *FACE CODE, ALBEDO NAME* is entered as many times as necessary to define the appropriate albedo boundary conditions. If multiple entries are made for a face, the *ALBEDO NAME* associated with the last *FACE CODE* specifying that face is used.

The *FACE CODES* are described in Table F11.4.3 and the *ALBEDO NAMES* are given in Table F11.4.4.

Example: Use a 24-in. concrete albedo boundary condition on the -Z face of the problem and use mirror image reflection on the +X and -X faces to represent an infinite linear array on a 2-ft-thick concrete pad.
READ BOUNDS -ZB=CON24 XFC=MIRROR END BOUNDS

F11.4.7 BIASING OR WEIGHTING DATA

BIASING INFORMATION . . . The average weight of a neutron that survives Russian roulette, *wtavg*, is defaulted to *dwtav* (**WTA=** in the parameter data, Sect. F11.4.3) for all *BIAS IDs* and can be overridden by entering biasing information.

The biasing information is used to relate a *BIAS ID* to the desired energy-dependent values of *wtavg*. This concept is similar to the way the *MIXTURE ID, mat*, is related to the macroscopic cross-section data.

The weighting functions used in KENO V.a are energy-dependent values of *wtavg* that are applicable over a given thickness interval of a material. For example, the weighting function for water[2] is composed of sets of energy-dependent values of *wtavg* for 11 intervals, each interval being 3-cm thick. The first set of *wtavg's* is for the 0- to 3-cm interval of water, the second set of *wtavg's* is for the 3- to 6-cm interval of water, etc. The eleventh set of *wtavg's* is for the 30- to 33-cm interval of water.

To input biasing information, a *BIAS ID* must be assigned to correspond to a set of *wtavg*. Biasing data can specify a *MATERIAL ID* from the existing KENO V.a weighting library or from the *AUXILIARY DATA* input. The materials available from the KENO V.a weighting library are listed in Table F11.4.5.

BIASING INFORMATION is entered in the following form:

READ BIAS KEYWORD CORRELATION DATA AUXILIARY DATA **END BIAS**

KEYWORD
enter **ID=**, **WT=**, or **WTS=**

ID= specifies that *CORRELATION DATA* will be entered next.

Table F11.4.3 Face codes for entering boundary (albedo) conditions

Face Code	Faces that are defined by the face codes
+XB=	Positive X face
&XB=	Positive X face
-XB=	Negative X face
+YB=	Positive Y face
&YB=	Positive Y face
-YB=	Negative Y face
+ZB=	Positive Z face
&ZB=	Positive Z face
-ZB=	Negative Z face
ALL=	All 6 faces
XFC=	Both positive and negative X faces
YFC=	Both positive and negative Y faces
ZFC=	Both positive and negative Z faces
+FC=	Positive X, Y, and Z faces
&FC=	Positive X, Y, and Z faces
-FC=	Negative X, Y, and Z faces
XYF=	Positive and negative X and Y faces
XZF=	Positive and negative X and Z faces
YZF=	Positive and negative Y and Z faces
+XY=	Positive X and Y faces
+YX=	Positive X and Y faces
&XY=	Positive X and Y faces
&YZ=	Positive X and Y faces
+XZ=	Positive X and Z faces
+ZX=	Positive X and Z faces
&XZ=	Positive X and Z faces
&ZX=	Positive X and Z faces
+YZ=	Positive Y and Z faces
+ZY=	Positive Y and Z faces
&YZ=	Positive Y and Z faces
&ZY=	Positive Y and Z faces
-XY=	Negative X and Y faces
-XZ=	Negative X and Z faces
-YZ=	Negative Y and Z faces
YXF=	Positive and negative X and Y faces
ZXF=	Positive and negative X and Z faces
ZYF=	Positive and negative Y and Z faces
-YX=	Negative X and Y faces
-ZX=	Negative X and Z faces
-ZY=	Negative Y and Z faces

Table F11.4.4 Albedo names available on the KENO V.a albedo library for use with the face codes

DP0H2O DPOH2O DP0 DPO	12-in. (30.48-cm) double P_0 water differential albedo with 4 incident angles
H2O WATER	12-in. (30.48-cm) water differential albedo with 4 incident angles
PARAFFIN PARA WAX	12-in. (30.48-cm) paraffin differential albedo with 4 incident angles
CARBON GRAPHITE C	78.74-in. (200.00-cm) carbon differential albedo with 4 incident angles
ETHYLENE POLY CH2	12-in. (30.48-cm) polyethylene differential albedo with 4 incident angles
CONC-4 CON4 CONC4	4-in. (10.16-cm) concrete differential albedo with 4 incident angles
CONC-8 CON8 CONC8	8-in. (20.32-cm) concrete differential albedo with 4 incident angles
CONC-12 CON12 CONC12	12-in. (30.48-cm) concrete differential albedo with 4 incident angles
CONC-16 CON16 CONC16	16-in. (40.64-cm) concrete differential albedo with 4 incident angles
CONC-24 CON24 CONC24	24-in. (60.96-cm) concrete differential albedo with 4 incident angles
VACUUM VOID VACU VAC	Vacuum condition
SPECULAR MIRROR MIRR SPEC SPE MIR REFL REFLECT	Mirror image reflection
PERIODIC PERI PER	Periodic boundary condition

WT= or **WTS=** specifies that *AUXILIARY DATA* will be entered next.

CORRELATION DATA are used to correlate a set of *wtavg* to a BIAS ID, *imp*, as specified in the geometry data. This causes the specified *wtavg* to be used as the weighting function in the volume defined by that geometry region.
CORRELATION DATA must be entered in the order shown.

id enter the identification (MATERIAL ID) for the material whose weighting function is to be used. A material ID can be chosen from the existing KENO V.a weighting library (Table F11.4.5) or from the *AUXILIARY DATA* as described later. If a material ID appears in both the KENO V.a weighting library and the *AUXILIARY DATA*, the *wtavg* from the auxiliary data will be used.

ibgn is the BIAS ID of the weighting function for the first interval of material *id*. The geometry card having *imp* = *ibgn* will use the group-dependent *wtavg's* from the first interval of material *id*.

iend is the BIAS ID of the group-dependent *wtavg's* from the (*iend* – *ibgn* + 1)*th* interval of material *id*.

GENERIC EXAMPLE: READ BIAS ID=mm ibgn iend END BIAS
where mm is a material ID from Table F11.4.5, *ibgn* is the bias ID associated with the 1st interval of material mm, and *iend* is the bias ID associated with the (*iend* – *ibgn* + 1) interval of material mm.

SPECIFIC EXAMPLE: Use CORRELATION DATA to utilize the water biasing factors in BIAS IDs 2 through 11.

```
READ BIAS   ID=500   2  11   END BIAS
```

AUXILIARY DATA are used to enter user-supplied biasing or weighting information. It can be used to supply biasing information for materials not found in the KENO V.a weighting library or to override the *wtavgs* from that library. When AUXILIARY DATA are entered, CORRELATION DATA must also be entered in order to use the data.
AUXILIARY DATA must be entered in the order shown.

wttitl enter an arbitrary title name (12 characters maximum), such as CONCRETE, WATER, SPECIALH2O, etc., to identify the material for which you are entering data. Embedded blanks are not allowed.

id enter an identification number (MATERIAL ID). The value is arbitrary. However, if the data are to be utilized in the problem, this ID must also be used at least once in the *CORRELATION DATA*.

nsets enter the number of sets of group structures for which *wtavg* will be read for this ID.

Table F11.4.5 IDs, group structure and incremental thicknesses for weighting data available on the KENO V.a weighting library

Material	Material ID	Group structure for which weights are available	Increment[a] thickness (cm)	Total number of increments available
Concrete	301	16	5	20
		27	5	20
		44	5	20
		218	5	20
		238	5	20
Paraffin	400	16	3	10
		27	3	10
		44	3	10
		218	3	10
		238	3	10
Water	500	16	3	10
		27	3	10
		44	3	10
		218	3	10
		238	3	10
Graphite	6100	16	20	10
		27	20	10
		44	20	10
		218	20	10
		238	20	10

[a] Group-dependent weight averages are supplied for each increment of the specified incremental thickness (i.e., for any given material) the first *ngp* (number of energy groups) weights apply to the first increment of the thickness specified in Table F11.4.4, the next *ngp* weights apply to the next increment of that thickness, etc. CAUTION--If bias IDs defined in the weighting information data are used in the geometry, the region thickness should be consistent with the incremental thickness of the weighting data in order to avoid overbiasing or underbiasing.

The sequence *thkinc, numinc, ngpwt, wtavg*, described below, is repeated *nsets* times.

thkinc enter the thickness of each increment for which *wtavg* will be read for this ID.

numinc enter the number of increments for which *wtavg* will be read for this ID.

ngpwt enter the number of energy groups for this set of *wtavg*.

wtavg enter *numinc x ngpwt* values of *wtavg*. For each value of *numinc, ngpwt* values of *wtavg* must be supplied.

GENERIC EXAMPLE of AUXILIARY DATA: READ BIAS WT=wttitl id nsets thkinc numinc ngpwt wtavg END BIAS

SPECIFIC EXAMPLE: Enter AUXILIARY DATA to specify biasing factors for SPECIALWATER to be used in BIAS IDs 6 and 7. The SPECIALWATER biasing factors have a value of 0.69 for BIAS ID 6 and 0.86 for BIAS ID 7 in each energy group. Sixteen-group cross sections are being used. Each weighting region is 3.048 cm thick. The MATERIAL ID is arbitrarily chosen to be 510. Note that CORRELATION DATA must be entered to allow the AUXILIARY DATA to be used for BIAS IDs 6 and 7.

READ BIAS WT=SPECIALWATER 510 1 3.048 2 16 16*0.69 16*0.86 ID=510 6 7 END BIAS

WARNING: The user should thoroughly understand weighted tracking before attempting to generate and use auxiliary data for biasing. Incorrect weighting can cause the code to produce incorrect results without obvious symptoms.

CAUTIONS:

1. Each set of auxiliary or correlation data must be completely described in conjunction with its keyword. Complete sets of these data can be interspersed in an arbitrary order but data within each set must be entered in the specified order.

2. Auxiliary data: If the same *id* is specified in more than one set of data, the last set having the group structure used in the problem is the set that will be utilized. When auxiliary data are entered, correlation data must also be entered in order to use the auxiliary data.

3. Correlation data: If biasing data define the same bias ID (*imp*, from the geometry data) more than once, the value that is entered last supersedes previous entries. *Be well aware that multiple definitions for the same bias ID can cause erroneous answers due to overbiasing.* Error messages K5-125 and K5-128 may be printed.

An example of multiple definitions for the same bias ID follows:

READ BIAS ID=400 2 4 ID=500 5 7 END BIAS .

The data for paraffin (ID=400) will be used for bias IDs 2, 3, and 4, and the data for water (ID=500) will be used for bias IDs 5, 6, and 7. The paraffin data for bias IDs 5, 6, and 7 have been overwritten by water data.

Multiple definitions for the same bias ID are not necessarily incorrect. However, the user should be cautious about doing it and ensure that the desired biasing or weighting functions are utilized in the desired geometry regions.

An example of how the *BIAS ID* relates to the energy-dependent values of *wtavg* is given below.

Assume that a paraffin reflector is to be used and it is desirable to use the weighting function from the KENO V.a weighting library to minimize the running time for the problem. Also assume that these weighting functions are to be used in the volumes defined in the geometry cards having *imp*= 6, 7, 8, and 9. *CORRELATION DATA* are then entered and *AUXILIARY DATA* will not be entered.

KEYWORD is **ID=**
id is 400, the ID for paraffin
ibgn is 6, the first *imp* that uses the weighting function
iend is 9, the last *imp* that uses the weighting function

The biasing data would be READ BIAS ID=400 6 9 END BIAS.

The results of these data are

(1) the group-dependent *wtavg* for the 0- to 3-cm interval of paraffin will be used in the volume defined by the geometry region having *imp*= 6.

(2) the group-dependent *wtavg* for the 3- to 6-cm interval of paraffin will be used in the volume defined by the geometry region having *imp*= 7.

(3) the group-dependent *wtavg* for the 6- to 9-cm interval of paraffin will be used in the volume defined by the geometry region having *imp*= 8.

(4) the group-dependent *wtavg* for the 9- to 12-cm interval of paraffin will be used in the volume defined by the geometry region having *imp*= 9.

F11.4.8 START DATA

START DATA . . . Special start options are available for controlling the initial neutron distribution. The default starting distribution for an array is flat over the overall array dimensions, in fissile material only. The default starting distribution for a single unit is flat over the system, in fissile material only. See Table F11.4.6 for the starting distributions available in KENO V.a.

READ START

The starting information that can be entered is given below. Enter only the data necessary to describe the desired starting distribution.

NST = *ntypst* start type, default = 0
Table F11.4.6 lists the available options under the heading, "Start type."

Table F11.4.6 Starting distributions available in KENO V.a

Start type	Required data	Optional data	Starting distribution
0	None	NST XSM XSP YSM YSP ZSM ZSP RFL PSP	Uniform throughout fissile material within the volume defined by (1) the outer region of a single unit, (2) the outer region of a reflected array having the reflector key set true, (3) the core boundary of the global array, or (4) a cuboid specified by XSM, XSP, YSM, YSP, ZSM, and ZSP.
1	NST	XSM XSP YSM YSP ZSM ZSP RFL PSP	The starting points are chosen according to a cosine distribution throughout the volume of a cuboid defined XSM, XSP, YSM, YSP, ZSM, and ZSP. Points that are not in fissile material are discarded.
2	NST NXS NYS NZS FCT	XSM XSP YSM YSP ZSM ZSP RFL PSP	An arbitrary fraction (FCT) of neutrons are started uniformly in the unit located at position NXS, NYS, NZS in the global array. The remainder of the neutrons are started in fissile material, from points chosen from a cosine distribution throughout the volume of a cuboid defined by XSM, XSP, YSM, YSP, ZSM, ZSP.
3	NST TFX TFY TFZ NXS NYS NZS	KFS PSP	All neutrons are started at position TFX, TFY, TFZ within the unit located at position NXS, NYS, NZS in the global array.
4	NST TFX TFY TFZ NBX	KFS PSP	All neutrons are started at position TFX, TFY, TFZ within units NBX in the global array.
5	NST NBX	PSP	Neutrons are started uniformly in fissile material in units NBX in the global array.
6	NST TFX TFY TFZ LNU[a]	NXS NYS NZS KFS PS6 PSP RDU	The starting distribution is arbitrarily input. LNU is the final neutron to be started at a point TFX, TFY, TFZ relative to the global coordinate system or at a point TFX, TFY, TFZ, relative to the unit located at the global array position NXS, NYS, NZS.

[a] When entering data for start 6, LNU must be the last entry for each set of data and the LNU in each successive set of data must be larger than the previous value of LNU. A set of data consists of required and optional data. The last LNU entered should be equal to the number per generation (parameter NPG= in the parameter input, Sect. F11.4.3).

TFX = *tfx* the x coordinate of the point at which neutrons are to be started. Default = 0.0.
Use for start types 3, 4, and 6.

TFY = *tfy* the y coordinate of the point at which neutrons are to be started. Default = 0.0.
Use for start types 3, 4, and 6.

TFZ = *tfz* the z coordinate of the point at which neutrons are to be started. Default = 0.0.
Use for start types 3, 4, and 6.

NXS = *nbxs* the x index of the unit's position in the global array. Default = 0.
Use for start types 2, 3, and 6.

NYS = *nbys* the y index of the unit's position in the global array. Default = 0.
Use for start types 2, 3, and 6.

NZS = *nbzs* the z index of the unit's position in the global array. Default = 0.
Use for start types 2, 3, and 6.

KFS = *kfis* the mixture whose fission spectrum is to be used for starting neutrons that are not in a fissionable medium. Defaulted to the fissionable mixture having the smallest mixture number.
Available for start types 3, 4, and 6.

LNU = *lfin* the final neutron to be started at a point. Default = 0. Each *lfin* should be greater than zero and less than or equal to NPG. Each successive *lfin* should be greater than the previous one.
Use for start type 6.

NBX = *nboxst* the unit or box type in which neutrons will be started. Default = 0
Use for start types 4 and 5.

FCT = *fract* the fraction of neutrons that will be started as a spike. Default = 0
Use for start type 2.

XSM = *xsm* the -X dimension of the cuboid in which the neutron will be started. For an array problem, XSM is defaulted to the minimum X coordinate of the global array. If the reflector key RFL is YES, and the outer reflector region is a cube or cuboid, XSM is defaulted to the minimum X coordinate of the outer reflector region. If RFL is YES and the outer region of the reflector is not a cube or cuboid, XSM must be entered in the start data and must fit inside the outer reflector region.
Available for start types 0, 1, and 2.

XSP = *xsp* the +X dimension of the cuboid in which the neutrons will be started. For an array problem, XSP is defaulted to the maximum X coordinate of the global array. If the reflector key RFL is YES, and the outer reflector region is a cube or cuboid, XSP is defaulted to the maximum X coordinate of the outer reflector region. If RFL is YES and the outer region of the reflector is not a cube or cuboid, XSP must be entered in the data and must fit inside the outer reflector region.
Available for start types 0, 1, and 2.

YSM = *ysm* the -Y dimension of the cuboid in which the neutron will be started. For an array problem, YSM is defaulted to the minimum Y coordinate of the global array. If the reflector key RFL is YES, YSM is defaulted to the minimum Y coordinate of the outer reflector region, provided that region is a cube or cuboid. If RFL is YES and the outer region of the reflector is not a cube or cuboid, YSM must be entered in the start data and must fit inside the outer reflector region.
Available for start types 0, 1, and 2.

YSP = *ysp* the +Y dimension of the cuboid in which the neutrons will be started. For an array problem, YSP is defaulted to the maximum Y coordinate of the global array. If the reflector key RFL is YES, YSP is defaulted to the maximum Y coordinate of the outer reflector region, provided that region is a cube or cuboid. If RFL is YES and the outer region of the reflector is not a cube or cuboid, YSP must be entered in the start data and must fit inside the outer reflector region.
Available for start types 0, 1, and 2.

ZSM = *zsm* the -Z dimension of the cuboid in which the neutrons will be started. For an array problem, XSM is defaulted to the minimum Z coordinate of the global array. If the reflector key RFL is YES, ZSM is defaulted to the minimum Z coordinate of the outer reflector region, provided that region is a cube or cuboid. If RFL is YES and the outer region of the reflector is not a cube or cuboid, ZSM must be entered in the start data and must fit inside the outer reflector region.
Available for start types 0, 1, and 2.

ZSP = *zsp* the +Z dimension of the cuboid in which the neutrons will be started. For an array problem, ZSP is defaulted to the maximum Z coordinate of the global array. If the reflector key RFL is YES, ZSP is defaulted to the maximum Z coordinate of the outer reflector region, provided that region is a cube or cuboid. If RFL is YES and the outer region of the reflector is not a cube or cuboid, ZSP must be entered in the start data and must fit inside the outer reflector region.
Available for start types 0, 1, and 2.

RFL = *rflkey* the reflector key. If the reflector key is YES, neutrons can be started in the reflector. If it is NO, all the neutrons will be started in the array. Enter YES or NO. Default = NO.
Available for start types 0, 1, and 2.

PS6 = *lprt6* the key for printing start type 6 input data. If the key is YES, start type 6 data are printed. If it is NO, start type 6 data are not printed. Enter YES or NO. Default = NO.
Available for start type 6.

PSP = *lpstp* the key for printing the neutron starting points using the tracking format. If the key is YES, print the neutron starting points. If it is NO, do not print the starting points. Enter YES or NO. Default = NO.
Available for all start types.

RDU = *n6unit* the logical unit number from which binary start data are to be read for start type 6. Default = 0.

END START

F11.4.9 EXTRA 1-D XSECS ID'S DATA

EXTRA 1-D CROSS-SECTION IDS . . . Extra 1-D cross-section IDs need not be entered. They are allowed as input in order to simplify future modifications to calculate reaction rates, etc.

READ X1DS *EXTRA 1-D CROSS SECTION IDS* **END X1DS**

EXTRA 1-D CROSS SECTION IDS

Enter a 1-D identification number for each extra 1-D cross section to be used. These cross sections must be available on the mixture cross-section library. X1D entries are expected to be read (see integer PARAMETER data).

F11.4.10 MIXING TABLE DATA

CROSS-SECTION MIXING TABLE . . . A cross-section mixing table must be entered if KENO V.a is being run "stand alone" and an ICE mixed cross-section MORSE/KENO V.a format library is not being used. If the parameter LIB=, Sect. F11.4.3, is entered, mixing table data must be entered. A cross-section mixing table is entered in the form:

READ MIXT *XSEC PARAMETERS MIXING TABLE* **END MIXT**

The *XSEC PARAMETERS* include the number of scattering angles and the cross-section message cutoff value.

The number of scattering angles specifies the number of discrete scattering angles to be used for the cross sections. It needs to be entered only once for a problem. If more than one value is entered, the last one is used for the problem. For assistance in determining the number of discrete scattering angles for the cross sections, see Sect. F11.5.4.3.

SCT = *nsct* where *nsct* is the number of discrete scattering angles, default = 1

The cross-section message cutoff value is the value of the P_o cross section for each energy transfer above which cross-section processing warning messages will be printed. The primary purpose of entering this cutoff value is to suppress printing these messages when they are generated during cross-section processing. For assistance in determining a value for EPS, see Sect. F11.5.4.4.

EPS = *pbxs* where *pbxs* is the value of the P_o cross section for each transfer, above which generated warning messages will be printed, default = 3×10^{-5}

The *MIXING TABLE* is used to specify each mixture and the nuclide IDs and number densities used in the mixtures. It consists of (a) a *MIXTURE ID* and a set of (b) *NUCLIDE IDs*, and (c) *NUMBER DENSITIES* (atoms/b-cm).

(a) *MIXTURE ID* **MIX** = *mix* where mix defines the mixture being described

(b) *NUCLIDE ID* *nucl* enter the nuclide ID number from the AMPX working format cross-section library

(c) *NUMBER DENSITY* *dens* enter the number density (atoms/b-cm) associated with nuclide ID number *nucl*

REPEAT the sequence (b) (c) until the mixture has been completely described.

REPEAT the sequence (a) (b) (c) until all the mixtures have been described.

NOTE: If a given nuclide ID is entered more than once in the same mixture, the number densities for that nuclide are summed.

If a mixture number is used as a nuclide ID, it is treated as a nuclide and the number density associated with it is used as a density modifier. (If the density is entered as 1, the mixture is mixed in at full density. If it is entered as 0.5, the mixture is mixed in at one-half its full density.) A Monte Carlo formatted cross-section library is generated on the unit defined by the parameter **XSC=**. If this data set is saved, subsequent cases can utilize these mixtures without remixing.

F11.4.11 PLOT DATA

PLOT DATA. . . Plots of slices specified through the geometry can be generated and displayed (1) as character plots using alphanumeric characters to represent mixture numbers, unit numbers or bias ID numbers or (2) as color plots which generate a GIF file using colors to represent mixture numbers, unit numbers or bias ID numbers. Color plots require an independent program to display the GIF file to a PC or workstation monitor or to convert the file to be displayed using a plotting device. The keyword **SCR=** is used to control the plot display method. **SCR=YES**, the default value, utilizes the color plot display method. **SCR=NO** utilizes the character plot display method. The value of **SCR** determines the plot display method for all the plots specified in a problem. If **SCR=** is entered more than once, the last entry determines the plot display method. In other words, all plots generated by a problem will be either character plots or color plots. The first LPI entered is used until it is changed. Therefore, the desired LPI should be entered before the end of the first set of plot data.

The *PLOT DATA* can include the data for any or all types of plots. A plot by mixture number is the default. The kind of plot is defined by the parameter **PIC=.** Character plots are printed after the volumes are printed and before the final preparations for tracking are completed. *PLOT DATA* is not required for a problem but can be used to verify the problem description. The actual plotting of the picture can be suppressed by entering **PLT= NO** in the parameter data or plot data. This allows plot data to be kept in the problem input for reference purposes without actually plotting the picture(s). Entering a value for **PLT** in the plot data will override any value entered in the parameter data. However, if a problem is restarted, the value of **PLT** from the parameter data is used. The upper-left and lower-right coordinates of the plot must be specified relative to the origin of the problem. See Sect. F11.5.9 for a discussion of plot origins and character plot data.

Enter **PLOT DATA** in the form:

READ PLOT *PLOT PARAMETERS* **END PLOT**

PLOT PARAMETERS are entered using keywords followed by the appropriate data. The plot title and the plot character string must be contained within delimiters. As many picture parameters as are necessary to describe the plot should be entered. Multiple sets of plot data can be entered. The parameter input for each plot is terminated by a labeled or unlabeled **END**. The labeled **END** cannot use the word **PLOT** as the first four characters of the label. For example, **END PLT1** is a valid label, but **END PLOT1** is not. If an unlabeled **END** is used, it cannot start in column 1.

TTL= *delim ptitl delim* Enter a one-character delimiter *delim* to signal the beginning of the title (132 characters maximum). The title is terminated when *delim* is encountered the second time.
Default = title of the KENO V.a case

PIC= *wrd* The plot type, *wrd*, is followed by one or more blanks and must be one of the keywords listed below. The plot type is initialized to MAT; the default is the value from the previous plot.

MAT
MIX
MIXT
MIXTURE
MEDI
MEDIA

These keywords will cause the plot to represent the mixture numbers used in the specified geometry slice.

BOX
BOXT
BOXTYPE
UNT
UNIT
UNITTYPE

These keywords will cause the plot to represent the units or box types used in the specified geometry slice. In the legend of the color plot, the material number actually refers to the units or box types.

IMP
BIAS
BIASID
WTS
WEIG
WEIGHTS
WGT
WGTS

These keywords will cause the plot to represent the bias ID numbers used in the specified geometry slice. In the legend of the color plot, the material number actually refers to the bias ID numbers.

Plot coordinates Enter values for the upper-left and lower-right coordinates of the plot as described below. **Data must be entered for all nonzero coordinates unless all six values from the previous plot are to be used.**

Upper left coordinates Enter the X, Y, and Z coordinates of the upper left-hand corner of the plot.

XUL=*fnam(1)* Enter the X coordinate of the upper left-hand corner of the plot.
Default = value from previous plot; initialized to zero if any other coordinates are entered.

YUL=*fnam(2)* Enter the Y coordinate of the upper left-hand corner of the plot.
Default = value from previous plot; initialized to zero if any other coordinates are entered.

ZUL=*fnam(3)* Enter the Z coordinate of the upper left-hand corner of the plot.
Default = value from previous plot; initialized to zero if any other coordinates are entered.

Lower right coordinates Enter the X, Y, and Z coordinates of the lower right-hand corner of the plot.

XLR=*fnam(4)* Enter the X coordinate of the lower right-hand corner of the plot.
Default = value from previous plot; initialized to zero if any other coordinates are entered.

YLR=*fnam(5)* Enter the Y coordinate of the lower right-hand corner of the plot.
Default = value from previous plot; initialized to zero if any other coordinates are entered.

ZLR=*fnam(6)* Enter the Z coordinate of the lower right-hand corner of the plot.
Default = value from previous plot; initialized to zero if any other coordinates are entered.

Direction cosines across the plot Enter direction numbers proportional to the direction cosines for the AX axis of the plot. The AX axis is from left to right across the plot. If any one of the AX direction cosines are entered, the other two are set to zero. The direction cosines are normalized by the code.

UAX=*fnam(7)* Enter the X component of the direction cosines for the AX axis of the plot.
Default = value from previous plot; initialized to zero if any other direction cosines are entered.

VAX=*fnam(8)* Enter the Y component of the direction cosines for the AX axis of the plot.
Default = value from previous plot; initialized to zero if any other direction cosines are entered.

WAX=*fnam(9)* Enter the Z component of the direction cosines for the AX axis of the plot.
Default = value from previous plot; initialized to zero if any other direction cosines are entered.

Direction cosines down the plot Enter direction numbers proportional to the direction cosines for the DN axis of the plot. The DN axis is from top to bottom down the plot. If any one of the DN direction cosines are entered, the other two are set to zero. The direction cosines are normalized by the code.

UDN=*fnam(10)* Enter the X component of the direction cosines for the DN axis of the plot.
Default = value from previous plot; initialized to zero if any other direction cosines are entered.

VDN=*fnam(11)* Enter the Y component of the direction cosines for the DN axis of the plot. Default = value from previous plot; initialized to zero if any other direction cosines are entered.

WDN=*fnam(12)* Enter the Z component of the direction cosines for the DN axis of the plot. Default = value from previous plot; initialized to zero if any other direction cosines are entered.

Scaling parameters Enter one or more scaling parameters to define the size of the plot. NOTE: If any of the scaling parameters are entered for a plot, the value of those that were not entered are recalculated. If none of the scaling parameters are specified for a plot, the values from the previous plot are used.

DLX=*fnam(13)* Horizontal spacing between points on the plot. Default = value from previous plot; initialized to zero if NAX or NDN is entered.

DLD=*fnam(14)* Vertical spacing between points on the plot. Default = value from previous plot; initialized to zero if NAX or NDN is entered.

NOTE: If either DLX or DLD is entered, the code will calculate the value of the other. If both are entered, the plot may be distorted.

NAX= *inam(15)* Number of intervals to be printed across the plot. Default = value from previous plot; initialized to zero if DLX or DLD is entered.

NDN= *inam(16)* Number of intervals to be printed down the plot. Default = value from previous plot; initialized to zero if DLX or DLD is entered.

Global scaling parameter

LPI= *lpi* is a scaling factor used to control the horizontal to vertical proportionality of a plot or plots. SCALE 4.3 and later versions allow **LPI** to be input as a floating point number. For an undistorted character plot, **LPI** should be specified as the number of characters down the page that occupy the same distance as ten characters across the page. For an undistorted color plot, **LPI** should be entered as ten times the ratio of the vertical pixel dimension to the horizontal pixel dimension. The default value of **LPI** is 8.0 for a character plot and 10.0 for a color plot. **LPI=10** will usually display an undistorted color plot.

The value entered for **LPI** applies to all plot data following it until a new value of **LPI** is specified.

NOTE: Plot data must include the specification of the upper left corner of the plot and the direction cosines across and down the plot.

Additional data required to generate a plot are:

1. the lower right corner of the plot, the global scaling parameter, **LPI**, and one of the scaling parameters (**DLX, DLD, NAX, NDN**).

2. the lower right corner of the plot, one of the scaling parameters related to the horizontal specifications of the plot (**DLX** or **NAX**), and one of the scaling parameters related to the vertical specification of the plot (**DLD** or **NDN**). **LPI**, even if specified will not be used.

3. **NAX** and **NDN** and any two of **LPI, DLX,** and **DLD**. If **LPI, DLX,** and **DLD** are all specified, **LPI** is not used.

The data required to generate a plot may be supplied from (1) defaulted values, (2) data from the previous plot, or (3) data that are specifically entered for the current plot.

Miscellaneous parameters — Enter miscellaneous parameters

RUN= *wrd* — Enter YES or NO. A value of YES means the problem will be executed if all the data were acceptable. A value of NO specifies the problem will be terminated after data checking is completed. The default value of **RUN** is YES.

PLT= *wrd* — Enter YES or NO. A value of YES specifies that a plot is to be made. If plot data are entered, **PLT** is defaulted to YES.

NOTE: The parameters RUN and PLT can also be entered in the PARAMETER data. See Sect. F11.4.3. It is recommended that these parameters be entered only in the parameter data in order to ensure that the data printed in the "Logical Parameters" table is what is actually performed. If RUN and/or PLT are entered in both the parameter data and plot data, the results vary, depending on whether the problem is run (1) stand-alone, (2) as a restarted problem, (3) as CSAS with parm=check, or (4) as CSAS without parm=check. These conditions are detailed below.

KENO V.a stand-alone and CSAS with PARM=CHECK — The values of RUN and/or PLT entered in the **KENO** parameter data are printed in the "Logical Parameters" table of the problem output. However, values for RUN and/or PLT entered in the **KENO plot data** will override the values entered in the parameter data.

Restarted KENO V.a — The values of RUN and/or PLT printed in the "Logical Parameters" table of the problem output are the final values from the "parent" problem unless those values are overridden by values entered in the **KENO parameter data** of the restarted problem. If the problem is restarted at generation 1, **KENO plot data** can be entered and the values for RUN and/or PLT will override the values printed in the "Logical Parameters" table.

CSAS without PARM=CHECK The values of RUN and/or PLT entered in the KENO parameter data override values entered in the KENO plot data. The values printed in the "Logical Parameters" table control whether the problem is to be executed and whether a plot is performed.

SCR= *wrd* The plot display method is specified by entering either YES or NO for *wrd*. The default value is YES. **SCR=YES** utilizes the color plot display method. **SCR=NO** utilizes the character plot display method. If SCR is entered more than once in a problem, the last value entered is the one that is used.

NCH= *delim CHAR delim* Enter only if plots are to be made utilizing the character plot display method (**SCR=NO**). Enter a one-character delimiter to signal the beginning of a character string, *CHAR*. The character string is terminated when *delim* is encountered the second time. *CHAR* is a character string with each entry representing a media (mixture) number, unit number, or bias ID. These are the characters that will be used in the plot. The first entry represents media, unit, or bias ID zero; the second entry represents the smallest media, unit, or bias ID used in the problem; the third, the next larger media, unit, or bias ID used in the problem; etc. For example, assume **PIC=MAT** is specified and 15 mixtures are defined in the mixing table and the geometry data uses only mixtures 3 and 7. By default, a blank will be printed for mixture zero, a 1 will be printed for mixture 3, and a 2 will be printed for mixture 7. If you wish to print a zero for a void (mixture 0), a 3 for mixture 3, and a 7 for mixture 7, enter NCH='037.' See Sect. F11.5.9 for examples.

The default values of *CHAR* are the following:

MEDIA	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
SYMBOL		1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
MEDIA	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
SYMBOL	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
MEDIA	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	
SYMBOL	W	X	Y	Z	#	,	$	-	+	)	|	&	>	:	;	
MEDIA	47	48	49	50	51	52	53	54	55	56	57	58				
SYMBOL	.	-	%	*	"	=	!	(	@	<	/	0				

CLR= *num(i) red(num(i)) green(num(i)) blue(num(i))* **END COLOR** enter only if plots are to be made utilizing the color plot display method (**SCR=YES**). After entering the keyword **CLR=**, 4 numbers are entered *i* times. The first number, *num*(i), represents a media (mixture) number, unit number, or bias ID. The next three numbers, whose values can range from 0 through 255, define the *red, green, and blue* components of the color that will represent this *num(i)* in the plot. The sequence *num(i) red(num(i)) green(num(i)) blue(num(i))* is repeated until the colors associated with all of the media (mixture) numbers, unit numbers, or bias IDs used in the problem have been defined. The smallest number that can be entered for *num(i)* is −1, representing undefined regions in the plot. A *num(i)* of 0 represents void regions, *num(i)* of 1 represents the smallest media, unit, or bias ID used in the problem, *num(i)* of 2 represents the next larger media, unit, or bias ID used in the problem, etc. The color plot definition data is terminated by entering the keywords **END COLOR**. A total of 256 default colors are provided in Table F11.4.7. Two of those colors represent undefined regions, *num*(-1), as black; and void regions, *num*(0), as gray. The remaining 254 colors represent the default values for mixtures, bias IDs, or unit numbers used in the problem. If ***num*** is entered as **-1**, the next three numbers define the color that will be used to represent undefined regions of the plot. The default color for undefined regions is black, represented as 0 0 0. If ***num*** is entered as **0**, the next three numbers define the color that will represent void regions in the plot. The default color for void is gray, represented as 200 200 200. For example, assume a color plot is to be made for a problem that utilizes void regions and mixture numbers 1, 3, and 5. By default, the undefined regions (Index -1) will be black; void regions (Index 0) will be gray; the first mixture, mixture 1 (Index 1), will be medium blue; the next larger mixture, mixture 3 (Index 2), will be turquoise2; and the last mixture, mixture 5 (Index 3), will be green2. If these values are acceptable, data need not be entered for CLR=. If the user decides to define void to be white (255 255 255), mixture 1 to be red (255 0 0), mixture 3 to be bright blue (0 0 255), and mixture 5 to be green (0 255 0), the following data could be entered:

```
CLR=0 255 255 255  1 255 0 0  2 0 0 255  3 0 255 0 END COLOR
```

In this example, the first number (0) defines the void and the next three numbers are the red, green, and blue components that combine as the color white. The fifth number (1) represents the smallest mixture number (mixture 1) and the next three numbers are the red, green, and blue components of red. The ninth number (2) represents the next larger mixture number (mixture 3), and the next three numbers are the red, green, and blue components of bright blue. The thirteenth number (3) represents the next larger mixture number (mixture 5), and the next three numbers are the red, green, and blue components of green. The END COLOR terminates the color definition data. Because color data were not entered for num(i) of -1, undefined regions will be represented by the color black, the default specification from Table F11.4.7. The red, green, and blue components of some bright colors are listed below.

Display Color	red	green	blue
black	0	0	0
white	255	255	255
"default void gray"	200	200	200
red	255	0	0
green	0	255	0
brightest blue	0	0	255
yellow	255	255	0
brightest cyan	0	255	255
magenta	255	0	255

The 256 default colors are listed in Table F11.4.7.

Table F11.4.7 Default color specifications for the color plot display method

-1	0	0	0	black gray0 grey0
0	200	200	200	*default
1	0	0	205	medium MediumBlue blue3
2	0	229	238	turquoise
3	0	238	0	green2
4	205	205	0	yellow3
5	238	0	0	red2
6	145	44	238	purple2
7	150	150	150	gray59 grey59
8	240	200	220	white gray100 grey100
9	0	191	255	deep DeepSkyBlue DeepSkyBlue1
10	224	255	255	light LightCyan LightCyan1
11	0	255	127	spring SpringGreen SpringGreen1
12	255	255	224	light LightYellow LightYellow1
13	255	0	0	red red1
14	255	0	255	magenta magenta1
15	67	110	238	RoyalBlue2
16	174	238	238	PaleTurquoise2
17	180	238	180	DarkSeaGreen2
18	238	220	130	LightGoldenrod2
19	238	99	99	IndianRed2
20	238	122	233	orchid2
21	25	25	112	midnight MidnightBlue
22	0	0	128	navy navy NavyBlue
23	100	149	237	cornflower CornflowerBlue
24	72	61	139	dark DarkSlateBlue
25	106	90	205	slate SlateBlue
26	123	104	238	medium MediumSlateBlue
27	30	144	255	dodger DodgerBlue DodgerBlue1
28	135	206	235	sky SkyBlue
29	135	206	250	light LightSkyBlue
30	70	130	180	steel SteelBlue
31	176	196	222	light LightSteelBlue
32	176	224	230	powder PowderBlue
33	0	206	209	dark DarkTurquoise
34	72	209	204	medium MediumTurquoise
35	95	158	160	cadet CadetBlue
36	102	205	170	medium MediumAquamarine aquamarine3
37	127	255	212	aquamarine aquamarine1
38	0	100	0	dark DarkGreen
39	85	107	47	dark DarkOliveGreen
40	143	188	143	dark DarkSeaGreen
41	60	179	113	medium MediumSeaGreen
42	32	178	170	light LightSeaGreen
43	152	251	152	pale PaleGreen
44	0	255	0	green green1
45	127	255	0	chartreuse chartreuse1
46	0	250	154	medium MediumSpringGreen
47	173	255	47	green GreenYellow
48	50	205	50	lime LimeGreen
49	154	205	50	yellow YellowGreen OliveDrab3
50	34	139	34	forest ForestGreen
51	107	142	35	olive OliveDrab
52	189	183	107	dark DarkKhaki
53	240	230	140	khaki
54	238	232	170	pale PaleGoldenrod
55	250	250	210	light LightGoldenrodYellow
56	255	255	224	light LightYellow LightYellow1
57	255	255	0	yellow yellow1
58	255	215	0	gold gold1
59	238	221	130	light LightGoldenrod
60	184	134	11	dark DarkGoldenrod
61	188	143	143	rosy RosyBrown
62	205	92	92	indian IndianRed
63	139	69	19	saddle SaddleBrown chocolate4
64	160	82	45	sienna
65	205	133	63	peru tan3
66	222	184	135	burlywood
67	245	245	220	beige
68	245	222	179	wheat
69	244	164	96	sandy SandyBrown
70	210	105	30	chocolate
71	178	34	34	firebrick
72	165	42	42	brown
73	233	150	122	dark DarkSalmon
74	250	128	114	salmon
75	255	160	122	light LightSalmon LightSalmon1
76	255	165	0	orange orange1
77	255	140	0	dark DarkOrange

Table F11.4.7 (continued)

78	255	127	80	coral
9	240	128	128	light LightCoral
80	255	99	71	tomato tomato1
81	255	69	0	orange OrangeRed OrangeRed1
82	255	0	0	red red1
83	255	105	180	hot HotPink
84	255	20	147	deep DeepPink DeepPink1
85	255	192	203	pink
86	255	182	193	light LightPink
87	219	112	147	pale PaleVioletRed
88	176	48	96	maroon
89	199	21	133	medium MediumVioletRed
90	208	32	144	violet VioletRed
91	238	130	238	violet
92	221	160	221	plum
93	218	112	214	orchid
94	153	50	204	dark DarkOrchid
95	148	0	211	dark DarkViolet
96	186	85	211	medium MediumOrchid
97	138	43	226	blue BlueViolet
98	160	32	240	purple
99	147	112	219	medium MediumPurple
100	216	191	216	thistle
101	238	233	233	snow2
102	238	229	222	seashell2
103	238	223	204	AntiqueWhite2
104	238	213	183	bisque2
105	238	203	173	PeachPuff2
106	238	207	161	NavajoWhite2
107	238	233	191	LemonChiffon2
108	238	232	205	cornsilk2
109	238	238	224	ivory2
110	224	238	224	honeydew2
111	238	224	229	LavenderBlush2
112	238	213	210	MistyRose2
113	224	238	238	azure2
114	122	103	238	SlateBlue2
115	67	110	238	RoyalBlue2
116	0	0	238	blue2
117	28	134	238	DodgerBlue2
118	92	172	238	SteelBlue2
119	0	178	238	DeepSkyBlue2
120	126	192	238	SkyBlue2
121	164	211	238	LightSkyBlue2
122	185	211	238	SlateGray2
123	188	210	238	LightSteelBlue2
124	178	223	238	LightBlue2
125	209	238	238	LightCyan2
126	174	238	238	PaleTurquoise2
127	142	229	238	CadetBlue2
128	0	238	238	cyan2
129	141	238	238	DarkSlateGray2
130	118	238	198	aquamarine2
131	180	238	180	DarkSeaGreen2
132	78	238	148	SeaGreen2
133	144	238	144	light lightGreen PaleGreen2
134	0	238	118	SpringGreen2
135	0	238	0	green2
136	118	238	0	chartreuse2
137	179	238	58	OliveDrab2
138	188	238	104	DarkOliveGreen2
139	238	230	133	khaki2
140	238	220	130	LightGoldenrod2
141	238	238	209	LightYellow2
142	238	238	0	yellow2
143	238	201	0	gold2
144	238	180	34	goldenrod2
145	238	173	14	DarkGoldenrod2
146	238	180	180	RosyBrown2
147	238	99	99	IndianRed2
148	238	121	66	sienna2
149	238	197	145	burlywood2
150	238	216	174	wheat2
151	238	154	73	tan2
152	238	118	33	chocolate2
153	238	44	44	firebrick2
154	238	59	59	brown2
155	238	130	98	salmon2
156	238	149	114	LightSalmon2
157	238	154	0	orange2
158	238	118	0	DarkOrange2
159	238	106	80	coral2
160	238	92	66	tomato2
161	238	64	0	OrangeRed2
162	238	0	0	red2
163	238	18	137	DeepPink2
164	238	106	167	HotPink2
165	238	169	184	pink2
166	238	162	173	LightPink2
167	238	121	159	PaleVioletRed2
168	238	48	167	maroon2
169	238	58	140	VioletRed2
170	238	0	238	magenta2
171	238	122	233	orchid2
172	238	174	238	plum2
173	209	95	238	MediumOrchid2
174	178	58	238	DarkOrchid2
175	145	44	238	purple2
176	159	121	238	MediumPurple2
177	238	210	238	thistle2
178	255	250	250	snow snow1
179	139	137	137	snow4
180	255	245	238	seashell seashell1
181	255	228	196	bisque bisque1
182	255	218	185	peach PeachPuff PeachPuff1
183	255	250	205	lemon LemonChiffon LemonChiffon1
184	255	248	220	cornsilk cornsilk1
185	255	255	240	ivory ivory1
186	240	255	240	honeydew honeydew1
187	255	240	245	lavender LavenderBlush LavenderBlush1
188	255	228	225	misty MistyRose MistyRose1
189	240	255	255	azure azure1
190	131	111	255	SlateBlue1
191	72	118	255	RoyalBlue1

Table F11.4.7 (continued)

192	30	144	255	dodger	225	255	127	36	chocolate1
				DodgerBlue	226	255	48	48	firebrick1
				DodgerBlue1	227	255	64	64	brown1
193	99	184	255	SteelBlue1	228	255	140	105	salmon1
194	0	191	255	deep	229	255	160	122	lightsalmon
				DeepSkyBlue					LightSalmon
				DeepSkyBlue1					LightSalmon1
195	135	206	255	SkyBlue1	230	255	165	0	orange
196	176	226	255	LightSkyBlue1					orange1
197	198	226	255	SlateGray1	231	255	127	0	DarkOrange1
198	202	225	255	LightSteelBlue1	232	255	114	86	coral1
199	191	239	255	LightBlue1	233	255	99	71	tomato
200	224	255	255	light					tomato1
				LightCyan	234	255	69	0	orangered
				LightCyan1					OrangeRed
201	187	255	255	PaleTurquoise1					OrangeRed1
202	152	245	255	CadetBlue1	235	255	20	147	deep
203	0	245	255	turquoise1					DeepPink
204	151	255	255	DarkSlateGray1					DeepPink1
205	127	255	212	aquamarine	236	255	110	180	HotPink1
				aquamarine1	237	255	181	197	pink1
206	193	255	193	DarkSeaGreen1	238	255	174	185	LightPink1
207	84	255	159	SeaGreen1	239	255	130	171	PaleVioletRed1
208	154	255	154	PaleGreen1	240	255	52	179	maroon1
209	0	255	127	spring	241	255	62	150	VioletRed1
				SpringGreen	242	255	131	250	orchid1
				SpringGreen1	243	255	187	255	plum1
210	127	255	0	chartreuse	244	224	102	255	MediumOrchid1
				chartreuse1	245	191	62	255	DarkOrchid1
211	192	255	62	OliveDrab1	246	155	48	255	purple1
212	202	255	112	DarkOliveGreen1	247	171	130	255	MediumPurple1
213	255	246	143	khaki1	248	255	225	255	thistle1
214	255	236	139	LightGoldenrod1	249	139	0	139	dark
215	255	255	224	light					darkMagenta
				LightYellow					magenta4
				LightYellow1	250	139	0	0	dark
216	255	215	0	gold					darkRed
				gold1					red4
217	255	193	37	goldenrod1	251	0	139	0	green4
218	255	185	15	DarkGoldenrod1	252	0	0	139	dark
219	255	193	193	RosyBrown1					darkBlue
220	255	106	106	IndianRed1					blue4
221	255	130	71	sienna1	253	0	139	139	dark
222	255	211	155	burlywood1					darkCyan
223	255	231	186	wheat1					cyan4
224	255	165	79	tan1	254	139	139	0	yellow4

F11.4.12 REFERENCES

1. N. M. Greene et al., *AMPX: A Modular Code System for Generating Coupled Multigroup Neutron-Gamma Libraries from ENDF/B,* ORNL/TM-3706, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., March 1976. Also see Sect. F2.3.8 of the SCALE manual.

2. J. R. Knight and L. M. Petrie, *16 and 123 Group Weighting Functions for KENO V.a,* ORNL/TM-4660, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., 1975.

Tables F11.4.8 through F11.4.15 summarize KENO V.a input data.

Table F11.4.8 Summary of parameter data

TITLE: The title must be entered first (80 characters, including blanks) See Sect. F11.4.3

PARAMETERS: Format: READ PARAM enter parameter data here END PARAM
If parameters are entered, they must follow the title. See Sects. F11.4.3, F11.5.2, and F11.5.3.

KEY	STD.	DEFINITION	KEY	STD.	DEFINITION	KEY	STD.	DEFINITION	KEY	STD.	DEFINITION
RND=	given	random number	FLX=	NO	fluxes	MKH=	NO	matrix by hole	XSC=	14	mixed xsecs
TME=	120 min	execution time (min)	FDN=	NO	fission densities	CKH=	NO	cofactor k by hole	ALB=	79	albedo
TBA=	0.5 min	batch time (min)	ADJ=	NO	adjoint calculation	FMH=	NO	fiss. prod. by hole	WTS=	80	weights
WTA=	0.5	average weight	AMX=	NO	all mixture xsecs	HHL=	NO	MKH at highest level	LIB=	0	working xsecs
WTH=	3.0	wt. for splitting	XAP=	NO	xsec angles & probs.	MKA=	NO	matrix by array	SKT=	16	scratch
WTL=	1/WTH	Russian Roulette wt.	XS1=	NO	1-D xsecs	CKA=	NO	cofactor k by array	RST=	0	read restart
GEN=	203	no. of generations	XS2=	NO	2-D xsecs	FMA=	NO	fiss. prod. by array	WRS=	0	write restart
NPG=	1000	no. per generation	PKI=	NO	fission spectrum	HAL=	NO	MKA at highest level			
NSK=	3	generations skipped	P1D=	NO	extra 1-D xsecs	BUG=	NO	debug print			
RES=	0	gens. between restart	FAR=	NO	fiss. & abs.	TRK=	NO	print neutron tracks			
NBK=	NPG+25	neutron bank	GAS=	FAR	FAR by energy group	PWT=	NO	print avg. weight			
XNB=	0	positions	MKP=	NO	matrix by location	PGM=	NO	unprocessed geometry			
NFB=	NPG	extra bank entries	CKP=	NO	cofactor k by loc.	SMU=	NO	self-multiplication			
XFB=	0	fission bank positions	FMP=	NO	fiss. prod. by loc.	NUB=	YES	neutrons per fission			
X1D=	0	extra bank entries	MKU=	NO	matrix by unit	PAX=	NO	albedo-xsec array			
LNG=	1000000	no. of extra 1-D's	CKU=	NO	cofactor k by unit	RUN=	YES	execute problem			
BEG=	1	words of storage	FMU=	NO	fiss. prod. by unit	PLT=	NO	printer plots			
NB8=	200	restart at this gen.									
NL8=	512	blocks for d.a. unit									
		length of d.a. block									

Table F11.4.9 Summary of array data

ARRAY Format: READ ARRAY array parameters data type orientation data END ARRAY See Sects. F11.5.5, F11.5.6, and F11.5.7.

Repeat the sequence ARRAY PARAMETERS DATA TYPE ORIENTATION DATA for each array used in the problem.

ARRAY PARAMETERS

KEYWORD	DEFAULT	DEFINITION
ARA=	1	no. defining the array
NUX=	1	no. of units in X direction
NUY=	1	no. of units in Y direction
NUZ=	1	no. of units in Z direction
GBL=	maxara	global, universe, or overall array number**
COM=	none	delim comment delim optional comment is a maximum of 132 characters

**Can be defaulted by the code. If specified, it need be entered only once per problem.

DATA TYPE

FILL
LOOP

ORIENTATION DATA FOR FILL

Enter unit numbers to define every position in the array. When entering data utilizing the options in this table, the count field and option field must be adjacent with no imbedded blanks. The operand field may be separated from the option field by one or more blanks. Orientation data for FILL is terminated by entering END FILL.

COUNT FIELD	OPTION FIELD	OPERAND FIELD	COMMENTS
		j	stores j at the current position in the array
i	R	j	stores j in the next i positions in the array
i	*	j	stores j in the next i positions in the array
i	$	j	stores j in the next i positions in the array
	F	j	fills remainder of the array with unit no. j starting with the current array position
	A	j	sets the current position in the array to j
i	S		increments current position in the array by i (This allows skipping i positions. The value of i may be positive or negative.)
i	Q	j	repeats the previous j entries i times. The default value of i is 1
i	N	j	repeats the previous j entries i times, inverting the sequence each time. The default value of i is 1
i	B	j	starting with the entry at -i from the current position, store entries in inverse order until position -(i+j) is reached. Default value of i=1
i	P	j	alternately stores j and -j in the next i positions of the array
i	I	j k	provides the end points, j and k, with i entries linearly interpolated between them (i.e., a total of i+2 points). At least one blank must separate j and k. When used for an integer array, the I option should only be used to generate integer steps (i.e., (k-j)/(i+1) should be a whole number).
	T		terminates the data reading for the array

ORIENTATION DATA FOR LOOP

Enter the unit number and nine numbers that define the position(s) of that unit. Data for each of these ten entries are repeated until every position in the array has been defined. Orientation data for LOOP is terminated by entering END LOOP.

ENTER DATA IN THE FORM:

DATA ENTRY	COMMENTS
LTYPE	The unit or box type. LTYPE must be greater than 0.
IX1	Starting position in the X direction. IX1 must be at least 1 and no larger than the value entered for NUX.
IX2	Ending position in the X direction. IX2 must be at least 1 and no larger than the value of NUX.
INCX	The number of units by which increments are made in the X direction.
IY1	The starting position in the Y direction. IY1 must be at least 1 and less than the value entered for NUY.
IY2	Ending position int he Y direction. IY2 must be at least 1 and no larger than the value of NUY.
INCY	The number of units by which increments are made in the positive Y direction.
IZ1	Starting position in the Z direction. IZ1 must be at least 1 and no larger than NUZ.
IZ2	Ending position in the Z direction. IZ2 must be at least 1 and no larger than NUZ.
INCZ	The number of units by which increments are made in the positive Z direction.

Table F11.4.10 Summary of biasing data

BIAS (weighting)	Format: READ BIAS keyword correlation data auxiliary data END BIAS See Sects. F11.4.7 and F11.5.8

KEYWORD	DESCRIPTION	
ID=	CORRELATION DATA will be read next.	
	CORRELATION DATA	
	id	material ID. Enter ID from table at right to use weighting data from the library
	ibgn	beginning bias ID
	iend	ending bias ID
WT= or WTS=	AUXILIARY DATA will be read next. AUXILIARY DATA will be read next.	
	AUXILIARY DATA	
	wttitl	material title (12-character maximum)
	id	material ID
	nsets	number of sets of group structures
	REPEAT	(THKINC, NUMINC, NGPWT, WTAVG) NSETS TIMES
	thkinc	thickness per increment
	numinc	number of increments
	ngpwt	number of energy groups for this set of wts
	wtavg	enter numinc x ngpwt values of wtavg

MATERIAL	ID	ENERGY GROUPS	THICKNESS/ INCREMENT
concrete	301	16,27,44,218,238	5 cm
paraffin	400	16,27,44,218,238	3 cm
water	500	16,27,44,218,238	3 cm
graphite	6100	16,27,44,218,238	20 cm

For CORRELATION DATA, the material ID is chosen from material ID column above (the keyword is ID=).
For AUXILIARY DATA, the material ID is chosen by the user and the keyword is WT= or WTS=. When AUXILIARY DATA are entered, CORRELATION DATA must also be entered to use the data.

Beginning and ending bias ID's are defined by the user. The geometry specificaiton that has the bias ID equal to the beginning bias ID utilizes the wt avg's from the first interval of material ID.

Table F11.4.11 Summary of boundary condition data

BNDS (albedo or boundary conditions)

Format: READ BNDS face code albedo name END BNDS
See Sect. F11.4.7

The sequence FACE CODE ALBEDO NAME is entered as many times as necessary to define the appropriate albedo boundary conditions. The default for all faces is vacuum.

FACE CODES FOR ENTERING BOUNDARY (ALBEDO) CONDITIONS

FACE CODE	DEFINITION	FACE CODE	DEFINITION	FACE CODE	DEFINITION	FACE CODE	DEFINITION
+XB=	positive X face	XFC=	both X faces	+YX=	positive X and Y faces	&ZY=	positive Y and Z faces
&XB=	positive X face	YFC=	both Y faces	&XY=	positive X and Y faces	-XY=	negative X and Y faces
-XB=	negative X face	ZFC=	both Z faces	&YX=	positive X and Y faces	-XZ=	negative X and Z faces
+YB=	positive Y face	+FC=	all positive faces	+XZ=	positive X and Z faces	-YZ=	negative Y and Z faces
&YB=	positive Y face	&FC=	all positive faces	+ZX=	positive X and Z faces	YXF=	all X and Y faces
-YB=	negative Y face	-FC=	all negative faces	&XZ=	positive X and Z faces	ZXF=	all X and Z faces
+ZB=	positive Z face	XYF=	all X and Y faces	&ZX=	positive X and Z faces	ZYF=	all Y and Z faces
&ZB=	positive Z face	XZF=	all X and Z faces	+YZ=	positive Y and Z faces	-YX=	negative X and Y faces
-ZB=	negative Z face	YZF=	all Y and Z faces	+ZY=	positive Y and Z faces	-ZX=	negative X and Z faces
ALL=	all 6 faces	+XY=	positive X and Y faces	&YZ=	positive Y and Z faces	-ZY=	negative Y and Z faces

ALBEDO NAMES AVAILABLE ON THE KENO V.a ALBEDO LIBRARY, FOR USE WITH THE FACE CODES

ALBEDO NAME	DESCRIPTION	ALBEDO NAME	DESCRIPTION	ALBEDO NAME	DESCRIPTION
DP0H2O	12-in. double P0 water	CONC-4	4-in. concrete differential	VACUUM	vacuum condition
DP0H2O	differential albedo with	CON4	albedo with 4 incident angles	VOID	
DP0	4 incident angles	CONC4		VACU	
DP0				VAC	
		CONC-8	8-in. concrete differential		
H2O	12-in. water differential	CON8	albedo with 4 incident angles	SPECULAR	mirror image reflection
WATER	albedo with 4 incident angles	CONC8		MIRROR	
				MIRR	
PARAFFIN	12-in. paraffin differential	CONC-12	12-in. concrete differential	SPEC	
PARA	albedo with 4 incident angles	CON12	albedo with 4 incident angles	SPE	
WAX		CONC12		MIR	
CARBON	200-cm carbon differential	CONC-15	16-in. concrete differential	PERIODIC	periodic boundary condition
GRAPHITE	albedo with 4 incident angles	CON16	albedo with 4 incident angles	PERI	
C		CONC16		PER	
ETHYLENE	12-in. polyethylene	CONC-24	24-in. concrete differential		
POLY	differential albedo with	CON24	albedo with 4 incident angles		
CH2	4 incident angles	CONC24			

Table F11.4.12 Summary of geometry data

GEOMETRY (region)

Format: READ GEOM enter geometry region data here END GEOM
See Sects. F11.4.4, F11.5.1.2, F11.5.6, and F11.5.7.

GEOMETRY REGION DATA consist of SIMPLE GEOMETRY REGION DATA and EXTENDED GEOMETRY REGION DATA.

ENTER GEOMETRY DATA IN THE FOLLOWING FORM:

OPTIONAL GLOBAL SPECIFICATION
UNIT n
OPTIONAL GEOMETRY COMMENT
GEOMETRY REGION DATA and/or EXTENDED GEOMETRY REGION DATA
* ENTER SIMPLE
GEOMETRY REGION DATA IN THE FOLLOWING FORM:

GLOBAL Enter only to specify this unit as the global unit.
UNIT n
COM=delim comment delim This optional comment can be up to 132 characters. It must begin and end with a delimiter.
fgeom mix no. bias ID dimensions optional origin data (ORIGN coordinates) optional chord data (CHORD distance)

Enter as many geometry descripton specifications as necessary to describe the unit and as many units as necessary to describe the system.

SIMPLE GEOMETRY REGION INPUT DATA REQUIREMENTS

TYPE OF DATA	TYPE 1 DATA	TYPE 2 DATA	TYPE 3 DATA	TYPE 4 DATA	TYPE 5 DATA	TYPE 6 DATA
fgeom	SPHERE HEMISPHERE HEMISPHE+X HEMISPHE-X HEMISPHE+Y HEMISPHE-Y HEMISPHE+Z HEMISPHE-Z	XCYLINDER XHEMICYL+Y XHEMICYL-Y XHEMICYL+Z XHEMICYL-Z	YCYLINDER YHEMICYL+X YHEMICYL-X YHEMICYL+Z YHEMICYL-Z	CYLINDER ZCYLINDER ZHEMICYL+X ZHEMICYL-X ZHEMICYL+Y ZHEMICYL-Y	CUBE	CUBOID
dimensions	R (radius)	R +H -H	R +H -H	R +H -H	+X -X	+X -X +Y -Y +Z -Z
optional origin coord.*	Enter the X Y Z coord. of origin	Enter the Y Z coord. of centerline	Enter the X Z coord. of centerline	Enter the X Y coord. of centerline	omit	omit
optional chord data**	Enter the dist. to plane	Enter the dist. to plane	Enter the dist. to plane	Enter the dist. to plane	omit	omit

*Enter ORIG or ORIGIN for fgeom.
**Enter CHORD for fgeom.

Note: Chord data are not applicable for SPHERE, XCYLINDER, YCYLINDER, CYLINDER, ZCYLINDER, CUBE, or CUBOID.
Origin data are not applicable for a CUBE or CUBOID.

Table F11.4.12 (continued)

GEOMETRY (region) (cont.)

ENTER EXTENDED GEOMETRY DATA IN THE FOLLOWING FORM:

fgeom ref. ID bias ID thickness per region origin coordinates nreg

EXTENDED GEOMETRY REGION INPUT DATA REQUIREMENTS

TYPE OF DATA	TYPE 1 DATA	TYPE 2 DATA	TYPE 3 DATA
fgeom	ARRAY CORE COREBDY COREBNDS COREBOUN	HOLE	REPLICATE REFLECTOR
ref. ID	array no.	emplaced unit number	mixture no. in generated regions
bias ID	omit for ARRAY	omit	first bias ID
thick./reg.	omit	omit	variable[a]
origin coord.	Enter the X Y Z coord. of most neg. pt. of array	Enter the X Y Z coord. of origin	omit
nreg	omit	omit	no. of regions to be generated

[a] The number of dimensions to be entered is the same as the region preceding the replicate or reflector specification because the generated regions have that shape. The value of the dimensions is the thickness of each generated region of material on that surface.

Table F11.4.13. Summary of mixing table data

MIXTURES

Format: READ MIXT xsec parameters END MIXT

These data are entered only if an AMPX working format library is being used. (LIB=) in the parameter data, Sect. F11.4.3. Do not enter if an ICE mixed library is used, (XSC=) in the parameter data. See Sects. F11.4.10 and F11.5.5.

XSEC PARAMETERS consists of keywords and associated values.
These parameters, if entered, need be entered only once.

KEYWORD	DEFAULT	DEFINITION
SCT=	1	no. of discrete scattering angles 0 is isotropic 1 is P1 2 is P3 3 is P5
EPS=	.00003	cross-section message cutoff value use to suppress message no. K5-60

MIXING TABLE DATA consists of

(1) a keyword and mixture ID for the mixture
The keyword is MIX=
The desired mixture number follows the keyword
(2) nuclide ID**
(3) number density**

**The sequence (2) (3) is repeated for each nuclide in the mixture.

REPEAT the sequence (1) (2)'s (3)'s until all the mixtures have been described.

Table F11.4.14 Summary of plot data

PLOT Format: READ PLOT plot parameters END PLOT plot parameters must be entered for each plot that is to be made.
See Sects. F11.4.1 and F11.5.9

KEYWORD	DEFAULT	DEFINITION
TTL=	prob. title	delim ptitl delim delim is a one-character delimiter that signals the beginning and end of the title. ptitl is the plot title (max. 132 char.)
PIC=	MAT	Type of plot: MIXTURE, UNIT NO. or BIAS ID NO. MIXTURE --------- MAT MIX MIXT MIXTURE MEDI MEDIA UNIT NO. ----------- BOX BOXT BOXTYPE UNT UNIT UNITTYPE BIAS ID NO. ------- IMP BIAS BIASID WTS WEIG WEIGHTS WGT WGTS
XUL=	prev. plot	X coord. of upper left corner of plot
YUL=	prev. plot	Y coord. of upper left corner of plot
ZUL=	prev. plot	Z coord. of upper left corner of plot
XLR=	prev. plot	X coord. of lower right corner of plot
YLR=	prev. plot	Y coord. of lower right corner of plot
ZLR=	prev. plot	Z coord. of lower right corner of plot

KEYWORD	DEFAULT	DEFINITION
UAX=	prev. plot 0 If VAX OR WAX is read	X component of direction cosine for the AX axis of the plot (across)
VAX=	prev. plot 0 IF UAX OR WAX is read	Y component of direction cosine for the AX axis of the plot (across)
WAX=	prev. plot 0 IF UAX OR VAX is read	Z component of direction cosine for the AX axis of the plot (across)
UDN=	prev. plot 0 IF VDN OR WDN is read	X component of direction cosine for the DN axis of the plot (down)
VDN=	prev. plot 0 IF UDN OR WDN is read	Y component of direction cosine for the DN axis of the plot (down)
WDN	prev. plot 0 IF UDN OR VDN is read	Z component of direction cosine for the DN axis of the plot (down)
DLX=		Horizontal spacing between points on plot
DLD=		Vertical spacing between points on plot
NAX=		No. of intervals to be printed across page
NDN=		No. of intervals to be printed down page
LPI=	8.0 (character plots) 10 (color plots)	Vertical to horizontal scaling factor for plot proportionality.
RUN=	YES	YES allows the problem to execute NO terminates problem after data checking
PLT=	YES	YES allows the plot(s) to be made NO allows reading the plot data without making a plot
SCR=	YES	Display plot method SCR=YES utilizes color plot display SCR=NO utilizes printer plot display
NCH=	CHRS*	delim CHRS delim a one character delimiter signals the beginning and end of the character string
CLR=	Table F11.4.7	num(i) red(num(i)) green (num(i)) blue (num(i)) num(i) defines mix. no., unit no., or bias ID next 3 entries define red, green, and blue components of the color representing num(i).

PLOT ORIGIN:

(1) SINGLE UNIT - coincides with origin of geometry description.

(2) BASE ARRAY - at the most negative point of the global array

(3) REFLECTED ARRAY - coincides with the origin of the CORE or ARRAY description of the global array.

*default values of CHRS are given below:

```
MEDIA 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
CHRS    1 2 3 4 5 6 7 8 9 A B C D E F G H I J K L M

MEDIA 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42
CHRS  N O P Q R S T U V W X Y Z # , $ - + ) |

MEDIA 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58
CHRS  & > : ; . - % * " = ! ( @ < / 0
```

Table F11.4.15. Summary of starting data

START Format: READ START enter start data here END START
The default value of start type is zero. See Sect. F11.4.8.

START TYPE	REQUIRED DATA	OPTIONAL DATA	STARTING DISTRIBUTION
0	none	NST XSM XSP YSM YSP ZSM ZSP RFL PSP	uniform
1	NST	XSM XSP YSM YSP ZSM ZSP RFL PSP	cosine
2	NST NXS NYS NZS FCT	XSM XSP YSM YSP ZSM ZSP RFL PSP	cosine with fraction in specified unit

START TYPE	REQUIRED DATA	OPTIONAL DATA	STARTING DISTRIBUTION
3	NST TFX TFY TFZ NXS NYS NZS	KFS PSP	spike
4	NST TFX TFY TFZ NBX	KFS PSP	multiple spikes
5	NST NBX	PSP	in specified units
6	NST TFX TFY TFZ LNU*	NXS NYS NZS KFS PS6 PSP RDU	arbitrary points

KEYWORD	DEFAULT	DEFINITION
NST=	0	start type
TFX=	0.0	X coordinate
TFY=	0.0	Y coordinate
TFZ=	0.0	Z coordinate
NXS=	0	X index of unit pos.
NYS=	0	Y index of unit pos.
NZS=	0	Z index of unit pos.
KFS=		fissile mixture no.
LNU=	0	number of last neutron
NBX=	0	source unit number
FCT=	0	fraction
XSM=	-X	-X of source cuboid
XSP=	+X	+X of source cuboid
YSM=	-Y	-Y of source cuboid
YSP=	+Y	+Y of source cuboid
ZSM=	-Z	-Z of source cuboid
ZSP=	+Z	+Z of source cuboid
RFL=	NO	start in reflector
PS6=	NO	print start 6 input
PSP=	NO	print starting points
RDU=	0	unit containing binary start data

*LNU must be the last entry for each set of start 6 data. The LNU of each successive set of data must be larger than the last.

F11.5 NOTES FOR KENO V.a USERS

This section provides assorted tips primarily designed to assist the KENO V.a user with problem mock-ups. Some information concerning methods utilized by KENO V.a is also included.

F11.5.1 DATA ENTRY

The KENO V.a data input is entered in blocks that begin and end with keywords as described in Sect. F11.4.1. Only one set of parameter data can be entered for a problem. However, for other data blocks, it is possible to enter more than one block of the same kind of data. When this is done, only the last block of that kind of data is retained for use by the problem.

Within data blocks, a number, x, can be repeated n times by specifying nRx, n*x, or n$x.

F11.5.1.1 *Multiple and Scattered Entries in the Mixing Table*

In the following examples, assume 1001 is the nuclide ID for hydrogen, 8016 is the nuclide ID for oxygen, 92235 is the nuclide ID for ^{235}U, and 92238 is the nuclide ID for ^{238}U. If a given nuclide ID is used more than once in the same mixture, the result is the summing of all the number densities associated with that nuclide. For example:

```
MIX=1 92235 4.3-2  92238 2.6-3  1001 3.7-2  92235 1.1-3  8016 1.8-2
```

would be the same as entering:

```
MIX=1 92235 4.41-2  92238 2.6-3  1001 3.7-2  8016 1.8-2
```

A belated entry for a mixture can be made as follows:

```
MIX=1 1001 6.6-2  MIX=2 92235 4.3-2 92238 2.6-3 MIX=1 8016 3.3-2
```

This is the same as entering:

```
MIX=1  1001 6.6-2 8016 3.3-2  MIX=2 92235 4.3-2  92238 2.6-3
```

F11.5.1.2 *Multiple Entries in Geometry Data*

Individual geometry regions cannot be replaced by adding an additional description. However, entire unit or box type descriptions can be replaced by adding a new description having the same unit number. The last description entered for a unit is used in the calculation. For example:

```
READ GEOM  UNIT 1 SPHERE 1 1 5.0 CUBE 0 1 10.0 -10.0
UNIT 2 CYLINDER 1 1 2.0 5.0 -5.0  CUBE  0 1 10.0 -10.0
UNIT 1  CUBOID  1 1 1.0 -1.5 2.5 -2.0 5.0 -6.0  CUBE  0 1 10.0 -10.0
END GEOM
```

is the same as entering:

READ GEOM UNIT 1 CUBOID 1 1 1.0 -1.5 2.5 -2.0 5.0 -6.0
CUBE 0 1 10.0 -10.0
UNIT 2 CYLINDER 1 1 2.0 5.0 -5.0 CUBE 0 1 10.0 -10.0 END GEOM

or

READ GEOM UNIT 2 CYLINDER 1 1 2.0 5.0 -5.0 CUBE 0 1 10.0 -10.0
UNIT 1 CUBOID 1 1 1.0 -1.5 2.5 -2.0 5.0 -6.0 CUBE 0 1 10.0 -10.0
END GEOM

The order of entry for unit or box type descriptions is not important because the unit number is assigned as the value following the word UNIT. They need not be entered sequentially nor be numbered sequentially. It is perfectly acceptable to input Units 2, 3, and 5, omitting Units 1 and 4 as long as Units 1 and 4 are not referenced in the problem. It is also acceptable to scramble the order of entry as in entering Units 3, 2, and 5.

F11.5.2 DEFAULT LOGICAL UNIT NUMBERS FOR KENO V.a

The logical unit numbers for data utilized by KENO V.a are listed in Table F11.5.1.

Table F11.5.1 KENO V.a logical unit numbers

Function	Parameter name	Unit No.	Variable name
Problem input data (EBCDIC or ASCII)		5	INPT
Problem input data (binary)		95	BIN
Program output		6	OUTPT
Albedo data	ALB=	79	ALBDO
Scratch unit	SKT=	16	SKRT
Read restart data	RST=	0.[a]	RSTRT
		34.[b]	RSTRT
Write restart data	WRS=	0.[a]	WSTRT
		35.[c]	WSTRT
Direct access storage for input data		8	DIRECT(1)
Direct access storage for supergrouped data		9	DIRECT(2)
Direct access storage for cross-section mixing		10	DIRECT(3)
Mixed cross-section data set	XSC=	14.[d]	ICEXS
Group-dependent weights	WTS=	80	WTS
AMPX working format cross sections	LIB=	0.[a]	AMPXS

[a] Defaulted to zero.
[b] Defaulted to 34 if BEG= a number greater than 1 and RSTRT = 0.
[c] Defaulted to 35 if RES= a number greater than zero and WSTRT = 0.
[d] Defaulted to 0; if LIB= a number greater than zero, ICEXS is defaulted to 14.

F11.5.3 PARAMETER INPUT

When the parameter data block is input for a problem, the same keyword may be entered several times. The last value that is entered is used in the problem. Data may be entered as follows:

READ PARAM FLX=YES NGP=1000 TME=0.5 TME=1.0
NPG=50 TME=10.0 FLX=NO
NPG=500
END PARA

This will result in the problem having **FLX=NO, TME=10.0,** and **NPG=500.** It may be more convenient for the user to insert a new value than to change the existing data.

Certain parameter default values should not be overridden unless the user has a very good reason to do so. These parameters are (1) **X1D=** which defines the number of extra 1-D cross sections. The use of extra 1-D cross sections - other than the use of the fission cross section for calculating the average number of neutrons per fission - requires programming changes to the code; (2) **NFB=** which defines the number of neutrons that can be entered in the fission bank; (3) **XFB=** which defines the number of extra positions in the fission bank (the fission bank is where the information related to a fission is stored); (4) **NBK=** which defines the number of neutrons that can be entered in the neutron bank; (5) **XNB=** which defines the number of extra positions in the neutron bank (the neutron bank contains information about each history); (6) **WTH=** which defines the factor that determines when splitting occurs; (7) **WTA=** which defines the default average weight given to a neutron that survives Russian roulette; (8) **WTL=** which defines the factor that determines when Russian roulette is played; and (9) **LNG=** which sets the maximum words of storage available to the program. It is recommended that **BUG=**, the flag for printing debug information, never be set to YES. The user would have to look at the FORTRAN coding to determine what information is printed. **BUG=YES** prints massive amounts of sparsely labeled information. The user should only rarely consider using **TRK=YES**. This generates thousands of lines of well-labeled print that provides information about each history at key locations during the tracking procedure. All other parameters can be changed at will to provide features the user wishes to activate.

F11.5.4 CROSS SECTIONS

KENO V.a always uses cross sections from a mixed cross-section data file. The format of this file is the Monte Carlo processed cross-section file from ICE-II or ICE-S[1] (Sect. F8). A mixed cross-section file can be created by (1) executing ICE, or (2) by using an AMPX working format library and entering mixing table data in KENO V.a.

F11.5.4.1 *Use an ICE Mixed Cross-Section MORSE/KENO V.a Format Library*

An ICE mixed cross-section MORSE/KENO V.a format library (premixed cross-section data file) from ICE or a previous KENO V.a case may be used. This file should be specified in the job control language on the unit number associated with the parameter XSC=. If a mixing table data block is entered, the premixed cross-section data file will be rewritten. Therefore, a mixing table should not be entered if a premixed cross-section data file is used. The user should verify that the mixtures created by ICE or the previous KENO V.a case are consistent with those used in the geometry data of the problem.

F11.5.4.2 *Use an AMPX Working Format Library*

When an AMPX working format library is used, a file definition must be specified in the job control language to specify the AMPX format working library on the unit associated with the parameter LIB=. If the mixed cross-section data file is to be saved, a file definition must be specified in the job control language to specify the mixed cross-section file on the unit associated with the parameter XSC=.

Mixing table data must always be entered when an AMPX working format library is used. IDs used in the mixing table must match the ID's on the AMPX working format library.

F11.5.4.3 *Number of Scattering Angles*

The number of scattering angles is defaulted to 1. This default is not adequate for many applications. The user should specify the scattering angle to be consistent with the cross sections being used. The number of scattering angles is entered in the cross-section mixing table by using the keyword SCT=. See Sect. F11.4.10.

The order of the last Legendre coefficient to be preserved in the scattering distribution is equal to (2 × SCT − 1). SCT=1 could be used with a P_1 cross-section set such as the 16-group Hansen Roach cross-section library, and SCT=2, for a P_3 cross-section set such as the SCALE 27-group cross-section library. Isotropic scattering is achieved by entering SCT=0.

F11.5.4.4 *Cross-Section Message Cutoff*

The cross-section message cutoff value, *pbxs*, is defaulted to 3×10^{-5}. Warning messages that are generated when errors are encountered in the P_L expansion of the group-to-group transfers will be suppressed if the P_0 cross section for that particular energy transfer is less than *pbxs*. The value of *pbxs* is specified in the cross-section mixing table by using the keyword EPS=. See Sect. F11.4.10.

The default value of *pbxs* is sufficient to assure that warning messages will not be printed for most of the SCALE P_1 and P_3 cross-section libraries. However, the 123 GROUPGMTH library requires a value of *pbxs* as large as 1×10^{-1} if P_3 cross sections are specified.

If the default value of *pbxs* allows too many warning messages to be printed, a value can be determined from the printed messages by choosing a number larger than the P_0 component on the first line as shown below.

```
THE LEGENDRE EXPANSION OF THE CROSS SECTION (P0-PN) IS
(P0)  (P1)  (P2)  ...  (Pn)
___   ___   ___        ___
THE MOMENTS CORRESPONDING TO THIS DISTRIBUTION ARE
(M1)  (M2)  ...  (Mn)
___   ___        ___
THE MOMENTS CORRESPONDING TO THE GENERATED DISTRIBUTION ARE
(M1)  (M2)  ...  (Mn)
___   ___        ___
THE LEGENDRE EXPANSION CORRESPONDING TO THESE MOMENTS IS
(P0)  (P1)  (P2)  ...  (Pn)
___   ___   ___        ___
____ MOMENTS WERE ACCEPTED
```

For the following messages, EPS=6.9-5 would cause all three messages to be suppressed. A value less than 5.615159-5 and greater than 4.767635-5 would suppress the second message, and a value less than 6.855362-5 and greater than 5.615159-5 would suppress the first two messages.

```
K5-60 THE ANGULAR SCATTERING DISTRIBUTION FOR MIXTURE 2 HAS BAD MOMENTS FOR THE TRANSFER FROM GROUP
      28 TO GROUP 72
                  1 MOMENTS WERE ACCEPTED
                THE LEGENDRE EXPANSION OF THE CROSS SECTION (P0-PN) IS
            5.615159E-05   1.155527E-06  -2.804013E-05  -1.732067E-06
                      THE MOMENTS CORRESPONDING TO THIS DISTRIBUTION ARE
            2.057870E-02   4.234578E-04   8.710817E-06
                       THE MOMENTS CORRESPONDING TO THE GENERATED DISTRIBUTION ARE

            2.057870E-02   4.235078E-04   8.710817E-06
                       THE LEGENDRE EXPANSION CORRESPONDING TO THESE MOMENTS IS

            5.615159E-05   1.155527E-06  -2.804011E-05  -1.732066E-06
                 THE WEIGHTS/ANGLES FOR THIS DISTRIBUTION ARE

            9.999995E-01   5.268617E-07

            2.057881E-02  -1.973451E-01
                       THE MOMENTS CORRESPONDING TO THIS DISTRIBUTION ARE

            2.057870E-02   4.235078E-04   8.710817E-06


K5-60 THE ANGULAR SCATTERING DISTRIBUTION FOR MIXTURE 2 HAS BAD MOMENTS FOR THE TRANSFER FROM GROUP
      31 TO GROUP 75
                   1 MOMENTS WERE ACCEPTED

                 THE LEGENDRE EXPANSION OF THE CROSS SECTION (P0-PN) IS

            4.767635E-05   7.834378E-07  -2.381887E-05  -1.174626E-06

                 THE MOMENTS CORRESPONDING TO THIS DISTRIBUTION ARE

            1.643242E-02   2.700205E-04   4.451724E-06

                 THE MOMENTS CORRESPONDING TO THE GENERATED DISTRIBUTION ARE

            1.643242E-02   2.700282E-04   4.437279E-06

                 THE LEGENDRE EXPANSION CORRESPONDING TO THESE MOMENTS IS

            4.767635E-05   7.834378E-07  -2.381885E-05  -1.174627E-06

                 THE WEIGHTS/ANGLES FOR THIS DISTRIBUTION ARE

            9.999858E-01   1.420136E-05
           1.643265E-02  -2.334324E-07
                 THE MOMENTS CORRESPONDING TO THIS DISTRIBUTION ARE

           1.643242E-02   2.700282E-04   4.437279E-06
K5-60 THE ANGULAR SCATTERING DISTRIBUTION FOR MIXTURE 2 HAS BAD MOMENTS FOR THE TRANSFER FROM GROUP
      32 TO GROUP 74
                   1 MOMENTS WERE ACCEPTED
                 THE LEGENDRE EXPANSION OF THE CROSS SECTION (P0-PN) IS
             6.855362E-05   1.341944E-06  -3.423741E-05  -2.011613E-06

                THE MOMENTS CORRESPONDING TO THIS DISTRIBUTION ARE
             1.957510E-02   3.831484E-04   7.601939E-06

                THE MOMENTS CORRESPONDING TO THE GENERATED DISTRIBUTION ARE
             1.957510E-02   3.832207E-04   7.502292E-06

                 THE LEGENDRE EXPANSION CORRESPONDING TO THESE MOMENTS IS

            6.855362E-05   1.341944E-06  -3.423740E-05  -2.011629E-06
                 THE WEIGHTS/ANGLES FOR THIS DISTRIBUTION ARE

             9.999056E-01   9.437981E-05

             1.957695E-02  -1.848551E-06

                 THE MOMENTS CORRESPONDING TO THIS DISTRIBUTION ARE

                  1.957510E-02   3.832207E-04   7.502292E-06
```

The user need not attempt to suppress all the K5-60 messages. They are printed to inform the user of the fact that the moments of the angular distribution are not moments of a valid probability distribution. The third through sixth lines of the message list the original P_n coefficients and their moments. The seventh through tenth lines list the new corrected moments and their corresponding P_n coefficients.

The weights and angles that are printed in lines 11 through 13 were generated from the corrected moments. The last two lines of the message list the moments generated from those weights and angles. They should match line 8, the moments corresponding to the generated distribution.

For most criticality problems, the first moment contributions are much more significant than the contributions of the higher order moments. Thus, the higher order moments may not affect the results significantly. The user can look at the original moments and corrected moments and make a judgment as to the significance of the change in the moments.

F11.5.5 MIXING TABLE

Mixtures can be utilized in defining other mixtures. When defining mixture numbers, care should be taken to avoid using a mixture number that is identical to a nuclide ID number if the mixture is to be used in defining another mixture. If a mixture number is defined more than once, it results in a summing effect.

The nuclide mixing loop is done before the mixture mixing loop which performs mixing in the order of data entry. Thus, the order of mixing mixtures into other mixtures is important because a mixture must be defined before it can be used in another mixture. Some examples of correct and incorrect mixing are shown below, using 1001 as the nuclide ID for hydrogen, 8016 as the nuclide ID for oxygen, 92235 as the nuclide ID for ^{235}U, and 92238 as the nuclide ID for ^{238}U.

EXAMPLES OF CORRECT USAGE

(1) READ MIXT MIX=1 1001 6.6-2 8016 3.3-2 MIX=2 1 0.5 END MIXT

This results in mixture 1 being full-density water and mixture 2 being half-density water.

(2) READ MIXT MIX=1 2 0.5 MIX=3 1 0.5 MIX=2 1001 6.6-2 8016 3.3-2 END MIXT

This results in mixture 1 being half-density water, mixture 2 being full-density water, and mixture 3 being quarter-density water. Because the nuclide mixing loop is done first, mixture 2 is created first and is available to create mixture 1, which is then available to create mixture 3.

(3) READ MIXT MIX=1 1001 6.6-2 8016 3.3-2 MIX=2 92235 7.5-4 92238 2.3-2 8016
4.6-2 1 .01 END MIXT

This results in mixture 1 being full-density water and mixture 2 being uranium oxide containing 0.01 density water.

(4) READ MIXT MIX=1 1001 6.6-2 8016 3.3-2 MIX=2 92235 4.4-2 92238 2.6-3 MIX=1
1 0.5 END MIXT

This results in mixture 1 being water at 1.5 density (1001 9.9-2 and 8016 4.95-2) and mixture 2 is highly enriched uranium metal.

EXAMPLES OF INCORRECT USAGE

(1) READ MIXT MIX=3 1 0.75 MIX=1 2 0.5 MIX=2 1001 6.6-2 8016 3.3-2 END MIXT

Here the intent is for mixture 2 to be full-density water, mixture 1 to be half-density water, and mixture 3 to be 3/8 (0.75 × 0.5) density water. Instead, the result for mixture 3 is a void, mixture 1 is half-density water, and mixture 2 is full-density water. This is because the nuclide mixing loop is done first, thus defining mixture 2. The mixture mixing loop is done next. Mixture 3 is defined to be mixture 1 multiplied by 0.75, but since mixture 1 has not been defined, 0.75 of zero is zero. Mixture 1 is then defined to be mixture 2 multiplied by 0.5. If the definition of mixture 1 preceded the definition of mixture 3, as in (2) under examples of correct usage, it would work correctly.

(2) READ MIXT MIX=1 1001 6.6-2 8016 3.3-2 MIX=1001 92235 4.4-2 92238 2.6-3 MIX=2
1001 0.5 END MIXT

This results in mixture 1 being full-density water, mixture 1001 being uranium metal and mixture 2 being hydrogen with a number density of 0.5 because 1001 is the nuclide ID number for hydrogen. When a mixture number is identical to a nuclide ID and is used in mixing, that number is assumed to be a nuclide ID rather than a mixture number. The intent was for mixture 1 to be full-density water, mixture 1001 to be uranium metal, and mixture 2 to be half-density uranium metal.

F11.5.6 GEOMETRY

In general, KENO V.a geometry descriptions consist of (1) geometry data (Sect. F11.4.4) defining the geometrical shapes present in the problem, and (2) array data (Sect. F11.4.5) defining the placement of the units that were defined in the geometry data. The geometry data block is prefaced by READ GEOM and the array data block is prefaced by READ ARRAY.

When a 3-D geometrical configuration is described as KENO V.a geometry data, it may be necessary to describe portions of the configuration individually. These individual partial descriptions of the configuration are called units. KENO V.a geometry modeling is subject to the following restrictions:

1. In general, each geometry region in a unit must completely enclose all geometry regions which precede it. Boundaries of the surfaces of the regions may be shared or tangent, but they must not intersect. The use of holes provides an exception to this complete enclosure rule.

2. All geometrical surfaces must be describable as spheres, hemispheres, cylinders, hemicylinders, cubes, or cuboids.

3. When one or more units are utilized to describe an array, each unit used in the array must have a cube or cuboid as its outer region.

4. When several units are utilized to describe an array, the adjacent faces of units in contact with each other must be the same size and shape.

5. As many holes as will snugly fit without intersecting can be placed in a region. Holes cannot intersect each other or any of the regions within the unit they are placed in. Holes are described in more detail in Sect. F11.5.6.1, and nested holes are described in Sect. F11.5.6.2.

6. Complicated systems may require multiple arrays to describe the system. Arrays may be placed in units. These units may be used to create other arrays or may be placed in other units by using holes. Multiple arrays are described in more detail in Sect. F11.5.6.3.

The geometry package allows any applicable shape to be enclosed by any other applicable shape, subject only to the complete enclosure restriction. The implication of this type of description is that the entire volume between two sequential geometrical surfaces contains only one mixture unless holes are present. Therefore, the entire volume within the surface, defined by the first geometry description in a unit, contains the mixture that is specified by that description. The volume between the surfaces, defined by two consecutive geometry descriptions, contains the mixture that is specified by the second description. A void is specified by a mixture ID of zero. If holes are present in the volume between two surfaces, the volume of that region is reduced by the hole volume(s).

If the problem requires several units to describe its geometrical characteristics, each unit that is used in an array must have as its outer surface a rectangular parallelepiped. Thus, it may be necessary to define a void region that is used to achieve the desired spacing. In order to describe the composite overall geometrical characteristics of the problem, these units may be arranged in a rectangular array by specifying the number of units in the x, y, and z directions. If more than one unit is involved, data must be entered to define the number assigned to the array and the placement of the individual units in the array. The array number, the number of units in the x, y, and z directions, and the placement data are called array data (Sect. F11.4.5).

Surrounding regions of any shape may be placed around an array and may consist of any number of regions in any order, subject to the complete enclosure restriction. All of the surrounding regions must be described about a point of origin; consequently, the location of this origin is defined by the ARRAY PLACEMENT description. The surrounding regions are then described relative to this origin.

In addition to array problems, the geometry package allows single unit problems (i.e., problems that do not contain array data). The last geometrical region in a single unit problem need not be a cube or cuboid. Boundary conditions can be applied to a single unit problem only if the outer region is a rectangular parallelepiped. Matrix information can be calculated only if the corresponding geometry description is utilized in the problem. For example, matrix by hole number requires the utilization of holes; matrix by array position requires an array, etc.

To create a geometry mock-up from a physical configuration, the user should exercise a degree of ingenuity and keep in mind the restrictions mentioned earlier. It is important to realize there may be several ways of correctly describing the same physical configuration. Careful analysis of the system can pay off in terms of a simpler mock-up and shorter computer running time. A mock-up with fewer geometry regions may run faster than the same mock-up with extraneous regions. The number of units or box types used can affect the running time, because a transformation of coordinates must be made every time a history moves from one unit into another. Thus, if the size of a unit is small, relative to the mean free path, a larger percentage of time is spent processing the transformation of coordinates.

CORE or ARRAY: The geometry word CORE was used in KENO IV and early versions of KENO V to create a rectangular parallelepiped that snugly fit the exterior of the array. The purpose of this region was to define the location of the array relative to the geometry regions external to the CORE. KENO IV and early versions of KENO V allowed only one array in a problem.

KENO V.a allows multiple arrays within a problem, so the geometry word CORE is used to specify the array number of the array, in addition to encasing it in a snug fitting rectangular parallelepiped. The data that must be entered are the geometry word (CORE), the array number, the bias ID, and the x, y, and z coordinates of the most negative point in the array. The geometry word ARRAY was added to allow specifying the array number and encasing the specified array in a snug fitting rectangular parallelepiped without having to specify the bias ID. The data that must be entered are the geometry word (ARRAY), the array number, and the x, y, and z coordinates of the most negative point in the array. The authors feel that the geometry word ARRAY is more descriptive of the function performed, but the geometry word CORE has been retained to allow executing problems that were run on early versions of KENO V.

REFLECTOR or REPLICATE: The geometry word REFLECTOR was used in KENO IV to generate cuboidal regions external to the array. If a snug fitting rectangular parallelepiped (CORE) was not entered in the problem, it was generated by the code. KENO V and KENO V.a retained the geometry word REFLECTOR but with a different function. It does not generate a snug fitting rectangular parallelepiped (CORE) for an array, and it is not limited to generating cuboidal regions. Instead, it generates regions having the shape of the geometry region preceding the REFLECTOR description. The geometry word REPLICATE is used in KENO V.a as a synonym for REFLECTOR. The data are identical for both geometry words and their function is identical. Because these geometry words produce regions having the shape of the previous region, REPLICATE may be considered to be more representative of the actual situation.

Geometry dimensions: The geometry dimensions utilized in KENO V.a have traditionally required an entry for each required dimension. For example, a 20 × 20 × 2.5-cm rectangular parallelepiped would have been described as: CUBOID 1 1 10.0 -10.0 10.0 -10.0 1.25 -1.25. By using the P option (see Table F11.4.2), the same rectangular parallelepiped could be described as: CUBOID 1 1 4P10.0 2P1.25. The P option simply repeats the dimension following the P, the number of times stated before the P and reverses the sign every other time. 6P8.0 is equivalent to 8.0 -8.0 8.0 -8.0 8.0 -8.0.

Geometry comments: A comment can be entered for each unit in the geometry region data. Similarly, a comment can be entered for each array in the array definition data. A comment can be entered using the keyword COM=. This is followed by a comment whose maximum length is 132 characters. The comment must be preceded and terminated by a delimiter character, which is the first nonblank character encountered after the COM=. One comment is allowed for each unit in the geometry region data. If multiple comments are entered for a unit, the last one is used. The comment can be entered anywhere after the UNIT NUMBER DESCRIPTION where a keyword is expected (Sect. F11.4.4). See the following example.

```
READ GEOM
UNIT 1
COM=*SPHERICAL METAL UNIT*
SPHERE 1 1 5.0
CUBE 0 1 2P5.0
UNIT 2
CYLINDER 1 1 5.0 2P5.0
CUBE 0 1 2P5.0
COM=/CYLINDRICAL METAL UNIT/
UNIT 3
HEMISPHE+X 1 1 5.0
COM=’HEMISPHERICAL METAL UNIT’
CUBE 0 1 2P5.0
UNIT 4
COM=’ARRAY OF SPHERICAL UNITS’
ARRAY 1 3*0.0
UNIT 5
COM=’ARRAY OF CYLINDRICAL UNITS’
ARRAY 2 3*0.0
UNIT 6
COM=’ARRAY OF HEMISPHERICAL UNITS’
ARRAY 3 3*0.0
END GEOM
```

One comment is allowed for each array in the ARRAY DEFINITION DATA. The rules governing these comments are the same as those listed above. However, the comment for an array must precede the UNIT ORIENTATION DESCRIPTION. It can precede the array number (Sect. F11.4.5). Examples of correct array comments are given below.

READ ARRAY
COM='ARRA OF SPHERICAL METAL UNITS'
ARA=1 NUX=2 NUY=2 NUZ=2 FILL F1 END FILL
ARA=2 COM='ARRAY OF CYLINDRICAL METAL UNITS'
NUX=2 NUY=2 NUZ=2 FILL F2 END FILL
ARA=3 NUX=2 NUY=2 NUZ=2
COM='ARRAY OF HEMISPHERICAL METAL UNITS'
FILL F3 END FILL
ARA=4 COM='COMPOSITE ARRA OF ARRAYS. Z=1 IS SPHERES, Z=2 IS CYLINDERS, Z=3 IS HEMISPHERES'
NUX=1 NUY=1 NUZ=3 FILL 4 5 6 END FILL

Some of the basics of KENO V.a geometry are illustrated in the following examples:

EXAMPLE 1. Assume a stack of six cylindrical disks, each 5 cm in radius and 2 cm thick. The bottom disk is composed of material 1, and the next disk is composed of material 2, etc., alternately throughout the stack. A square plate of material 3, 20 cm on a side and 2.5 cm thick, is centered on top of the stack. This configuration is shown in Fig. F11.5.1.

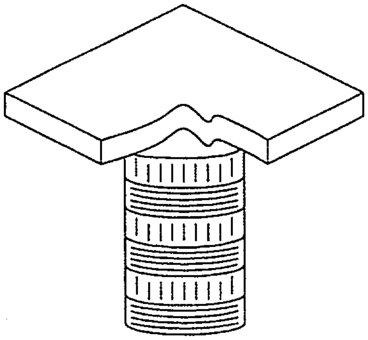


Figure F11.5.1 Stack of disks with a square cap

This problem can be described as a single unit problem, by describing the cylindrical portion first. In this instance, the origin has been chosen at the center bottom of the bottom disk. The bottom disk is defined by the first cylinder description; the next disk is defined by the difference between the first and second cylinder descriptions. That is, since they both have a radius of 5.0 and a -z length of 0.0, the first cylinder containing material 1 exists from z = 0.0 to z = 2.0 and the second cylinder, containing material 2, exists from z = 2.0 to z = 4.0. When all the disks have been described, a void cuboid having the same x and y dimensions as the square plate and the same z dimensions as the stack of disks is defined. The square plate of material 3 is then defined on top of the stack. Omission of the first cuboid description would result in the stack of disks being encased in a solid cuboid of material 3, instead of having a flat plate on top of the stack. The geometry input is shown below.

Data description 1, Example 1.

```
READ GEOM
CYLINDER 1  1     5.0 2.0    0.0
CYLINDER 2  1     5.0 4.0    0.0
CYLINDER 1  1     5.0 6.0    0.0
CYLINDER 2  1     5.0 8.0    0.0
CYLINDER 1  1     5.0 10.0   0.0
CYLINDER 2  1     5.0 12.0   0.0
CUBOID 0 1 10.0 -10.0 10.0 -10.0 12.0 0.0
CUBOID 3 1 10.0 -10.0 10.0 -10.0 14.5 0.0
```

An alternative description of the same example is given below. The origin has been chosen at the center of the disk of material 1, nearest the center of the stack. This disk of material 1 is defined by the first cylinder description, and the disks of material 2 on either side of it are defined by the second cylinder description. The top and bottom disks of material 1 are defined by the third cylinder, and the top disk of material 2 is defined by the last cylinder. The square plate is defined by the two cuboids.

Data description 2, Example 1.

```
READ GEOM
CYLINDER 1 1  5.0   1.0 -1.0
CYLINDER 2 1  5.0   3.0 -3.0
CYLINDER 1 1  5.0   5.0 -5.0
CYLINDER 2 1  5.0   7.0 -5.0
CUBOID   0 1 10.0 -10.0 10.0 -10.0 7.0 -5.0
CUBOID   3 1 10.0 -10.0 10.0 -10.0 9.5 -5.0
```

Example 1 can also be described as a bare array. Define three different unit types. Unit 1 will define a disk of material 1, Unit 2 will define a disk of material 2, and Unit 3 will define the square plate of material 3. The origin of each unit is defined at the center bottom of the disk or plate being described. The geometry input for this arrangement is shown below.

Data description 3, Example 1.

```
READ GEOM
UNIT 1
CYLINDER  1 1  5.0   2.0  0.0
CUBOID    0 1 10.0 -10.0 10.0 -10.0 2.0 0.0
UNIT 2
CYLINDER  2 1  5.0   2.0  0.0
CUBOID    0 1 10.0 -10.0 10.0 -10.0 2.0 0.0
UNIT 3
CUBOID    3 1 10.0 -10.0 10.0 -10.0 2.5 0.0
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=7 FILL 1 2 1 2 1 2 3 END ARRAY
```

If the user wishes the origin of each unit to be at its center, the geometry region data can be input as shown below. The array data would be identical to that of data description 3, Example 1.
Data description 4, Example 1.

```
READ GEOM
UNIT 1
CYLINDER 1 1 5.0  1.0 -1.0
CUBOID   0 1 10.0 -10.0 10.0 -10.0 1.0 -1.0
UNIT 2
CYLINDER 2 1 5.0  1.0 -1.0
CUBOID   0 1 10.0 -10.0 10.0 -10.0 1.0 -1.0
UNIT 3
CUBOID 3 1 10.0 -10.0 10.0 -10.0 1.25 -1.25
END GEOM
```

Be aware that each unit in a geometry description can have its origin defined independent of the other units. It would be correct to use Units 1 and 3 from data descriptions 3, and Unit 2 from data description 4. The array data would remain the same as data description 3, Example 1. The user should define the origin of each unit to be as convenient as possible for the chosen description.

Another method of describing Example 1 as a bare array is to define Unit 1 to be a disk of material 1, topped by a disk of material 2. The origin has been chosen at the center bottom of the disk of material 1. Unit 2 is the square plate of material 3 with the origin at the center of the unit. The array consists of three Unit 1's, topped by a Unit 2 as shown below.

Data description 5, Example 1.

```
READ GEOM
UNIT 1
CYLINDER 1 1 5.0  2.0 0.0
CYLINDER 2 1 5.0  4.0 0.0
CUBOID   0 1 10.0 -10.0 10.0 -10.0 4.0 0.0
UNIT 2
CUBOID   3 1 10.0 -10.0 10.0 -10.0 1.25 -1.25
END GEOM
READ ARRAY NUX=1 NUY=1 NUZ=4 FILL 3R1 2 END ARRAY
```

Example 1 can be described as a reflected array by treating the square plate as a reflector in the positive z direction. One means of describing this situation is to define Units 1 and 2 as in data description 3, Example 1. The origin of the core boundary is defined to be at the center of the array. The corresponding input geometry is shown below.

Data description 6, Example 1.

```
READ GEOM
UNIT 1
CYLINDER 1 1  5.0  2.0  0.0
CUBOID   0 1  10.0 -10.0 10.0 -10.0 2.0  0.0
UNIT 2
CYLINDER 2 1  5.0  2.0  0.0
CUBOID   0 1  10.0 -10.0 10.0 -10.0 2.0  0.0
CORE     0 1 -10.0 -10.0 -6.0
```

```
CUBOID   3 1  10.0 -10.0 10.0 -10.0 8.5 -6.0
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=6 FILL 1 2 1 2 1 2 END ARRAY
```

The user could have chosen the origin of the core boundary to be at the center bottom of the array. The last two geometry descriptions would then be:

```
CORE    0 1 -10.0 -10.0 0.0
CUBOID  3 1 10.0 -10.0 10.0 -10.0 14.5 0.0
```

or

```
ARRAY   1 -10.0 -10.0 0.0
REPLICATE  3 5*0.0 2.5 1
```

The CORE and ARRAY are interchangeable. The reflector region at the top of the array can be added by using a CUBOID or by using a REPLICATE description.

A simpler method of describing Example 1 as a reflected array is to define only one unit as in data description 5, Example 1. The square plate is treated as a reflector as in data description 6, Example 1. The input for this arrangement is given below.

Data description 7, Example 1.

```
READ GEOM
CYLINDER  1 1   5.0   2.0  0.0
CYLINDER  2 1   5.0   4.0  0.0
CUBOID    0 1  10.0 -10.0 10.0 -10.0 4.0 0.0
CORE      0 1 -10.0 -10.0  0.0
CUBOID    3 1  10.0 -10.0 10.0 -10.0 14.5 0.0
END GEOM
READ ARRAY NUX=1 NUY=1 NUZ=3 END ARRAY
```

Note that the unit orientation data need not be entered in the array data if only one unit is defined in the geometry region data. Similarly, a unit or box-type definition need not be entered when only one unit is defined.

EXAMPLE 2. Assume the stack of six disks in Example 1 is placed at the center bottom of a cylindrical container composed of material 6 whose inside diameter is 16.0 cm. The bottom and sides of the container are 0.25 cm thick, the top is open, and the total height of the container is 18.25 cm. Assume the square plate of Example 1 is centered on top of the container.

The geometry input can be described utilizing most of the data description methods associated with Example 1. One method of describing Example 2 as a single unit is given below.

Data description 1, Example 2.

```
READ GEOM
CYLINDER  1 1  5.0   1.0  -1.0
CYLINDER 2 1 5.0   3.0 -3.0
CYLINDER 1 1 5.0   5.0 -5.0
CYLINDER 2 1 5.0   7.0 -5.0
CYLINDER 0 1 8.0  13.0 -5.0
CYLINDER 6 1 8.25 13.0 -5.25
CUBOID   0 1 10.0 -10.0 10.0 -10.0 13.0 -5.25
CUBOID   3 1 10.0 -10.0 10.0 -10.0 15.5 -5.25
END GEOM
```

In the above description, the origin is defined to be at the center of the disk of material 1 nearest the center of the stack. This disk is defined by the first cylinder description. The disks of material 2 above and below it are defined by the second cylinder description. The disks of material 1 above and below them are defined by the third cylinder description. The top disk of material 2 is defined by the fourth cylinder description. The void interior of the container is defined by the fifth cylinder description. The container is defined by the last cylinder description. The first cuboid description is used to define a void whose x and y dimensions are the same as the square plate, and whose z dimensions are the same as the container. The last cuboid description defines the square plate. Omission of the first cuboid description would result in the container being encased in a solid cuboid of material 3. Thus, both cuboids are necessary to properly define the square plate.

Example 2 can be described as a reflected array. One of the descriptions uses only one unit and is similar to data description 7, example 1. This description is shown below. Note that the use of only one unit type allows omission of the unit type definition and the unit orientation data.

Data description 2, Example 2.

```
READ GEOM
CYLINDER 1 1 5.0   2.0  0.0
CYLINDER 2 1 5.0   4.0  0.0
CUBOID   0 1 5.0  -5.0  5.0  -5.0 4.0 0.0
CORE     0 1 -5.0  -5.0  0.0
CYLINDER 0 1 8.0  18.0  0.0
CYLINDER 6 1 8.25 18.0 -0.25
CUBOID   0 1 10.0 -10.0 10.0 -10.0 18.0 -0.25
CUBOID   3 1 10.0 -10.0 10.0 -10.0 20.5 -0.25
END GEOM
READ ARRAY NUX=1 NUY=1 NUZ=3 END ARRAY
```

In this data description, the first two cylinder descriptions define a disk of material 1 with a disk of material 2 directly on top of it. A tight-fitting void cuboid is placed around them so they can be stacked three high to achieve the stack of disks shown in Example 1, Fig. F11.5.1. This array comprises the core or array portion of the geometry region description. The origin of the core boundary, a tight fitting cube or cuboid that encompasses the core, is defined by the CORE description. Everything after the CORE description is considered part of the reflector. The first cylinder after the CORE defines the void interior of the cylindrical container. The next cylinder defines the walls of the container. The next-to-last cuboid defines a void volume outside the container from its bottom to its top and having the same x and y dimensions as the square plate. The last cuboid defines the square plate of material 3 that is sitting on top of the container.

Example 2 cannot be described as a reflected array if the inner radius of the container is smaller than 7.071 cm ($\sqrt{2 \times 5.0^2}$). This is the radius at which the inner boundary of the container is tangent to the cuboid around the disks. If the inner radius is smaller than 7.071 cm, the data must be described as a single unit; or alternatively, the container must be included within the units. Assume the container of Example 2 has an inner radius of 6.0 cm with all other data remaining the same. An example of the geometry data describing this configuration is given below in data description 3, Example 2.

Data description 3, Example 2.

```
READ GEOM
UNIT 1
CYLINDER 6 1  6.25   0.25   0.0
CUBOID   0 1 10.0  -10.0  10.0 -10.0 0.25  0.0
UNIT 2
CYLINDER 1 1  5.0    2.0    0.0
CYLINDER 2 1  5.0    4.0    0.0
CYLINDER 0 1  6.0    4.0    0.0
CYLINDER 6 1  6.25   4.0    0.0
CUBOID   0 1 10.0  -10.0  10.0 -10.0 4.0   0.0
UNIT 3
CYLINDER 0 1  6.0    3.0   -3.0
CYLINDER 6 1  6.25   3.0   -3.0
CUBOID   0 1 10.0  -10.0  10.0 -10.0 3.0  -3.0
CUBOID   3 1 10.0  -10.0  10.0 -10.0 5.5  -3.0
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=5 FILL 1 3R2 3 END ARRAY
```

In the above description, Unit 1 is the bottom of the cylindrical container. The void cuboid is only as tall as the bottom of the container, and its x and y dimensions are the same as the square plate on top of the container. If all the units in the array utilize these same dimensions in the x and y directions, the restriction that adjacent faces of units in contact with each other be the same size and shape is satisfied. This array is stacked in the z direction, so all units must have the same overall dimensions in the x direction and in the y direction. Unit 2 will be used in the array three times to create the stack of disks. It contains a disk of material 1, topped by a disk of material 2. The portion of the container that contains the disks and the cuboid that defines the outer boundaries of the unit are included in Unit 2. Unit 3 describes the empty top portion of the container and the square plate on top of it. The z dimensions of Unit 3 were determined by subtracting three times the total z dimension of Unit 2 from the inside height of the container [18.0 - (3 × 4.0) = 6.0]. This can also be determined from the overall height of the container by subtracting off the bottom thickness of the container and three times the height of Unit 2 [18.25 - 0.25 - (3 × 4.0) = 6.0]. The origin of Unit 3 is located at the center of this distance. If the origin were chosen at the bottom of that height, it would be described as:

```
UNIT 3
CYLINDER 0 1  6.0    6.0  0.0
CYLINDER 6 1  6.25   6.0  0.0
CUBOID   0 1 10.0  -10.0 10.0 -10.0 6.0  0.0
CUBOID   3 1 10.0  -10.0 10.0 -10.0 8.5  0.0
```

EXAMPLE 3. Refer to Example 1, Fig. F11.5.1, and imagine a hole 1.5 cm in diameter is drilled along the centerline of the stack through the disks and the square plate. This eliminates the possibility of describing the system as a single unit because the hole in the center of the alternating materials of the stack cannot be described in a manner that allows each successive geometry region to encompass the regions interior to it. Therefore, it must be described as an array. The square plate on the top of the disks is defined as a unit in the array. In the geometry description given below, the square plate is defined in Unit 3.

Data description 1, Example 3.

```
READ GEOM
UNIT 1
CYLINDER  0 1  0.75   2.0  0.0
CYLINDER  1 1  5.0    2.0  0.0
CUBOID    0 1 10.0 -10.0 10.0 -10.0 2.0 0.0
UNIT 2
CYLINDER  0 1  0.75   2.0  0.0
CYLINDER  2 1  5.0    2.0  0.0
CUBOID    0 1 10.0 -10.0 10.0 -10.0 2.0 0.0
UNIT 3
CYLINDER  0 1  0.75   2.5  0.0
CUBOID    3 1 10.0 -10.0 10.0 -10.0 2.5 0.0
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=7 FILL 1 2 2Q2 3 END FILL END ARRAY
```

In data description 1, Example 3 above, Unit 1 describes a disk of material 1 with a hole through its centerline. The first cylinder defines the hole, the second defines the rest of the disk, and the cuboid defines the size of the unit to be consistent with the square plate so they can be stacked together in an array. Unit 2 describes a disk of material 2 in similar fashion. Unit 3 describes the square plate of material 3 with a hole through its center. The cylinder defines the hole and the cuboid defines the square plate. These three units are stacked in the z direction to achieve the composite system. This is represented by FILL 1 2 2Q2 3. The 2Q2 repeats the two entries preceding the 2Q2, two times. Alternatively, this can be achieved by entering FILL 1 2 1 2 1 2 3 END FILL. The same array can also be achieved using the LOOP option. An example of the data for this option is:

```
LOOP 1 6R1  1 5 2  2 6R1 2 6 2  3 6R1 7 7 1  END LOOP.
```

Unit 1 is placed at the x = 1, y = 1, and z = 1,3,5 positions of the array by entering 1 6R1 1 5 2. Unit 2 is positioned at the x = 1, y = 1 and z = 2,4,6 positions in the array by entering 2 6R1 2 6 2. Unit 3 is placed at the x = 1, y = 1, z = 7 position of the array by entering 3 6R1 7 7 1. See Sect. F11.4.5 for additional information regarding array specifications. Table F11.4.2 lists other available input options.

EXAMPLE 4. Assume two large cylinders, 2.5 cm in radius and 5 cm long, are connected by a smaller cylinder, 0.5 cm in radius and 10 cm long, as shown in Fig. F11.5.2. All of the cylinders are composed of material 1. By starting the geometry description in the small cylinder, this system can be described as a single unit.

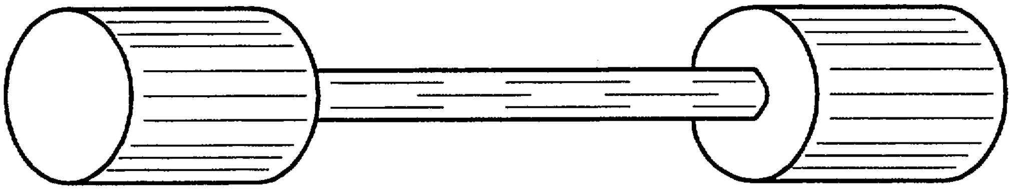


Figure F11.5.2 Two large cylinders joined axially by a small cylinder

Data description 1, Example 4.

```
READ GEOM
CYLINDER 1 1 0.5  5.0  -5.0
CYLINDER 0 1 2.5  5.0  -5.0
CYLINDER 1 1 2.5 10.0 -10.0
END GEOM
```

The origin is at the center of the small cylinder which is described by the first cylinder description. The second cylinder description defines a void cylinder surrounding the small cylinder. Its radius is the same as the large cylinders, and its height (length) coincides with that of the small cylinder. The last cylinder description defines the large cylinders on either end of the small cylinder. Because this problem does not specify otherwise, the length of the cylinders is assumed to coincide with the Z axis.

EXAMPLE 5. Assume two large cylinders with a center-to-center spacing of 15 cm, each having a radius of 2.5 cm and length of 5 cm, are connected radially by a small cylinder having a radius of 1.5 cm, as shown in Fig. F11.5.3.

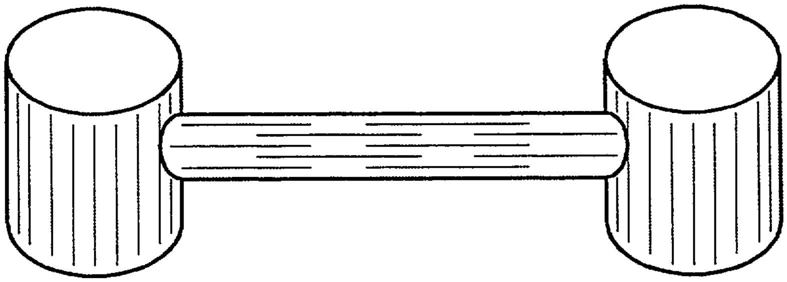


Figure F11.5.3 Two large cylinders radially connected by a small cylinder

This system cannot be rigorously described in KENO V.a geometry because the intersection of the cylinders cannot be described. However, it can be approximated two ways, as shown in Fig. F11.5.4. The top approximation is described in data description 1, example 5. The bottom approximation is described in data description 2, example 5, and data description 3, example 5. These may be poor approximations for criticality safety calculations.

ORNL-DWG 83-13259

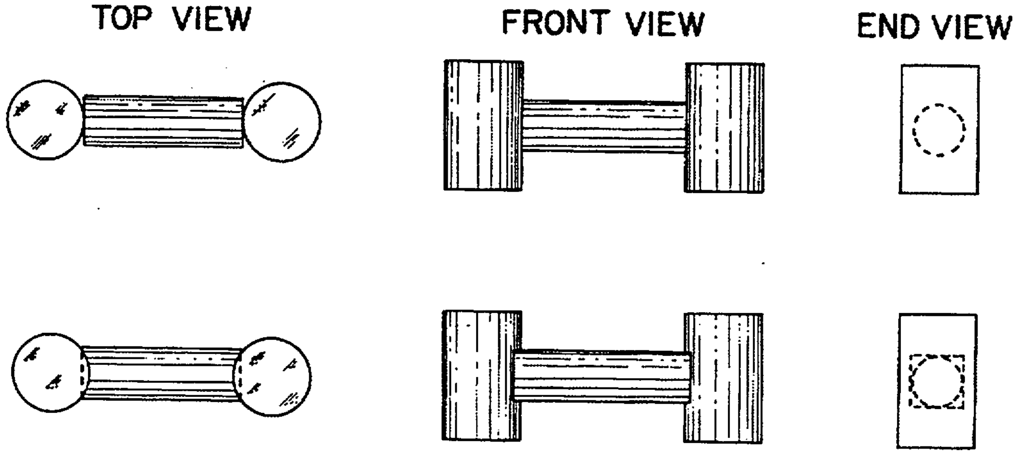


Figure F11.5.4 KENO V.a approximations of cylindrical intersections

Data description 1, Example 5.

```
READ GEOM
UNIT 1
CYLINDER  1 1 2.5 2.5 -2.5
CUBE      0 1 2.5 -2.5
UNIT 2
XCYLINDER 1 1 1.5 5.0 -5.0
CUBOID    0 1 5.0 -5.0 2.5 -2.5 2.5 -2.5
END GEOM
READ ARRAY NUX=3 NUY=1 NUZ=1 FILL 1 2 1 END ARRAY
```

Unit 1 defines a large cylinder, and Unit 2 describes the small cylinder. In both units the origin is at the center of the cylinder. The large cylinders have their centerlines along the Z axis and the small cylinder has its length along the X axis.

Data description 2, Example 5.

```
READ GEOMETRY
UNIT 1
CYLINDER 1 1 2.5 1.0 0.0
CUBOID 0 1 4P2.5 1.0 0.0
UNIT 2
ZHEMICYL-X 1 1 2.5 2P1.5 CHORD 2.0
CUBOID 0 1 2.0 3P-2.5 2P1.5
UNIT 3
ZHEMICYL+X 1 1 2.5 2P1.5 CHORD 2.0
CUBOID 0 1 2.5 -2.0 2P2.5 2P1.5
UNIT 4
XCYLINDER 1 1 1.5 2P5.5
CUBOID 0 1 2P5.5 2P2.5 2P1.5
UNIT 5
CUBOID 0 1 2P5.0 2P2.5 1.0 0.0
UNIT 6
ARRAY 1 3*0.0
UNIT 7
ARRAY 2 3*0.0
END GEOMETRY
READ ARRAY ARA=1 NUX=3 NUY=1 NUZ=1 FILL 1 5 1 END FILL
ARA=2 NUX=3 NUY=1 NUZ=1 FILL 2 4 3 END FILL
ARA=3 NUX=1 NUY=1 NUZ=3 FILL 6 7 6 END FILL
END ARRAY
```

This geometry description uses arrays of arrays (see Section F11.5.6.3) to describe the bottom approximation of Fig. F11.5.4. Unit 1 defines a large cylinder 2.5 cm in radius and 1.0 cm tall inside a close-fitting cuboid. This is used in both large cylinders as the portion of the large cylinder that exists both above and below the region where the small cylinder joins it. Unit 5 is the spacing between the tops of the two large cylinders and the spacing between the bottoms of the two large cylinders. Array 1 thus defines the bottom of the system: two short cylinders (Unit 1's) separated by 10 cm (Unit 5 is the separation). Unit 6 contains array 1.

Unit 2 is the left "hemicylinder" that adjoins the horizontal cylinder, and Unit 3 is the right "hemicylinder" that adjoins the horizontal cylinder. Unit 4 defines the horizontal cylinder. Array 2 contains Units 2, 4, and 3, left to right. This defines the central portion of the system where the horizontal cylinder adjoins the two "hemicylinders." These "hemicylinders" are larger than half cylinders. Unit 7 contains array 2. The entire system is achieved by stacking a Unit 6 above and below the Unit 7 as defined in array 3, the global array.

Data description 3, Example 5.

```
READ GEOMETRY
UNIT 1
CYLINDER 1 1 2.5 1.0 0.0
```

```
UNIT 2
CYLINDER 1 1 2.5 1.0 0.0
CUBOID 0 1 17.5 -2.5 2P2.5 1.0 0.0
HOLE 1 15.0 0.0 0.0
UNIT 3
ZHEMICYL-X 1 1 2.5 2P1.5 CHORD 2.0
UNIT 4
ZHEMICYL+X 1 1 2.5 2P1.5 CHORD 2.0
UNIT 5
XCYLINDER  1 1 1.5 2P5.5
CUBOID 0 1  2P10.0 2P2.5 2P1.5
HOLE 3 -7.5 2*0.0
HOLE 4 7.5 2*0.0
END GEOMETRY
READ ARRAY
ARA=1 NUX=1 NUY=1 NUZ=3 FILL 2 5 2 END FILL
END ARRAY
```

This geometry description uses holes (see Sect. F11.5.6.1) to describe the bottom approximation of Fig. F11.5.4. Unit 1 defines a large cylinder 2.5 cm in radius and 1.0 cm tall. Unit 2 defines the same cylinder within a cuboid that extends from x = -2.5 to x = 17.5, from y = -2.5 to y = 2.5, and z = 0.0 to z = 1.0. The origin of the cylinder is at (0.0,0.0,0.0). Thus Unit 2 describes the top and bottom of the cylinder on the left. Unit 1 is positioned within this cuboid as a hole with its origin at (15.0,0.0,0.0) to describe the top and bottom of the cylinder on the right. Unit 3 is the left "hemicylinder" that adjoins the horizontal cylinder, and Unit 4 is the right "hemicylinder" that adjoins the horizontal cylinder. Unit 5 defines the horizontal cylinder with its origin at the center within a cuboid that extends from x = -10.0 to x = +10.0, y = -2.5 to y = 2.5, and z = -1.5 to z = 1.5. Unit 3 is positioned to the left of the horizontal cylinder, and Unit 4 is positioned to the right of the horizontal cylinder by using holes. The entire system is achieved by stacking a Unit 2 above and below Unit 5 as shown in the array data.

This same geometry description can be used with Unit 2 redefined to have its origin defined so the unit extends from x = -10 to x = 10, y = -2.5 to y = 2.5, and z = 0.0 to z = 1. In this instance, the geometry data would be identical except for Unit 2. This alternative description of Unit 2 is

```
UNIT 2
CYLINDER 1 1 2.5 1.0 0.0 ORIGIN -7.5 0.0
CUBOID 0 1 2P10.0 2P2.5 1.0 0.0
HOLE 1 7.5 0.0 0.0
```

EXAMPLE 6. Assume 2 small cylinders 1.0 cm in radius and 10 cm long are connected by a large cylinder 2.5 cm in radius and 5 cm long as shown in Fig. F11.5.5.

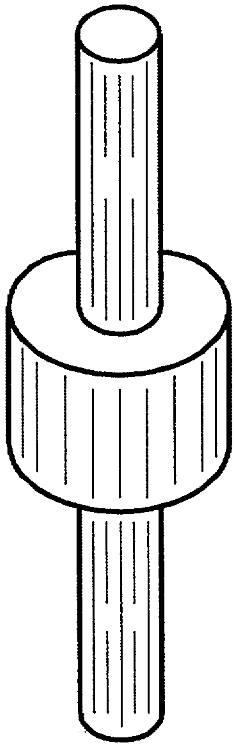


Figure F11.5.5 Two small cylinders joined axially by a large cylinder

This problem is very similar to example 4, but it cannot be described as a single unit. It must be described as an array. Unit 1 defines the large cylinder, and Unit 2 defines the small cylinder. The origin of each unit is at its center. The composite system consists of two Unit 2's and one Unit 1 as shown below. Assume the centerline of the cylinders lies along the Z axis.

Data description 1, Example 6.

```
READ GEOM
UNIT 1
CYLINDER  1 1 2.5 2.5 -2.5
CUBE      0 1 2.5 -2.5
UNIT 2
CYLINDER  1 1 1.0 5.0 -5.0
CUBOID    0 1 2.5 -2.5 2.5 -2.5 5.0 -5.0
END GEOM
READ ARRAY NUX=1 NUY=1 NUZ=3 FILL 2 1 2 END ARRAY
```

EXAMPLE 7. Assume an 11 × 5 × 3 square-pitched array of spheres of material 1, radius 3.75 cm, with a center-to-center spacing of 10 cm in the x, y, and z directions. The data for this system are given below.

Data description 1, Example 7.

```
READ GEOM
SPHERE 1 1 3.75
CUBE   0 1 5.0 -5.0
END GEOM
READ ARRAY NUX=11  NUY=5 NUZ=3 END ARRAY
```

EXAMPLE 8. Assume an 11 × 5 × 3 square-pitched array of spheres of material 1 whose radius is 3.75 cm, and whose center-to-center spacing is 10 cm in the x direction, 15 cm in the y direction, and 20 cm in the z direction. The input for this geometry is given below.

Data description 1, Example 8.

```
READ GEOM
SPHERE  1 1 3.75
CUBOID  0 1 5.0  -5.0 7.5 -7.5 10.0 -10.0
END GEOM
READ ARRAY  NUX=11 NUY=5 NUZ=3 END ARRAY
```

EXAMPLE 9. Assume an 11 × 5 × 3 square-pitched array of spheres of material 1 whose radius is 3.75 cm, and whose center-to-center spacing is 10 cm in the x, y, and z directions. This array is reflected by 30 cm of material 2 (water) on all faces and weighted tracking (biasing) is to be used in the water reflector. The array spacing defines the perpendicular distance from the outer layer of spheres to the reflector to be 5 cm in the x, y, and z directions. The geometry input for this system is given below.

Data description 1, Example 9.

```
READ GEOM
SPHERE     1 1  3.75
CUBE       0 1  5.0  -5.0
CORE       0 1 -55.0  -25.0  -15.0
REFLECTOR   2 2 6*3.0  10
END GEOM
READ ARRAY  NUX=11 NUY=5 NUZ=3 END ARRAY
READ BIAS ID=500 2 11 END BIAS
```

The core boundary defines the origin of the reflector to be at the center of the array. The 6*3.0 in the reflector description repeats the 3.0 six times. The reflector card is used to generate ten reflector regions, each 3.0 cm thick, on all six faces of the array. The first bias ID is 2, so the last bias ID will be 11 if 10 regions are created. The biasing data block is necessary to apply the desired weighting or biasing function to the reflector. The biasing material ID is obtained from Table F11.4.5. If the biasing data block is omitted from the problem description, the 10 reflector regions will not have a biasing function applied to them, and the default value of the average weight will be used. This may cause the problem to execute more slowly, and therefore require the use of more computer time.

EXAMPLE 10. Assume the reflector in Example 9 is present only on both x faces, both y faces, and the negative z face. The reflector is only 15.24 cm thick on these faces. The top of the array (positive z face) is unreflected.

Data description 1, Example 10.

```
READ GEOM
SPHERE    1 1   3.75
CUBE      0 1   5.0  -5.0
CORE      0 1 -55.0  -25.0 -15.0
REFLECTOR 2 2 4*3.0    0.0   3.0  5
REFLECTOR 2 7 4*0.24   0.0   0.24 1
READ ARRAY NUX=11 NUY=5 NUZ=3 END ARRAY
READ BIAS ID=500 2 7 END BIAS
```

The first reflector description generates five regions around the array, each region being 3.0 cm thick in the +x, -x, +y, -y, and -z directions, and of zero thickness in the +z direction. This defines a total thickness of 15 cm of reflector material on the appropriate faces. The second reflector description generates the last 0.24 cm of material 2 on those faces. Thus, the total reflector thickness is 15.24 cm on each face of the array, except the top which has no reflector. Five reflector regions were generated by the first reflector description, and one was generated by the second reflector description; so, six biasing regions must be defined in the biasing data. Thus, the beginning bias ID is 2, and the ending bias ID is 7. The biasing material ID and thickness per region are obtained from Table F11.4.5. The thickness per region should be very nearly the thickness per region from the table to avoid overbiasing in the reflector. Partial increments at the outer region of a reflector are exempt from this recommendation. If a biasing function is not to be applied to a region generated by the reflector card, the thickness per region can be any desired thickness and the biasing data block is omitted.

EXAMPLE 11. Assume the array of example 7 has the central unit of the array replaced by a cylinder of material 4, 5 cm in radius and 10 cm tall. Assume a 20-cm-thick spherical reflector of material 3 (concrete) is positioned so its inner radius is 65 cm from the center of the array. The minimum inner radius of a spherical reflector for this array is 62.25 cm ($\sqrt{55^2 + 25^2 + 15^2}$). If the inner radius is smaller than this, the problem cannot be described using KENO V.a geometry.

Data description 1, Example 11.

```
READ GEOM
UNIT 1
SPHERE 1 1 3.75
CUBE 0 1 5.0 -5.0
UNIT 2
CYLINDER 4 1 5.0 5.0 -5.0
CUBE 0 1 5.0 -5.0
```

```
CORE 0 1 -55.0 -25.0 -15.0
SPHERE 0 1 65.0
REPLICATE 3 2 5.0 4
END GEOM
READ ARRAY NUX=11 NUY=5 NUZ=3 LOOP 1  1  11  1  1 5 1  1 3 1
2  6 6 1  3 3 1  2 2 1 END ARRAY
READ BIAS ID=301 2 5 END BIAS
```

Unit 1 describes the sphere and spacing utilized in the array. Unit 2 defines the cylinder that is located at the center of the array. The CORE defines the origin of the reflector to be at the center of the array. The sphere following the core description defines the inner radius of the reflector. The replicate card will generate four spherical regions of material 3, each 5.0 cm thick. The first 10 entries following the word LOOP fills the 11 × 5 × 3 array with Units 1. The next 10 entries position Unit 2 at the center of the array (x = 6, y = 3, and z = 2), replacing the Unit 1 that had been placed there by the first 10 entries. The biasing data block is used to apply the biasing function for concrete to the generated reflector regions.

EXAMPLE 12. Assume a data profile such as fission densities is desired in a cylinder at 1-cm intervals in the radial direction and 1.5-cm intervals axially. The cylinder, composed of material 1, has a radius of 15 cm and a height of 45 cm. The REPLICATE or REFLECTOR description can be used to generate these regions as shown below. A biasing data block is not entered because default biasing is desired throughout the cylinder.

Data description 1, Example 12.

```
READ GEOM
CYLINDER  1 1 1.0 1.5  -1.5
REFLECTOR 1 2 1.0 2*1.5 14
END GEOM
```

F11.5.6.1 *Use of Holes in the Geometry*

Section F11.5.6 tells how each KENO V.a geometry region in a unit must completely enclose all previously described regions in that unit. Holes can be used to circumvent this restriction to some degree. A HOLE is a means of placing an entire unit within a geometry region. A separate HOLE description is required for every location in a geometry region where a unit is to be placed. The information contained in a hole description is: (1) the geometry word, HOLE, (2) the unit number of the unit to be placed, and (3) the x, y, and z coordinates specifying where the origin of the placed unit is to be located. A hole is placed inside the geometry region that precedes it (excluding holes ... i.e., if a CUBE geometry region is followed by four HOLE descriptions, all four of the HOLES are located within the CUBE.) Holes are subject to the restriction that they cannot intersect any other geometry region. Holes can be nested to any depth (see Sect. F11.5.6.2). It is not advisable to use holes that are tangent to other holes or geometry regions, although it is theoretically possible to do so. Frequently holes that are exactly tangent to each other or to other geometry regions may fail to run because the computer code finds that the regions are intersecting due to precision and roundoff. It is not uncommon for a problem that runs on one type of computer to fail on another type using the same data. Therefore, it is recommended that tangency and boundaries shared with holes be avoided.

Tracking in regions that contain holes is less efficient than tracking in regions that do not contain holes. Therefore holes should be used only when the system cannot be easily described by conventional methods. One example of the use of holes is shown in Fig. F11.5.6, representing nine close-packed rods in an annulus.

The large rods are 1.4 cm in radius and composed of mixture 3. The small rods are 0.6 cm in radius and composed of mixture 1. The inside radius of the annulus is 3.6 cm, and the outside radius is 3.8 cm. The annulus is made of mixture 2. The rods and annulus are both 30 cm long. The annulus is centered in a cuboid having an 8-cm-square cross section and a length of 32 cm. All nonshaded areas are void. In this mock-up, a small rod is defined by Unit 1 and a large rod is defined by Unit 2. Unit 3 defines the central small rod, the annulus and the cuboid. Units 1 and 2 are placed within the annulus using holes.

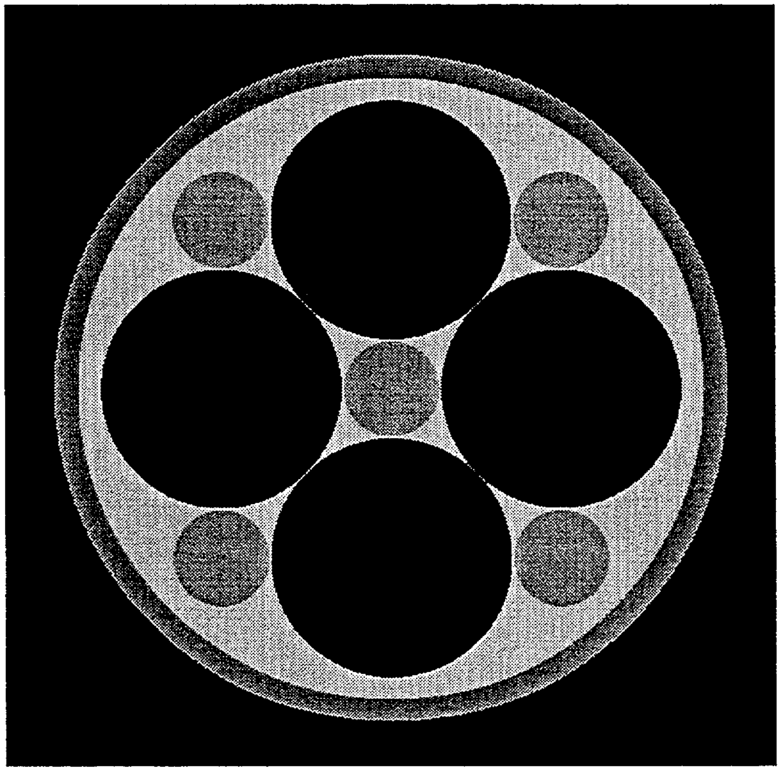


Figure F11.5.6 Close-packed rods in an annulus

The geometry mock-up can be given by:

```
READ GEOM
UNIT 1
CYLINDER 1 1 0.6 2P15.0
```

```
UNIT 2
CYLINDER 3 1 1.4 2P15.0
UNIT 3
CYLINDER 1 1 0.6 2P15.0
CYLINDER 0 1 3.6 2P15.0
HOLE 2 0.0 -2.0 0.0
HOLE 1 2.0 -2.0 0.0
HOLE 2 2.0 0.0 0.0
HOLE 1 2.0 2.0 0.0
HOLE 2 0.0 2.0 0.0
HOLE 1 -2.0 2.0 0.0
HOLE 2 -2.0 0.0 0.0
HOLE 1 -2.0 -2.0 0.0
CYLINDER  2 1 3.8 2P15.0
CUBOID    0 1 4P4.0 2P16.0
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=1  FILL F3 END ARRAY
```

The first HOLE description represents the bottom large rod. It says to take Unit 2 and place its origin at (0.0,-2.0,0.0) relative to the origin of Unit 3. The second HOLE description represents the small rod to the right of the large rod just discussed. It places the origin of Unit 1 at (2.0,-2.0,0.0) in Unit 3. The third HOLE description represents the large rod to the right. It places the origin of Unit 2 at (2.0,0.0,0.0) in Unit 3. This procedure is repeated in a counterclockwise direction until all eight rods have been placed within the region that defines the inner surface of the annulus. The CYLINDER that defines the outer surface of the annulus is described after all the holes for the previous region have been placed. Then the outer cuboid is described. This example illustrates that a unit that is to be placed using a HOLE description need not have a cube or cuboid as its last region.

An alternative mock-up for this problem would specify Unit 3 as the global unit and omit the array data. Another method would omit the first region in Unit 3 and put it in as a hole. The hole description for the central rod would be: HOLE 1 3*0.0.

The order of the HOLE cards in any given region is not important (they can be interchanged with each other randomly). However, they must always appear immediately after the region in which they are placed.

An array of the arrangement shown in Fig. F11.5.6 can be easily described by altering the array description data. For example, a 5 × 3 × 2 array of these shapes with a center-to-center spacing of 8 cm in x and y and 32 cm in z can be achieved by utilizing the following array data:

```
READ ARRAY  NUX=5 NUY=3 NUZ=2  FILL F3  END FILL  END ARRAY
```

or

```
READ ARRAY  NUX=5 NUY=3 NUZ=2  FILL 30*3  END FILL  END ARRAY
```

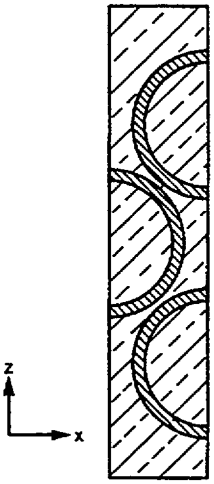
or

```
READ ARRAY  NUX=5 NUY=3 NUZ=2  LOOP  3 1 5 1 1 3 1 1 2 1  END ARRAY
```

Another example of the use of holes is shown in Fig. F11.5.7.

Assume that an infinite linear array of annular rods are stacked three high in a triangular pitch. The array is infinite in the x direction, and the annular rods are stacked three high in the z direction. The rods are finite in the y direction and stacked only one deep in y. The pitch is $\sqrt{13}$. This array can be created by describing the geometrical arrangement shown in Fig. F11.5.7 and applying specular or mirror image reflection on both x faces.

The annular rods are all composed of mixture number one, have an inside radius of 1.3 cm, an outside radius of 1.6 cm, and are 24 cm long. The pitch is $\sqrt{13}$, approximately 3.606 cm. This system can be described in many ways.



ORNL-DWG 83-10098

Figure F11.5.7 Annular rods in triangular pitch lattice

The following geometry mock-up assumes the origin of the basic unit to be on the left face (-x face), halfway up the cuboid. This is defined as Unit 1. Unit 2 is a hemicylindrical annulus which will be placed along the right face (+x face) using holes.

```
READ GEOM
UNIT 1
YHEMICYL+X  0 1 1.3 2P12.0
YHEMICYL+X  1 1 1.6 2P12.0
CUBOID 0 1 2.0 0.0 2P12.0 2P5.0
HOLE 2 2.0 0.0 -3.0
HOLE 2 2.0 0.0 3.0
UNIT 2
YHEMICYL-X  0 1 1.3 2P12.0
YHEMICYL-X  1 1 1.6 2P12.0
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=1  FILL F1  END ARRAY
READ BOUNDS  XFC= MIRROR  END BOUNDS
```

In the above geometry description, the hemicylindrical annulus on the left (-x face) is described in Unit 1. A hemicylindrical annulus that is oriented in the opposite direction is described in Unit 2. The first HOLE description locates the origin of Unit 2 at (2.0,0.0,-3.0) in Unit 1. This positions the lower hemicylindrical annulus as shown in Fig. F11.5.7. The second HOLE description locates the origin of Unit 2 at (2.0,0.0,3.0) in Unit 1. This positions the top hemicylindrical annulus as shown in the figure. The two holes can be interchanged without altering the results. The order in which holes are described is not important.

If Units 1 and 2 are interchanged, the geometry mock-up would be as follows:

```
READ GEOM
UNIT 1
YHEMICYL-X  0 1  1.3  2P12.0
YHEMICYL-X  1 1  1.6  2P12.0
UNIT 2
YHEMICYL+X  0 1  1.3  2P12.0
YHEMICYL+X  1 1  1.6  2P12.0
CUBOID     0 1  2.0  0.0  2P12.0  2P5.0
HOLE       1   2.0   0.0  -3.0
HOLE       1   2.0   0.0   3.0
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=1  FILL F2 END ARRAY
READ BOUNDS  XFC=MIRROR  END BOUNDS
```

In the above mock-up, Unit 2 describes the hemicylindrical annulus on the left and the cuboid exactly as Unit 1 did in the previous geometry description. Unit 1 now describes a hemicylindrical annulus that is oriented in the opposite direction (exactly the same as Unit 2 in the previous description). The first HOLE description places the origin of Unit 1 at (2.0,0.0,-3.0) in Unit 2. The second HOLE description places the origin of Unit 1 at (2.0,0.0,3.0) in Unit 2. This generates the same configuration as shown in Fig. F11.5.7.

Another way of describing this problem uses more unit descriptions. Define the origin of the basic unit to be on the right face at the origin of the lower annulus and call it Unit 3. Define Unit 1 to be a hemicylindrical annulus that has its origin on the left face. Define Unit 2 to be a hemicylindrical annulus that has its origin on the right face. The geometry mock-up for this situation is:

```
READ GEOM
UNIT 1
YHEMICYL+X  0 1 1.3 2P12.0
YHEMICYL+X  1 1 1.6 2P12.0
UNIT 2
YHEMICYL-X  0 1 1.3 2P12.0
YHEMICYL-X  1 1 1.6 2P12.0
UNIT 3
YHEMICYL-X  0 1 1.3 2P12.0
YHEMICYL-X  1 1 1.6 2P12.0
CUBOID  0 1 0.0 -2.0 2P12.0 8.0 -2.0
HOLE  1 -2.0 0.0 3.0
HOLE  2 0.0 0.0 6.0
END GEOM
```

```
READ ARRAY  NUX=1 NUY=1 NUZ=1  FILL F3 END ARRAY
READ BOUNDS  XFC=MIRROR  END BOUNDS
```

Assume that the origin of the basic unit is at the center of the cuboid and include the lower right hemicylindrical annulus in the basic unit. Define the basic unit to be Unit 3. Define Unit 1 to be the hemicylindrical annulus having its origin on the left face. Define Unit 2 to be the upper hemicylindrical annulus. This mock-up will be the same as the previous one except for changes required by the shift in the location of the origin.

```
READ GEOM
UNIT 1
YHEMICYL+X  0 1 1.3 2P12.0
YHEMICYL+X  1 1 1.6 2P12.0
UNIT 2
YHEMICYL-X  0 1 1.3 2P12.0
YHEMICYL-X  1 1 1.6 2P12.0
UNIT 3
YHEMICYL-X  0 1 1.3 2P12.0 ORIGIN 1.0 -3.0
YHEMICYL-X  1 1 1.6 2P12.0 ORIGIN 1.0 -3.0
CUBOID  0 1 2P1.0 2P12.0 2P5.0
HOLE  1 -1.0 0.0 0.0
HOLE  2 1.0 0.0 3.0
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=1  FILL F3 END ARRAY
READ BOUNDS  XFC=MIRROR  END BOUNDS
```

Define Unit 1 to be a hemicylindrical annulus oriented like the top and bottom ones in Fig. F11.5.7. Define Unit 2 to be a hemicylindrical annulus oriented like the middle one in Fig. F11.5.7. Define Unit 3 to be the cuboid with the origin at its center. Use holes to place all three hemicylindrical annuli in the cuboid. The geometry mock-up for this situation follows:

```
READ GEOM
UNIT 1
YHEMICYL-X  0 1 1.3 2P12.0
YHEMICYL-X  1 1 1.6 2P12.0
UNIT 2
YHEMICYL+X  0 1 1.3 2P12.0
YHEMICYL+X  1 1 1.6 2P12.0
UNIT 3
CUBOID  0 1 2P1.0 2P12.0 2P5.0
HOLE  1 1.0 0.0 -3.0
HOLE  1 1.0 0.0 3.0
HOLE  2 -1.0 0.0 0.0
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=1  FILL F3 END ARRAY
READ BOUNDS  XFC=MIRROR  END BOUNDS
```

In the above description, the first HOLE description places the lower annulus at its proper location. The second HOLE description places the upper annulus at its proper position and the third HOLE description places the middle annulus at its correct position.

In each of the preceding models, the unit containing the holes could be specified as the global unit and the array data could be omitted.

F11.5.6.2 *Nesting Holes*

This section illustrates how holes are nested. Holes can be nested to any level. Consider the configuration that was illustrated in Fig. F11.5.6 and replace the large rods with a complicated geometric arrangement. The resultant figure is shown in Fig. F11.5.8. Figure F11.5.9 shows the complicated geometric arrangement that replaced the large rods of Fig. F11.5.6. Figure F11.5.10 shows a component of the arrangement shown in Fig. F11.5.9.

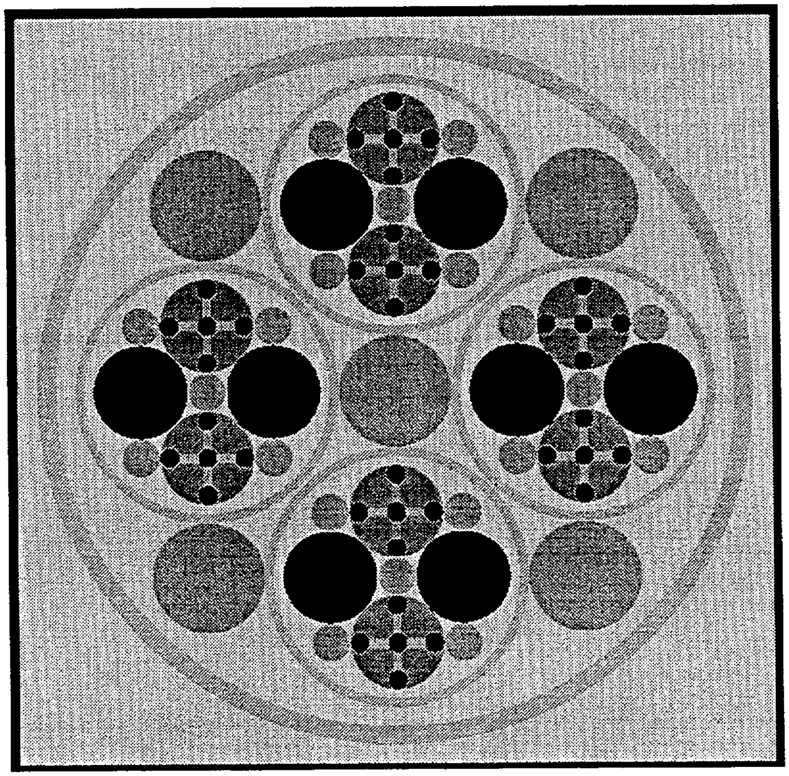


Figure F11.5.8 Configuration using nested holes

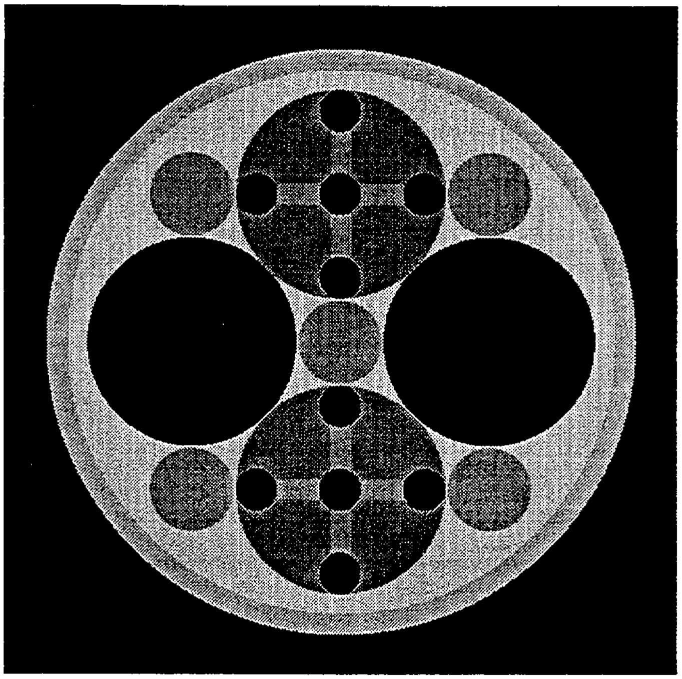


Figure F11.5.9 Complicated geometric arrangement represented by Unit 7

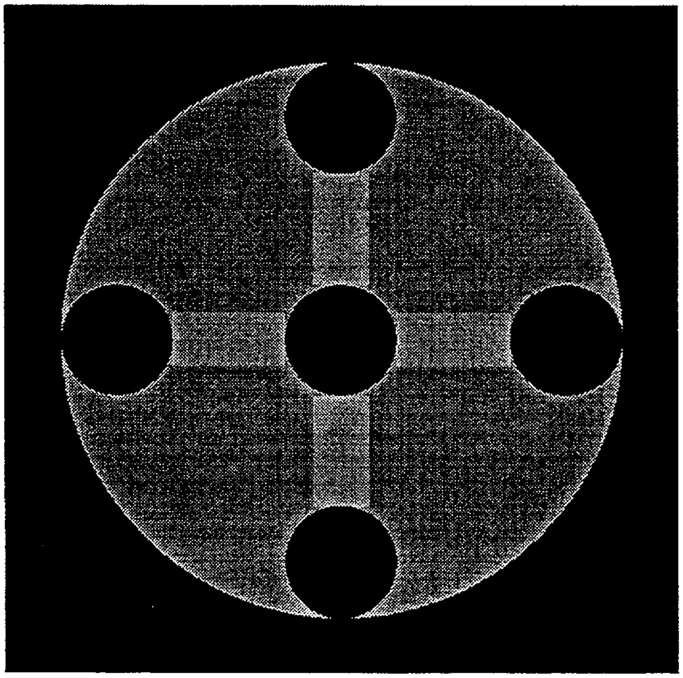


Figure F11.5.10 Geometric component represented by Unit 4

No predetermined way is the "best way" to create a geometry mock-up for a given physical system. The user should decide the order that is most convenient. In order to describe the configuration using nested holes, Fig. F11.5.8, it may be most convenient to start the geometry mock-up at the deepest nesting level as shown in Fig. F11.5.10. The small cylinders are composed of mixture 1, they are each 0.1 cm in radius and 30 cm long. There are five small cylinders used in Fig. F11.5.10. Their centers are located at (0,0,0) for the central one, at (0,-0.4,0) for the bottom one, at (0.4,0,0) for the right one, at (0,0.4,0) for the top one, and at (-0.4,0,0) for the left one. The rectangular parallelepipeds (cuboids) are composed of mixture 2. Each one is 30 cm long and 0.1 cm by 0.2 cm in cross section. The large cylinder containing the configuration is composed of mixture 3, is 30 cm long and has a radius of 0.5 cm. The geometry mock-up for this system is described as follows:

(1) define a small cylinder to be Unit 1,

(2) define a small cuboid with its length in the x direction to be Unit 2,

(3) define a small cuboid with its length in the y direction to be Unit 3,

(4) define Unit 4 to be the large cylinder and place the cylinders and cuboids in it using holes.

```
UNIT 1
CYLINDER 1 1 0.1 2P15.0
UNIT 2
CUBOID 2 1 2P0.1 2P0.05 2P15.0
UNIT 3
CUBOID 2 1 2P0.05 2P0.1 2P15.0
UNIT 4
CYLINDER  1 1 0.1 2P15.0
CYLINDER  3 1 0.5 2P15.0
HOLE 1 0.0 -0.4 0.0
HOLE 1 0.4 0.0 0.0
HOLE 1 0.0 0.4 0.0
HOLE 1 -0.4 0.0 0.0
HOLE 2 -0.2 0.0 0.0
HOLE 2 0.2 0.0 0.0
HOLE 3 0.0 -0.2 0.0
HOLE 3 0.0 0.2 0.0
```

The first cylinder description in Unit 4 places the central rod;
the **first** HOLE places the bottom cylinder;
the **second** HOLE places the cylinder at the right;
the **third** HOLE places the top cylinder;
the **fourth** HOLE places the cylinder at the left;
the **fifth** HOLE places the left cuboid whose length is in x;
the **sixth** HOLE places the right cuboid whose length is in x;
the **seventh** HOLE places the bottom cuboid whose length is in y; and
the **eighth** HOLE places the top cuboid whose length is in y.

Now that Fig. F11.5.10 has been described, consider Fig. F11.5.9. The large plain cylinders are composed of mixture 1 and are 0.5 cm in radius and 30 cm long. The cylindrical component of Unit 4 is the same size, an outer radius of 0.5 cm and a length of 30 cm. The small cylinders that are located in the interstices between the large cylinders are composed of mixture 2, are 0.2 cm in radius, and are 30 cm long. Define Unit 5 to be the large plain cylinder and Unit 6 to be the small cylinder; Unit 7 is the annulus that contains the cylinders. Its origin is at its center. The annulus is composed of mixture 4, has a 1.3-cm inside radius and a 1.4-cm outer radius. The volume between the inner cylinders is void. The large cylinders each have a radius of 0.5 cm and are tangent. Therefore, their origins are offset from the origin of the unit by 0.707107. This is from $x^2 + y^2 = 1.0$ (radius unit 4 + radius unit 5), where x and y are equal. The geometry mock-up for this portion of the problem follows:

```
UNIT 5
CYLINDER  1 1 0.5 2P15.0
UNIT 6
CYLINDER  2 1 0.2 2P15.0
UNIT 7
CYLINDER  2 1 0.2 2P15.0
CYLINDER  0 1 1.3 2P15.0
HOLE      5 0.707107 0.0 0.0
HOLE      6 0.707107 0.707107 0.0
HOLE      4 0.0 0.707107 0.0
HOLE      6 -0.707107 0.707107 0.0
HOLE      5 -0.707107 0.0 0.0
HOLE      6 -0.707107 -0.707107 0.0
HOLE      4 0.0 -0.707107 0.0
HOLE      6 0.707107 -0.707107 0.0
CYLINDER  4 1 1.4 2P15.0
```

In Unit 7, the first cylinder description defines the central rod and the second cylinder defines the void volume in which the other cylinders are to be placed.

The **first** HOLE places the larger cylinder of mixture 1 at the right with its origin at (0.707107,0.0,0.0),
the **second** HOLE places the small cylinder in the upper-right quadrant,
the **third** HOLE places the top cylinder that contains the geometric component defined in Unit 4,
the **fourth** HOLE places the small cylinder in the upper-left quadrant,
the **fifth** HOLE places the larger cylinder of mixture 1 at the left,
the **sixth** HOLE places the small cylinder in the lower-left quadrant,
the **seventh** HOLE places the bottom cylinder that contains the geometric component defined in Unit 4,
and the **eighth** HOLE places the small cylinder in the lower-right quadrant.

The last cylinder defines the outer surface of the annulus.

To complete the geometry mock-up, consider Fig. F11.5.8.

Define Unit 8 to be a cylinder of mixture 2 having a radius of 0.6 cm and a length of 30 cm.

Define Unit 9 to be the central rod and the large annulus centered in a cuboid having an 8-cm-square cross section and being 32 cm long.

```
UNIT 8
CYLINDER  2 1 0.6 2P15.0
UNIT 9
CYLINDER  2 1 0.6 2P15.0
CYLINDER  0 1 3.6 2P15
HOLE   7 2.0 0.0 0.0
HOLE   8 2*2.0 0.0
HOLE   7 0.0 2.0 0.0
HOLE   8 -2.0 2.0 0.0
HOLE   7 -2.0 2*0.0
HOLE   8 2*-2.0 0.0
HOLE   7 0.0 -2.0 0.0
HOLE   8 2P2.0 0.0
CYLINDER  4 1 3.8 2P15.0
CUBOID    0 1 4P4.0 2P16.0
```

In Unit 9, the first cylinder defines the rod of mixture 2, centered in the annulus. The second cylinder defines the void volume between the central rod and the annulus.

The **first** HOLE places the composite annulus of Unit 7 to the right of the central rod,
the **second** HOLE places a rod defined by Unit 8 in the upper right quadrant of the annulus,
the **third** HOLE places the composite annulus of Unit 7 above the central rod,
the **fourth** HOLE places a rod defined by Unit 8 in the upper left quadrant of the annulus,
the **fifth** HOLE places the composite annulus of Unit 7 to the left of the central rod,
the **sixth** HOLE places a rod defined by Unit 8 in the lower left quadrant,
the **seventh** HOLE places the composite annulus of Unit 7 below the central rod,
and the **eighth** HOLE places a rod defined by Unit 8 in the lower right quadrant.

The last cylinder defines the outer surface of the annulus. The outer cuboid is the last region.

This problem illustrates three levels of hole nesting. The total input data for the problem is given below. The nuclide IDs are for the 16-group Hansen-Roach working format library. The mixtures used in this problem are not realistic or meaningful. However, the geometry description accurately recreates the geometry arrangement of Fig. F11.5.8. This problem includes the data for a character plot to be used to verify the validity of the geometry description. The plot data specify a character plot that is 260 characters wide, so the plot is generated in two pieces. The left half of the character plot is shown in Fig. F11.5.11, and the right half is given in Fig. F11.5.12. The user can tape the two halves together. If the plot were specified to be 130 characters wide, it would all print in one piece. However, some of the detail might have been lost.

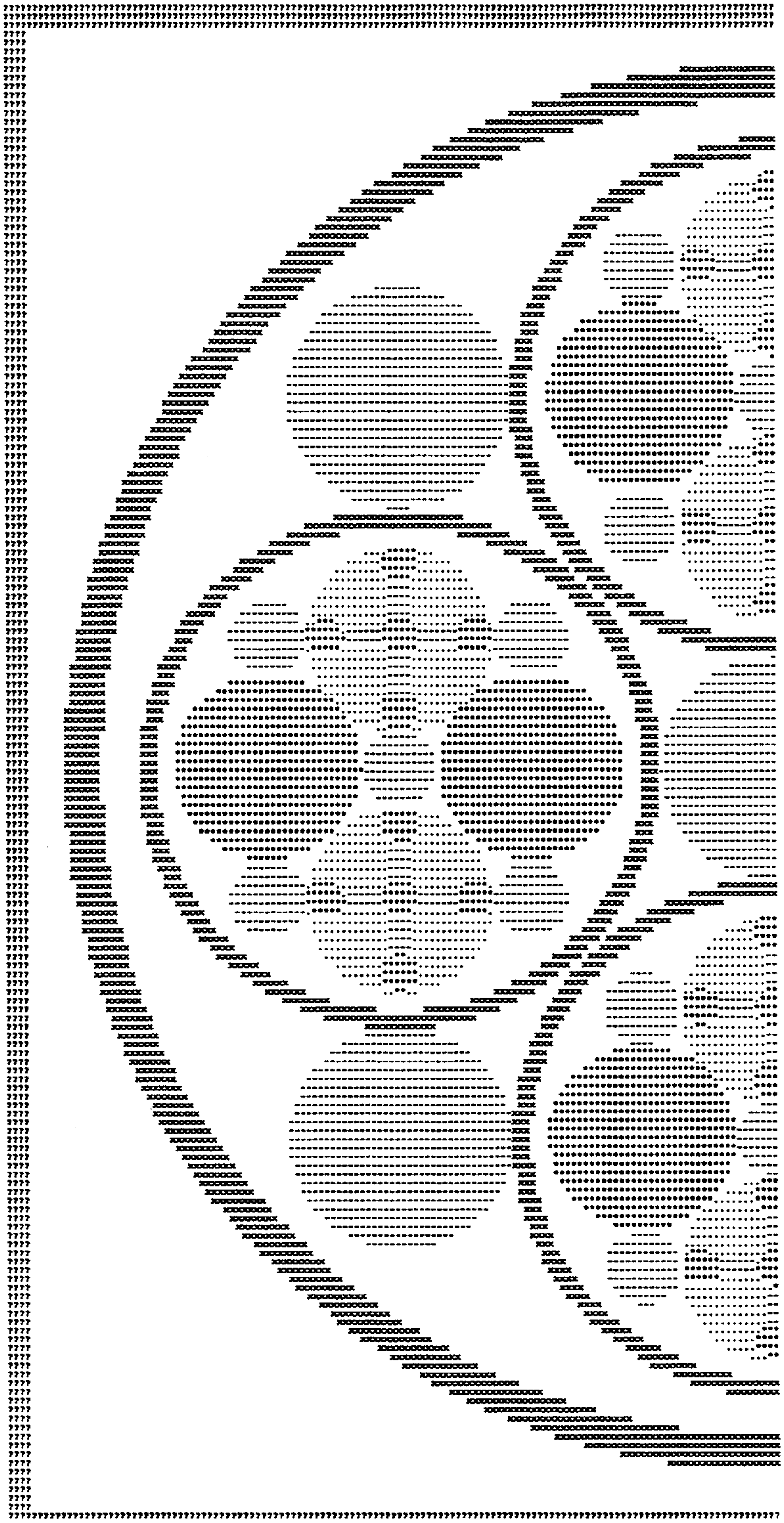


Figure F11.5.11a Left half of printer plot

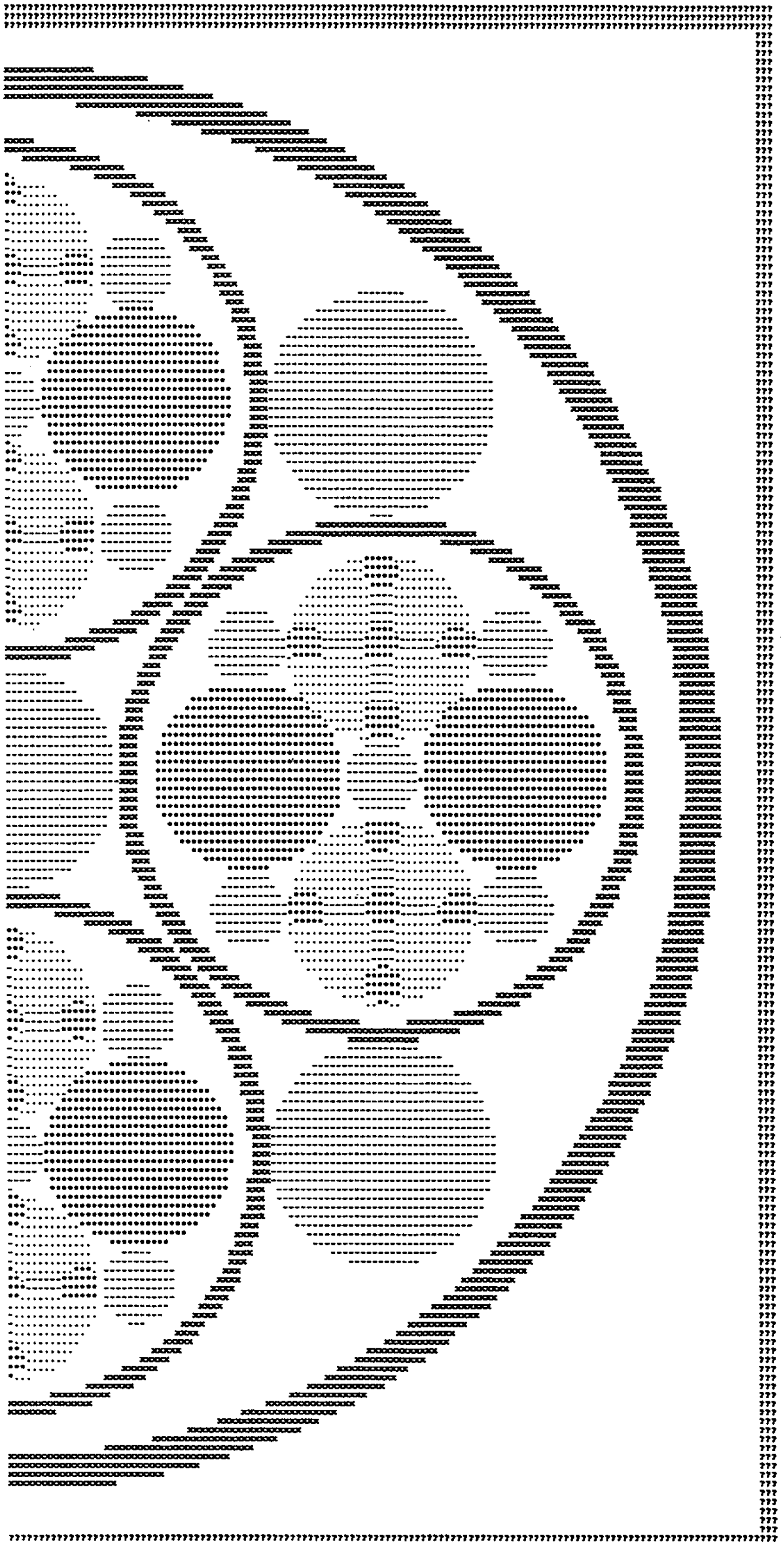


Figure F11.5.11b Right half of printer plot

```
NESTED HOLES SAMPLE
READ PARAM  RUN=NO LIB=41 END PARAM
READ MIXT  SCT=1  MIX=1 92500 4.7048-2 MIX=2 200 1.0 MIX=3 502 0.1
MIX=4 200 1.0
END MIXT
READ GEOM
UNIT 1
CYLINDER 1 1 0.1 2P15.0
UNIT 2
CUBOID 2 1 2P0.1 2P0.05 2P15.0
UNIT 3
CUBOID 2 1 2P0.05 2P0.1 2P15.0
UNIT 4
CYLINDER 1 1 0.1 2P15.0
CYLINDER 3 1 0.5 2P15.0
HOLE 1  0.0 -0.4 0.0
HOLE 1  0.4  0.0 0.0
HOLE 1  0.0  0.4 0.0
HOLE 1 -0.4  0.0 0.0
HOLE 2 -0.2  0.0 0.0
HOLE 2  0.2  0.0 0.0
HOLE 3  0.0 -0.2 0.0
HOLE 3  0.0  0.2 0.0
UNIT 5
CYLINDER 1 1 0.5 2P15.0
UNIT 6
CYLINDER 2 1 0.2 2P15.0
UNIT 7
CYLINDER 2 1 0.2 2P15.0
CYLINDER 0 1 1.3 2P15.0
HOLE 5 0.707107 2*0.0
HOLE 6 0.707107 0.707107 0.0
HOLE 4 0.0 0.707107 0.0
HOLE 6 -0.707107 0.707107 0.0
HOLE 5 -0.707107 0.0 0.0
HOLE 6 -0.707107 -0.707107 0.0
HOLE 4 0.0 -0.707107 0.0
HOLE 6 0.707107 -0.707107 0.0
CYLINDER 4 1 1.4 2P15.0
UNIT 8
CYLINDER 2 1 0.6 2P15.0
UNIT 9
CYLINDER 2 1 0.6 2P15.0
CYLINDER 0 1 3.6 2P15.0
HOLE 7 2.0 0.0 0.0
HOLE 8 2*2.0 0.0
HOLE 7 0.0 2.0 0.0
HOLE 8 -2.0 2.0 0.0
HOLE 7 -2.0 2*0.0
HOLE 8 2*-2.0 0.0
HOLE 7 0.0 -2.0 0.0
HOLE 8 2P2.0 0.0
CYLINDER 4 1 3.8 2P15.0
CUBOID 0 1 4P4.0 2P16.0
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=1  FILL 9 END ARRAY
READ PLOT  TTL='X-Y SLICE AT Z MIDPOINT.  NESTED HOLES'
XUL=-0.1  YUL=8.1 ZUL=16.0 XLR=8.1 YLR=-0.1 ZLR=16
UAX=1.0 VDN=-1.0  NAX=260 NCH=' *-.X' SCR=NO END PLOT
END DATA
END
```

F11.5.6.3 *Multiple Arrays*

Section F11.5.6 demonstrates how units are composed of geometry regions and how these units can be stacked in an array. This same procedure can be extended to create multiple arrays. Furthermore, arrays can be used as building blocks within other arrays.

Consider Sample Problem 12 from Sect. F11.D. The description of this sample problem is restated below as Sample Problem 19.

This problem is a critical experiment consisting of a composite array[2,3] of four highly enriched uranium metal cylinders and four cylindrical Plexiglas containers filled with uranyl nitrate solution. The metal units in this experiment are designated in Table II of ref. 1 as cylinder index 11 and reflector index 1. A photograph of the experiment is given in Fig. F11.D.3. The coordinate system is defined to be z up the page, y across the page, and x out of the page.

The Plexiglas containers have an inside radius of 9.525 cm and an outside radius of 10.16 cm. The inside height is 17.78 cm, and the outside height is 19.05 cm. Four of these containers are stacked with a center-to-center spacing of 21.75 cm in the "y" direction and 20.48 cm in the "z" direction (vertical). This arrangement of four Plexiglas containers can be described as follows: mixture 2 is the uranyl nitrate and mixture 3 is Plexiglas, so the Plexiglas container with its appropriate spacing cuboid can be described as Unit 1. This considers the array to be bare and suspended with no supports.

```
UNIT 1
CYLINDER  2 1 9.525  2P8.89
CYLINDER  3 1 10.16  2P9.525
CUBOID    0 1 4P10.875  2P10.24
```

The array of four Plexiglas containers can be described as array 1 in the array data as follows:

```
ARA=1  NUX=1  NUY=2  NUZ=2  FILL  F1  END FILL
```

The four metal cylinders each have a radius of 5.748 cm and are 10.765 cm tall. They have a center-to-center spacing of 13.18 cm in the "y" direction and 12.45 cm in the "z" direction (vertical). Thus, one of the metal cylinders with its appropriate spacing cuboid can be described as Unit 2. This array is also considered to be bare and unsupported.

```
UNIT 2
CYLINDER 1 1  5.748 2P5.3825
CUBOID   0 1  4P6.59 2P6.225
```

The array of four metal cylinders can be described as array 2 in the array data.

```
ARA=2  NUX=1  NUY=2  NUZ=2  FILL  F2  END FILL
```

Now two arrays have been described. The overall dimensions of the array of Plexiglas containers is 21.75 cm in x, 43.5 cm in y, and 40.96 cm in z. The overall dimensions of the array of metal cylinders is 13.18 cm in x, 26.36 cm in y, and 24.9 cm in z.

In order to describe the composite array, these two arrays must be stacked together into an array. In order for them to be stacked into an array, the adjacent faces must match. This is accomplished by defining a Unit 3 which contains array 1, the array of Plexiglas solution containers. The overall dimensions of this unit are 21.7 cm in x, 43.5 cm in y, and 40.96 cm in z. These dimensions are calculated by the code and need not be specified. Unit 3 is defined as follows:

```
UNIT 3
ARRAY 1  3*0.0
```

The array of metal cylinders will be defined to be Unit 4. However, this array is 17.14 cm smaller in the y and 16.06 cm smaller in the z dimensions than the array of Plexiglas units. Therefore, a void region must be placed around the array in those directions so Unit 4 and Unit 3 will be the same size in y and z. This can be accomplished by using a reflector card as follows:

```
UNIT 4
ARRAY  2  3*0.0
REPLICATE  0 1 2*0.0  2*8.57  2*8.03  1
```

Now that Unit 3 and Unit 4 have been defined and their y-z faces are the same size, they must be placed in the global or universe array to define the physical arrangement of the eight pieces. This procedure is done in the array data and the global array number is set to 3 as follows:

```
GBL=3  ARA=3  NUX=2  NUY=1  NUZ=1  FILL  4 3  END FILL
```

This completes the geometry description for the problem. The complete input description for the problem is given below. The nuclide IDs are for the 16-group Hansen-Roach working format library.

```
=KENOV
SAMPLE PROBLEM 19 4 AQUEOUS 4 METAL ARRAY OF ARRAYS
READ PARAM  LIB=41  RUN=NO END PARAM
READ MIXT SCT=1  MIX=1  92860 3.2275-3  92501 4.4802-2  MIX=2  1102 5.81-2
7100 1.9753-3  8100 3.6927-2  92501 9.8471-4  92860 7.7697-5
MIX=3  6100 3.5552-2  1102 5.6884-2  8100 1.4221-2  END MIXT
READ GEOM
UNIT 1
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 2P9.525
CUBOID  0 1 4P10.875 2P10.24
UNIT 2
CYLINDER  1 1 5.748 2P5.3825
CUBOID  0 1 4P6.59 2P6.225
UNIT 3
ARRAY 1 3*0.0
UNIT 4
ARRAY 2 3*0.0
REPLICATE 0 1 2*0.0 2*8.57 2*8.03 1
END GEOM
READ ARRAY  ARA=1 NUX=1 NUY=2 NUZ=2  FILL F1 END FILL
ARA=2 NUX=1 NUY=2 NUZ=2 FILL F2 END FILL GBL=3 ARA=3 NUX=2 NUY=1 NUZ=1
```

```
FILL 4 3 END FILL
END ARRAY
READ PLOT TTL='X-Y SLICE AT Z=10.24'
XUL=-1.0 YUL=44.5 ZUL=10.24 XLR=35.93 YLR=-1.0 ZLR=10.24
UAX=1.0 VDN=-1.0 NAX=640 PIC=MIX END
TTL='X-Z SLICE AT Y=10.875'
XUL=-1.0 YUL=10.875 ZUL=41.96 XLR=35.93 YLR=10.875 ZLR=-1.0
UAX=1.0 WDN=-1.0 PIC=MIX END  END PLOT
END DATA
END
```

A color plot of an x-y slice taken through the bottom layer of the array is shown in Fig. F11.5.12. A color plot of an x-z slice taken through the left half of the array is shown in Fig. F11.5.13. These color plots were used to verify the geometry mockup.

STORAGE ARRAY

Consider a storage array of highly enriched uranium buttons, each 1 in. tall and 4 in. in diameter. These buttons are stored on stainless steel shelves with a center-to-center spacing of 2 ft between them. The shelves are 1/4 in. thick, 18 in. wide, 20 ft long, and are 18 in. from the top of a shelf to the bottom of the shelf above it. Each rack of storage shelves is four shelves high, with the first shelf being 6 in. above the floor. The storage room is 19.5 ft in the x direction by 43 ft in the y direction with 12-ft ceilings in the z direction. The walls, ceiling, and floor are composed of concrete, 1 ft thick. All the aisles between the storage racks are 3 ft wide. The racks are arranged with their length in the y direction and an aisle between them. The array of racks are arranged with two in the y direction and five in the x direction. Mixture 1 is the uranium metal, mixture 2 is the stainless steel, and mixture 3 is the concrete.

First, describe the metal button and its center-to-center spacing. The void vertical spacing has arbitrarily been chosen to extend from the bottom of the button to the next shelf above the button. The shelf of stainless steel is described under the button.

```
UNIT 1
COM='METAL BUTTONS'
CYLINDER 1 1 5.08 2.54 0.0
CUBOID  0 1 2P22.86 2P30.48 45.72 0.0
CUBOID  2 1 2P22.86 2P30.48 45.72 -0.635
```

Array 1 creates an array of these buttons that fills one shelf. Unit 2 then contains one of the shelves shown in Fig. F11.5.14.

```
ARA=1 COM='SINGLE SHELF CONTAINING 10 METAL BUTTONS'
    NUX=1 NUY=10 NUZ=1 FILL F1 END FILL
```

```
UNIT 2
COM='SINGLE SHELF (1 X 10 X 1 ARRAY OF METAL BUTTONS ON A SHELF)'
ARRAY 1 3*0.0
```

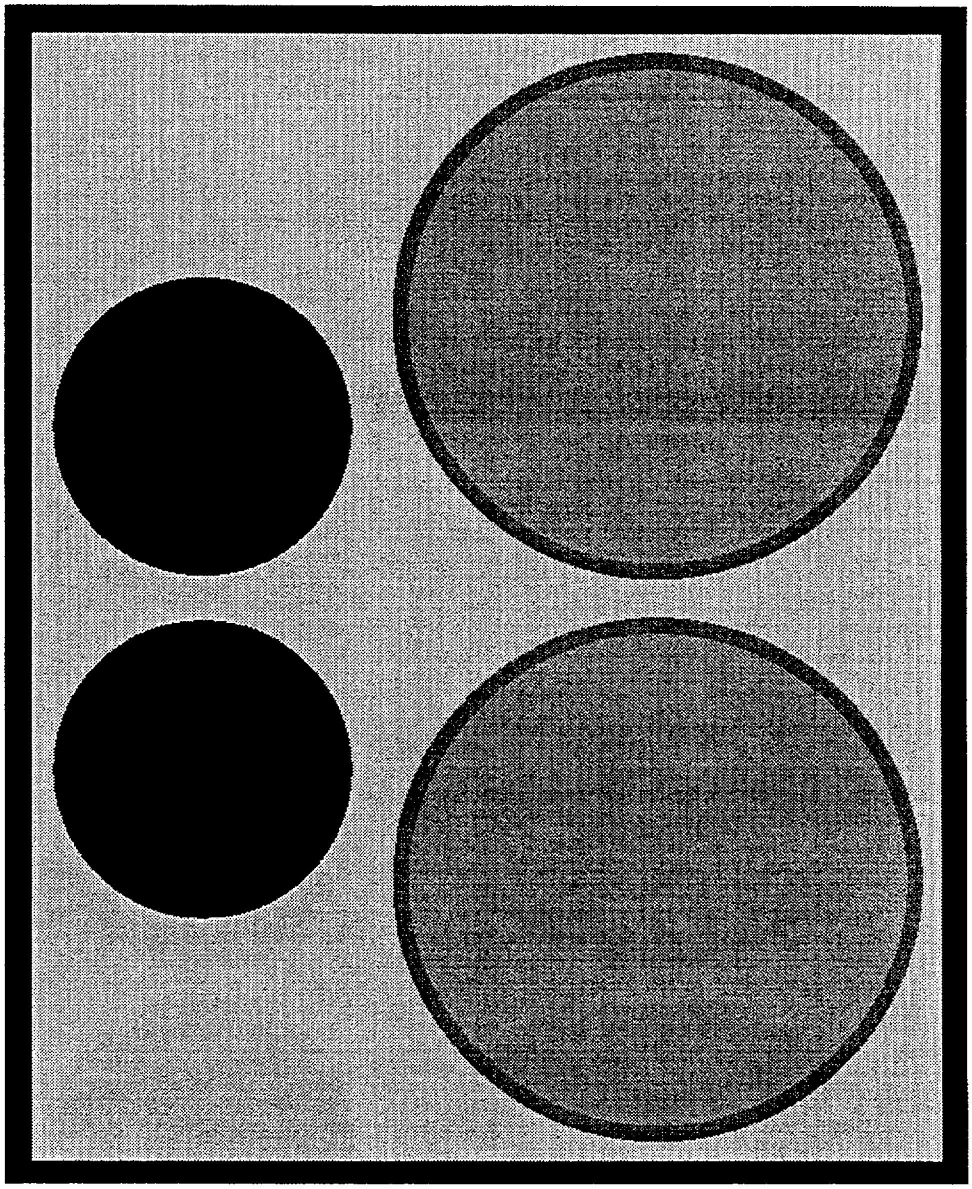


Figure F11.5.12 x-y plot of mixed array

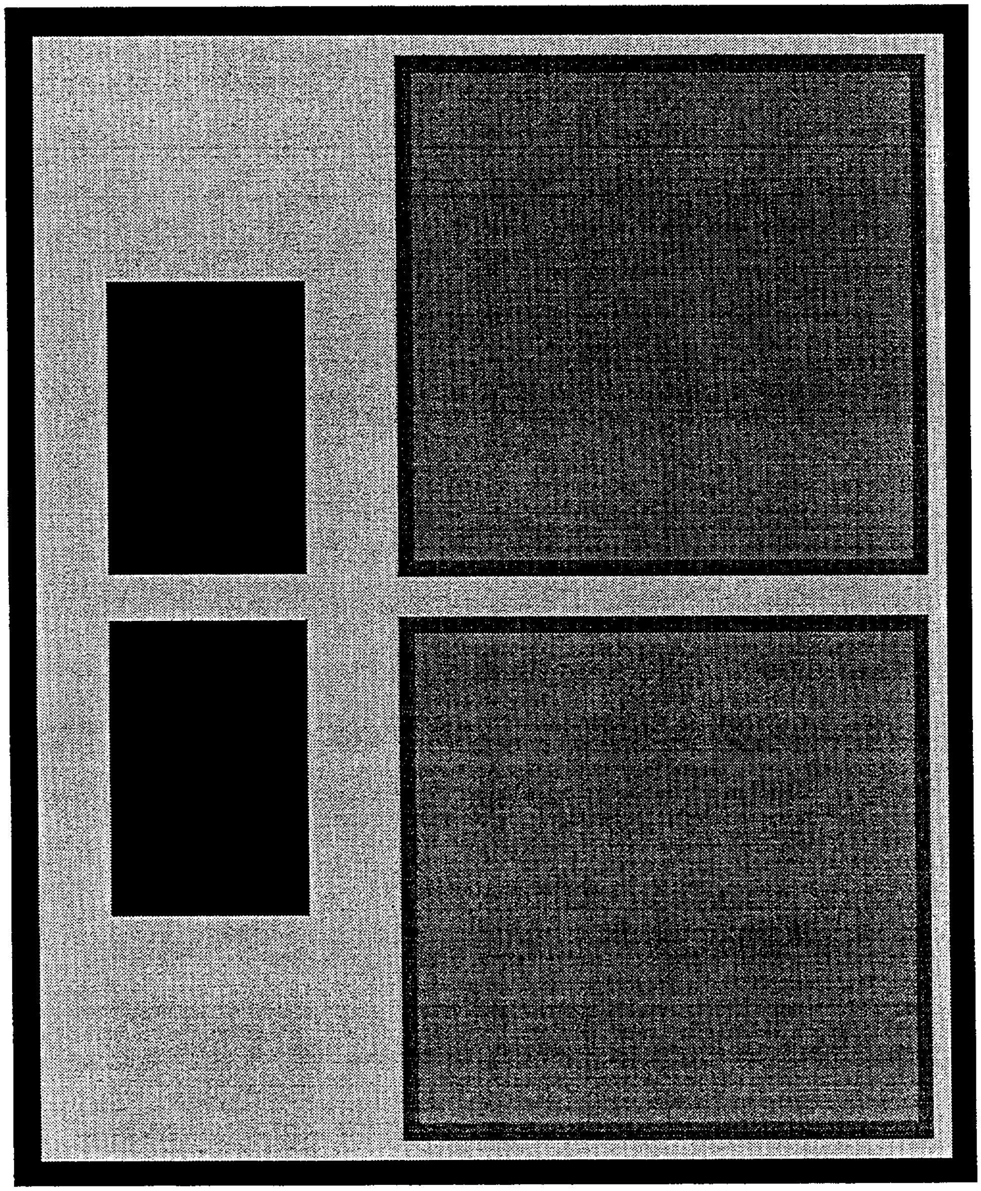


Figure F11.5.13 x-z plot of mixed array

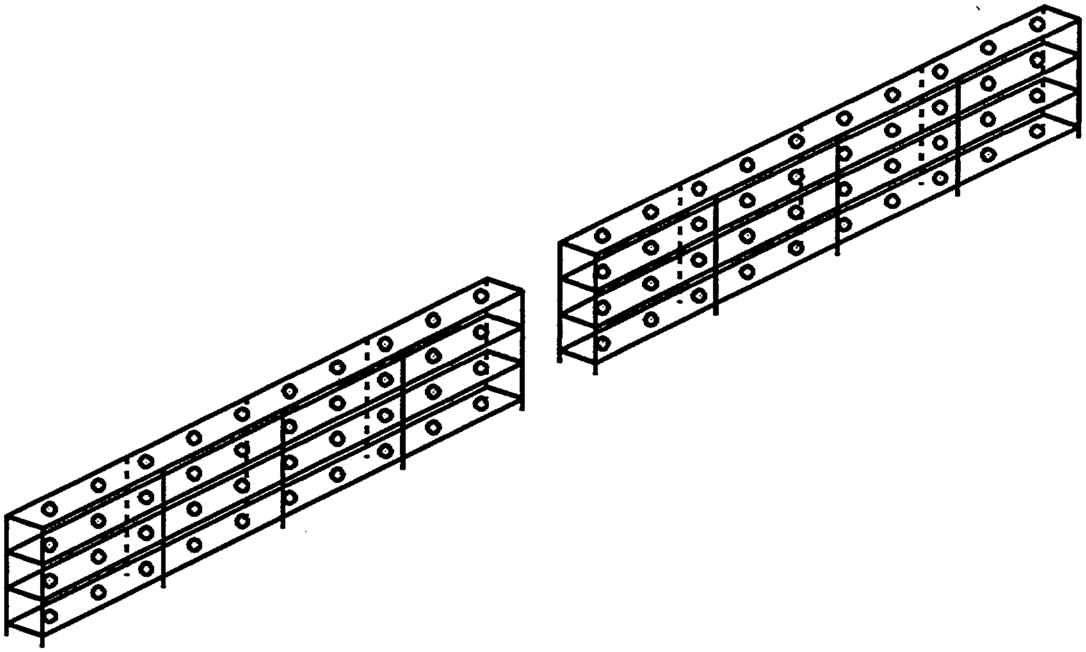


Figure F11.5.14 Two racks of uranium buttons

Stack four Unit 2's vertically to obtain one of the racks shown in Fig. F11.5.14. One rack is defined by array 2.

```
ARA=2 COM='SINGLE RACK OF 4 SHELVES'
   NUX=1 NUY=1 NUZ=4 FILL F2 END FILL
```

Generate a Unit 3 that contains a rack of shelves and a Unit 4 that is the aisle between the ends of the two racks in the y direction.

```
UNIT 3
COM='SINGLE RACK (4 SHELVES TALL)'
ARRAY 2 3*0.0
```

```
UNIT 4
COM='CENTRAL AISLE UNIT SAME HEIGHT AS 4 SHELVES'
CUBOID 0 1 2P22.86 2P45.72 185.42 0.0
```

Stack Units 3 and 4 together in the y direction to create Unit 5 which contains both racks in the y direction and the aisle between them. This configuration is shown in Fig. F11.5.14.

```
ARA=3 COM='TWO RACKS END TO END WITH CENTRAL AISLE'
   NUX=1 NUY=3 NUZ=1 FILL 3 4 3 END FILL
```

```
UNIT 5
COM='SET OF TWO RACKS END TO END SEPARATED BY THE CENTRAL AISLE'
ARRAY 3 3*0.0
```

Create a Unit 6 which is an aisle 3 ft wide in the x direction and 43 ft in the y direction (full length of the room).

```
UNIT 6
COM='AISLE BETWEEN ADJACENT SETS OF TWO RACKS & CENTRAL AISLE (UNITS 5)'
 CUBOID 0 1 91.44 0.0 1310.64 0.0 185.42 0.0
```

Stack Units 5 and 6 in the x direction to achieve the array of racks in the room. Then put the 6-in. spacing below the bottom of the racks, the spacing between the top of the top rack and the ceiling, and add the concrete floor, walls, and ceiling around the array. Array 4 describes the array of racks in the room. The core description encompasses this array, and the first reflector descriptions are used to add the spacing between the top rack and the ceiling. The last two reflector descriptions add the ceiling, walls and floor. A perspective of the room is shown in Fig. F11.5.15.

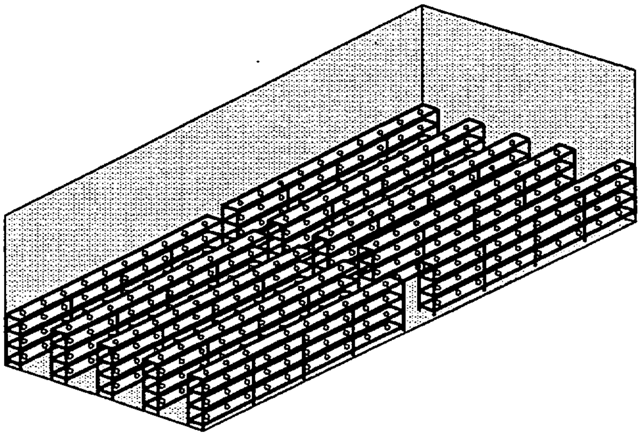


Figure F11.5.15 Entire storage array in the room

```
ARA=4 COM=’ENTIRE STORAGE ARRAY’
    NUX=9 NUY=1 NUZ=1 FILL 5 6 3Q2 5 END FILL
GLOBAL
UNIT 7
COM=’STORAGE ARRAY IN THE ROOM WITH WALLS, FLOOR AND CEILING’
4 3*0.0
0 1 4*0.0 165.1 15.24 1
REFLECTOR 3 2 6*5.0 6
REFLECTOR 3 8 6*0.48 1
```

The final mockup for this room is given below: The plots for this problem must be quite large in order to see all the detail because the array is sparse and the shelves are thin. Therefore, the plots for this system are not included as figures. The user can generate the plots if it is desirable to see them. The nuclide IDs used in this problem are for the 16-group Hansen-Roach working format library.

```
=KENO5
STORAGE ARRAY
READ PARAMETERS  FDN=YES LIB=41
END PARAMETERS
READ MIXT  SCT=1  MIX=1 92500 4.48006-2  92800 2.6578-3  92400 4.827-4
92600 9.57-5  MIX=2 200 1.0 MIX=3 301 1 END MIXT
READ GEOMETRY
UNIT 1
COM=’METAL BUTTONS’
CYLINDER 1 1 5.08 2.54 0.0
CUBOID  0 1 2P22.86 2P30.48 45.72 0.0
CUBOID  2 1 2P22.86 2P30.48 45.72 -0.635
UNIT 2
COM=’SINGLE SHELF (1 X 10 X 1 ARRAY OF METAL BUTTONS ON A SHELF)’
ARRAY 1 3*0.0
UNIT 3
COM=’SINGLE RACK (4 SHELVES TALL)’
ARRAY 2 3*0.0
UNIT 4
COM=’CENTRAL AISLE UNIT SAME HEIGHT AS 4 SHELVES’
CUBOID 0 1 2P22.86 2P45.72 185.42 0.0
UNIT 5
COM=’SET OF TWO RACKS END TO END SEPARATED BY THE CENTRAL AISLE’
ARRAY 3 3*0.0
UNIT 6
COM=’AISLE BETWEEN ADJACENT SETS OF TWO RACKS & CENTRAL AISLE (UNITS 5)’
CUBOID 0 1 91.44 0.0 1310.64 0.0 185.42 0.0
GLOBAL
```

```
UNIT 7
COM='STORAGE ARRAY IN THE ROOM WITH WALLS, FLOOR AND CEILING'
ARRAY 4 3*0.0
REFLECTOR 0 1 4*0.0 165.1 15.24 1
REFLECTOR 3 2 6*5.0 6
REFLECTOR 3 8 6*0.48 1
END GEOMETRY
READ ARRAY
ARA=1 COM='SINGLE SHELF CONTAINING 10 METAL BUTTONS'
   NUX=1 NUY=10 NUZ=1 FILL F1 END FILL
ARA=2 COM='SINGLE RACK OF 4 SHELVES'
   NUX=1 NUY=1 NUZ=4 FILL F2 END FILL
ARA=3 COM='TWO RACKS END TO END WITH CENTRAL AISLE'
   NUX=1 NUY=3 NUZ=1 FILL 3 4 3 END FILL
ARA=4 COM='ENTIRE STORAGE ARRAY'
   NUX=9 NUY=1 NUZ=1 FILL 5 6 3Q2 5 END FILL
END ARRAY
READ BIAS ID=301 2 8 END BIAS
READ START NST=5  NBX=5 END START
READ PLOT TTL='X-Z SLICE AT Y=30.48 WITH Z ACROSS AND X DOWN'
XUL=594.8 YUL=30.48 ZUL=-1.0 XLR=-0.5 YLR=30.48 ZLR=186.0
WAX=1.0 UDN=-1.0 NAX=640  END
TTL='Y-Z SLICE OF LEFT RACKS, X=22.86 WITH Z ACROSS AND Y DOWN'
XUL=22.86 YUL=1311.0 ZUL=-0.5 XLR=22.86 YLR=-3.0 ZLR=186.0
WAX=1.0 VDN=-1.0 NAX=640 END
TTL='X-Y SLICE OF ROOM THROUGH SHELF Z=0.3175 WITH X ACROSS AND Y DOWN'
XUL=-1.0 YUL=1312.0 ZUL=0.3175 XLR=596.0 YLR=-2.5 ZLR=0.3175
UAX=1.0 VDN=-1.0 NAX=320 END
END PLOT
END DATA
END
```

F11.5.6.4 *Arrays and Holes*

Sections F11.5.6.1 and 2 describe the use of holes and Sect. F11.5.6.3 describes multiple arrays and arrays of arrays. Holes can be used to place arrays at locations that would have been impossible with earlier geometry restrictions. This section contains examples to illustrate the combined use of arrays and holes.

EXAMPLE 1. A SIMPLE CASK

Consider a cylindrical mild steel container having an inside radius of 4.15 cm and a radial wall thickness of 0.45 cm. The thickness of the ends of the container is 1.27 cm, and the inside height is 10.1 cm. Highly enriched uranium rods 1 cm in diameter and 10 cm long are banded together into square bundles of four. These bundles are then positioned in the mild steel container as shown in Fig. F11.5.16. The rods sit on the floor of the container and have a 0.1-cm gap between their tops and the top of the container.

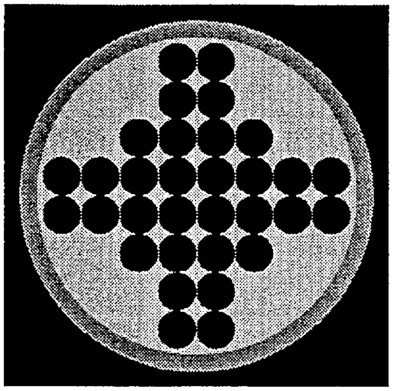


Figure F11.5.16 Uranium rods in a cylindrical container

To generate the geometry description for this system, define Unit 1 to be one uranium rod and its associated square-pitch close-packed spacing region.

```
UNIT 1
CYLINDER  1 1  0.5  2P5.0
CUBOID    0 1  4P0.5  2P5.0
```

Define array 1 to be the central square array consisting of four bundles of rods.

```
ARA=1  NUX=4  NUY=4  NUZ=1  FILL  F1  END FILL
```

Define array 2 to be a bundle of four rods.

```
ARA=2  NUX=2  NUY=2  NUZ=1  FILL  F1  END FILL
```

Now place array 2 in Unit 2. This defines the outer boundaries of an imaginary cuboid that contains the array. It is convenient to have the origin of the array at its center, so the most negative point of the array will be (-1,-1,-5).

```
UNIT 2
ARRAY 2  -1.0 -1.0 -5.0
```

A core description is used to place array 1 in the global unit. Then the cylindrical container is described around it and holes are used to place the four outer bundles around the central array.

```
CORE  1 1  -2.0  -2.0  -5.0
CYLINDER   0 1  4.15  5.1  -5.0
HOLE  2    0.0 -3.0  0.0
HOLE  2    3.0  0.0  0.0
HOLE  2    0.0  3.0  0.0
HOLE  2   -3.0  0.0  0.0
CYLINDER  2  1  4.6  6.37 -6.27
```

The first hole places the bottom bundle of four rods, the second hole places the bundle of four rods at the right, the third hole places to top bundle of rods and the fourth hole places the left bundle of rods.

The overall problem description is shown below. Two of the color plots used for verification of this mock-up are shown in Figs. F11.5.17 and F11.5.18.

```
=KENOV
CASK ARRAY
READ PARAMETERS  FDN=YES LIB=41 GEN=10
END PARAMETERS
READ MIXT  SCT=1  MIX=1 92500 4.48006-2  92800 2.6578-3  92400 4.827-4
92600 9.57-5  MIX=2 100 1.0 END MIXT
READ GEOMETRY
UNIT 1
CYLINDER 1 1 0.5 2P5.0
CUBOID  0 1 4P0.5 2P5.0
UNIT 2
ARRAY 2 -1.0 -1.0 -5.0
CORE 1 1 -2.0 -2.0 -5.0
CYLINDER 0 1 4.15 5.1 -5.0
HOLE 2 0.0 -3.0 0.0
HOLE 2 3.0 0.0 0.0
HOLE 2 0.0 3.0 0.0
HOLE 2 -3.0 0.0 0.0
CYLINDER 2 1 4.6 6.37 -6.27
END GEOM
READ ARRAY
ARA=1 NUX=4 NUY=4 NUZ=1 FILL F1 END FILL
ARA=2 NUX=2 NUY=2 NUZ=1 FILL F1 END FILL
END ARRAY
READ PLOT TTL='X-Z SLICE AT Y=0.25 WITH X ACROSS AND Z DOWN'
XUL=-5.0 YUL=0.25 ZUL=6.5 XLR=5.0 YLR=0.25 ZLR=-6.5
UAX=1.0 WDN=-1.0 NAX=640  END
TTL='X-Y SLICE AT Z=0.0 WITH X ACROSS AND Y DOWN'
XUL=-5.0 YUL=5.0 ZUL=0.0 XLR=5.0 YLR=-5.0 ZLR=0.0
UAX=1.0 VDN=-1.0 NAX=640 END
END PLOT
END DATA
END
```

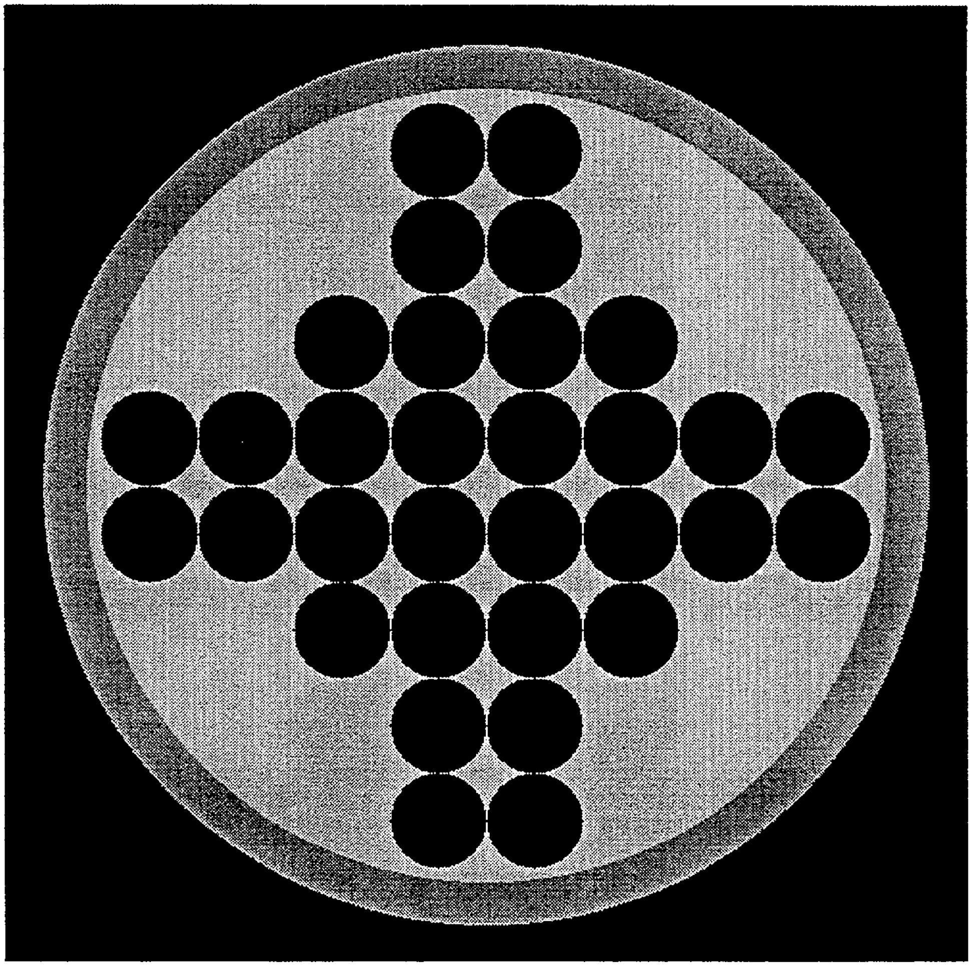


Figure F11.5.17 x-y slice of uranium rods in a cylindrical container

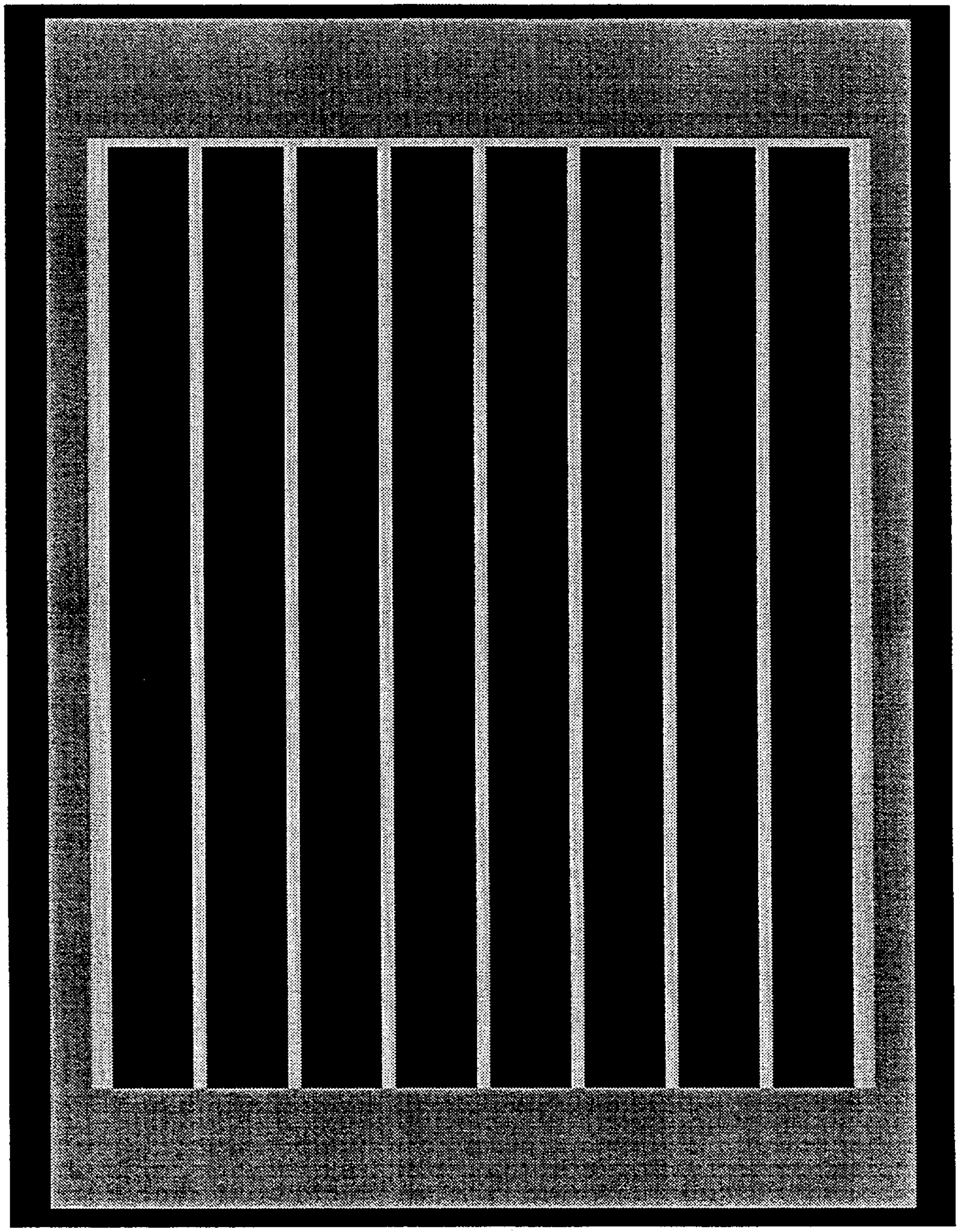


Figure F11.5.18 x-z slice of uranium rods in a cylindrical container

EXAMPLE 2. A TYPICAL PWR SHIPPING CASK

Consider a typical PWR shipping cask, illustrated in Fig. F11.5.19. The interior and exterior of the cask is carbon steel, and a depleted uranium gamma shield is present in the annulus. The shipping cask contains seven PWR fuel assemblies. Each assembly is a 17 × 17 array of fuel rods with water holes as shown. Each assembly is contained in stainless steel. Each fuel rod is clad with Zircaloy and is composed of 4% enriched UO_2. Rods of B_4C clad with stainless steel are positioned between the fuel assemblies. The entire cask is filled with water.

To describe the geometry of the cask, start by defining some simple units as shown in Fig. F11.5.20. Unit 1 represents a fuel rod and its associated square pitch spacing region. Unit 2 represents a water hole in a fuel assembly.

```
UNIT 1
CYLINDER  1 1 .41148 365.76 0.0
CYLINDER  2 1 .48133 365.76 0.0
CUBOID    3 1 .63754 -.63754 .63754 -.63754 365.76 0.0

UNIT 2
CUBOID    3 1 .63754 -.63754 .63754 -.63754 365.76 0.0
```

Units 3, 4, and 6 represent the B_4C rods with their various spacings, and Unit 5 is a water hole that is used in association with some of the B_4C rods.

```
UNIT 3
CYLINDER  4 1 .584 365.76 0.0
CYLINDER  5 1 .635 365.76 0.0
CUBOID    3 1 .9912 -.9912 2.2352 -1.27 365.76 0.0

UNIT 4
CYLINDER  4 1 .584 365.76 0.0
CYLINDER  5 1 .635 365.76 0.0
CUBOID    3 1 .9912 -.9912 1.2702 -2.235 365.76 0.0

UNIT 5
CUBOID    3 1 .9912 -.9912 1.7526 -1.7526 365.76 0.0

UNIT 6
CYLINDER  4 1 .584 365.76 0.0
CYLINDER  5 1 .635 365.76 0.0
CUBOID    3 1 1.1875215 -1.1875215 1.883706 -1.883706 365.76 0.0
```

Units 1 and 2 are stacked together into array 1 to form the array of fuel pins and water holes in a fuel assembly as shown in Fig. F11.5.21. This array is then encompassed with a layer of water and a layer of stainless steel to complete a fuel assembly (Unit 7) as shown in Fig. F11.5.22.

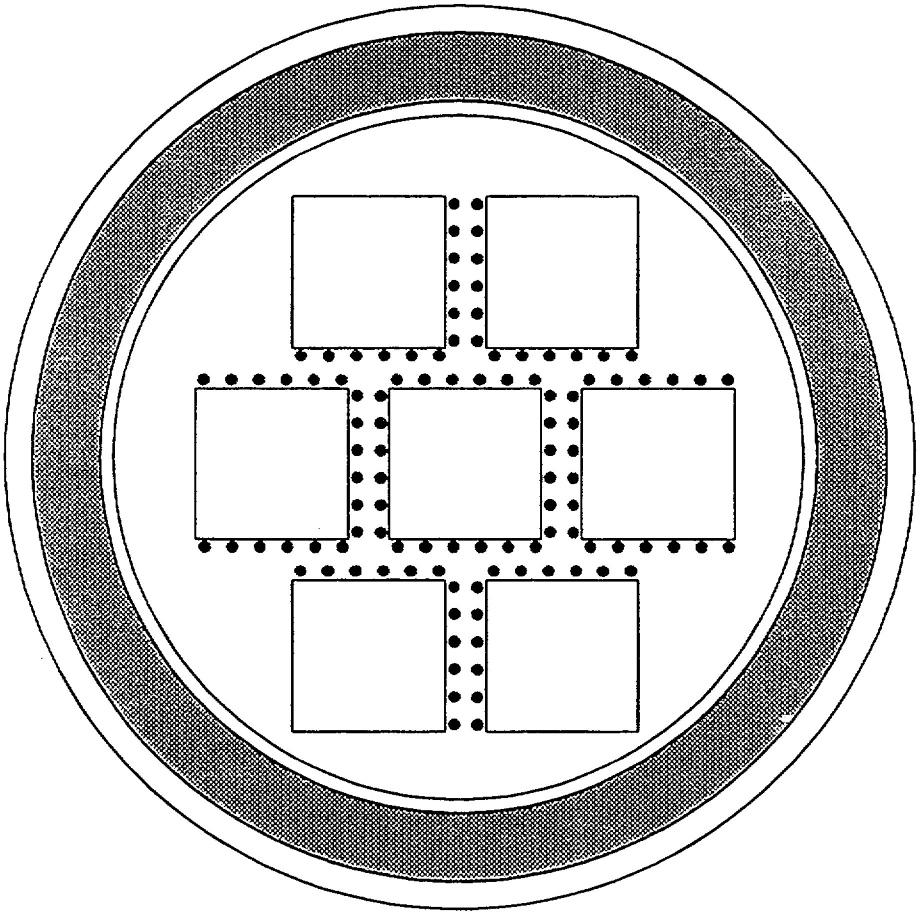


Figure F11.5.19 Typical PWR shipping cask

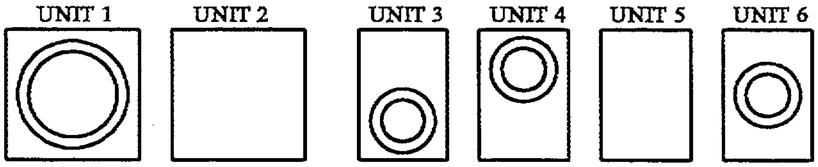


Figure F11.5.20 Simple units

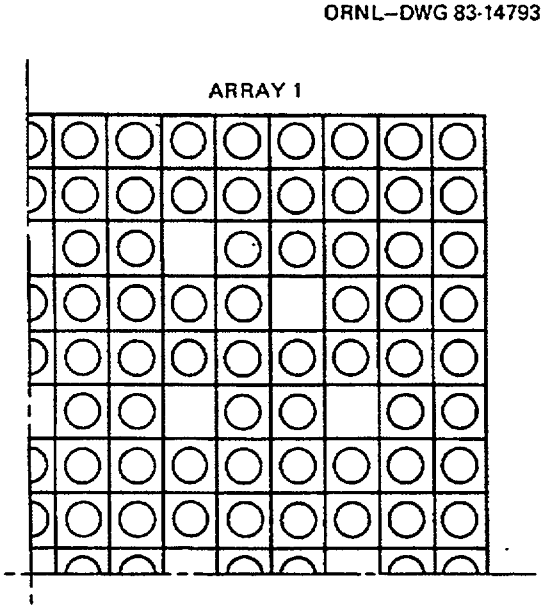


Figure F11.5.21 Quarter section of fuel pin array

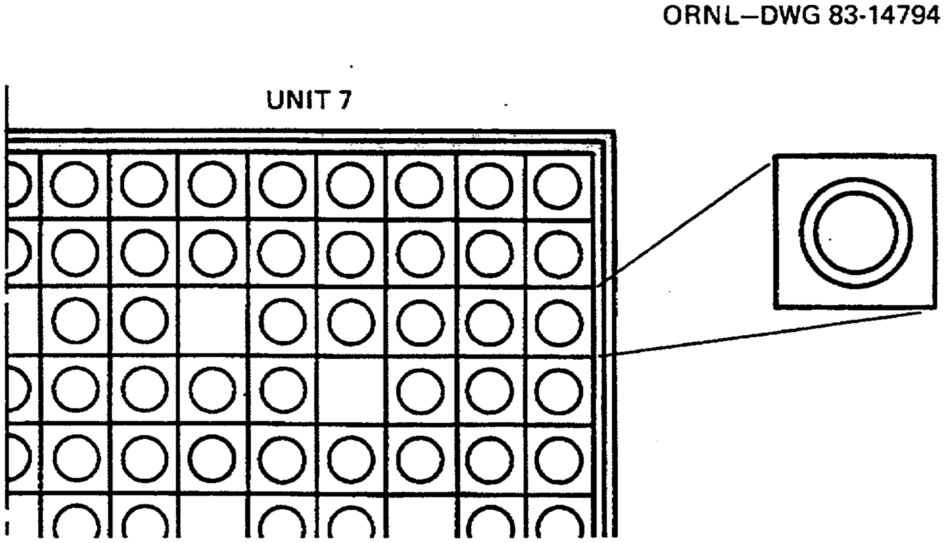


Figure F11.5.22 Quarter section of fuel assembly

```
ARA=1 NUX=17 NUY=17 NUZ=1 FILL
39R1 2 2Q3 8R1 2 9R1 2 22R1 2 4Q3 38R1 2 4Q3
Q51 22R1 2 Q10 Q9 2Q3 39R1            END FILL

UNIT 7    ARRAY 1 -10.83818 -10.83818 0.0
CUBOID    3 1 11.112495 -11.112495 11.112495 -11.112495 365.76 0.0
CUBOID    8 1 11.302238 -11.302238 11.302238 -11.302238 365.76 0.0
```

An array of Unit 6's is created to represent the array of B_4C rods that is positioned between the fuel assemblies. This array of B_4C rods is contained in Unit 8 as shown in Fig. F11.5.23.

```
ARA=2 NUX=2 NUY=6 NUZ=1    FILL F6       END FILL

UNIT 8    ARRAY 2 0 0 0
```

The next step is to create the central array of three fuel assemblies with B_4C rods between them. This is done by stacking fuel assemblies (Unit 7) and B_4C rod arrays (Unit 8) into an array (array 3) and placing it in a unit (Unit 9). The resultant geometry is shown in Fig. F11.5.24.

```
ARA=3 NUX=5 NUY=1 NUZ=1    FILL 7 8 7 8 7 END FILL

UNIT 9    ARRAY 3 0 0 0
```

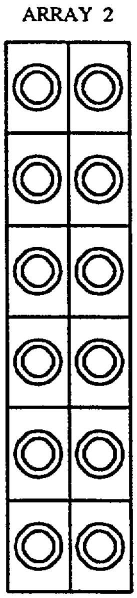


Figure F11.5.23 2 × 6 array of B_4C rods

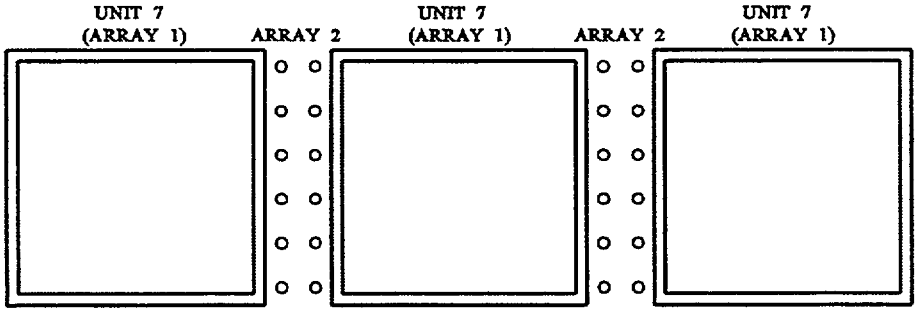


Figure F11.5.24 Central array

Units 3, 4, and 5 are used to define the arrays of B_4C rods that fit above and below the central array, as shown in Fig. F11.5.25.

ARA=4 NUX=39 NUY=1 NUZ=1 FILL 3 5 2Q2 3 4 2Q2 5 4 3 2Q2 5 3 4 2Q2 5 4 3 2Q2 5 2Q2 3 END FILL

UNIT 10 ARRAY 4 0 0 0

ARA=5 NUX=39 NUY=1 NUZ=1 FILL 4 5 2Q2 4 3 2Q2 5 3 4 2Q2 5 4 3 2Q2 5 3 4 2Q2 5 2Q2 4 END FILL

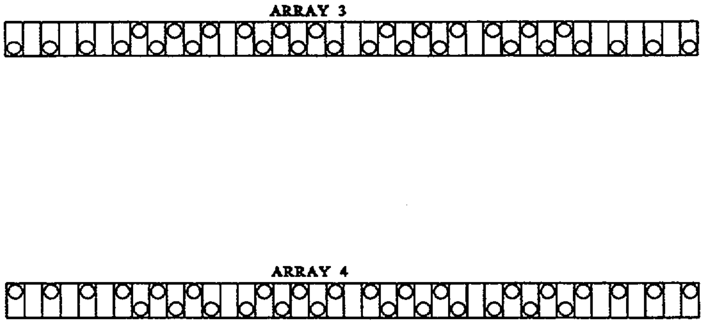


Figure F11.5.25 Long B_4C rod arrays

UNIT 11   ARRAY 5 0 0 0

Units 9, 10, and 11 are stacked to form the central array with B_4C rods as shown in Fig. F11.5.26.

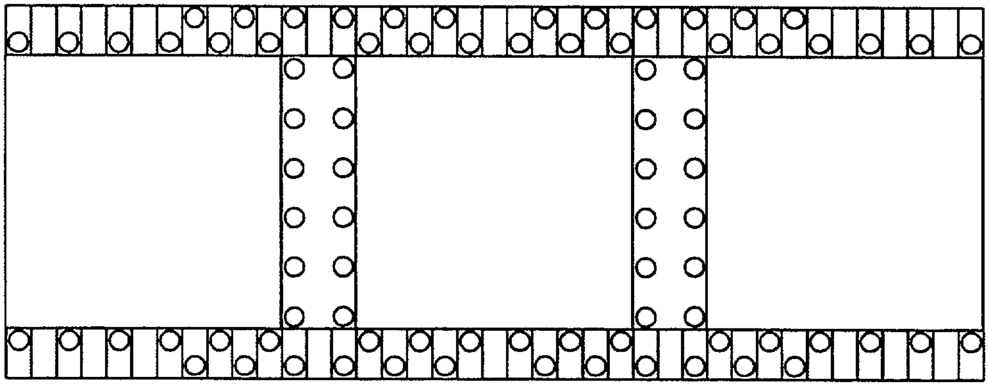


Figure F11.5.26 Central array with long B_4C arrays

ARA=6 NUX=1 NUY=3 NUZ=1   FILL 11 9 10 END FILL

This completes the three central fuel assemblies and all the B_4C rods associated with them. Next, Units 7 and 8 are stacked together to form the array of two fuel assemblies separated by B_4C rods as shown in Fig. F11.5.27. This is designated as array 7 and Unit 12. The origin of Unit 12 is specified at the center of the array in the x and y directions and the bottom of the fuel assemblies are at z=27.94 cm.

ARA=7 NUX=3 NUY=1 NUZ=1   FILL 7 8 7   END FILL

UNIT 12   ARRAY 7 -24.979519 -11.302238 27.94

Unit 13 is simply a cylindrical lid that fits on top of the shipping cask. It is described relative to the origin of the shipping cask and is made of depleted uranium.

UNIT 13
CYLINDER 6 1 47.625 457.2 449.58

The shipping cask is completed by specifying the origin of array 6 (see Fig. F11.5.26) to be at the center of the array in x and y and the bottom of the array is at z=27.94 cm. Note that the three dimensions specified when an ARRAY is placed in a unit are the coordinates of the most negative point in the array with respect to the origin of the unit that contains the array. A cylinder of water defining the interior of the shipping cask is described around the array. A HOLE is used to place a Unit 12 (Fig. F11.5.27) below the array and a second HOLE is used to place another Unit 12 above the array. Then a cylinder of steel is placed around the

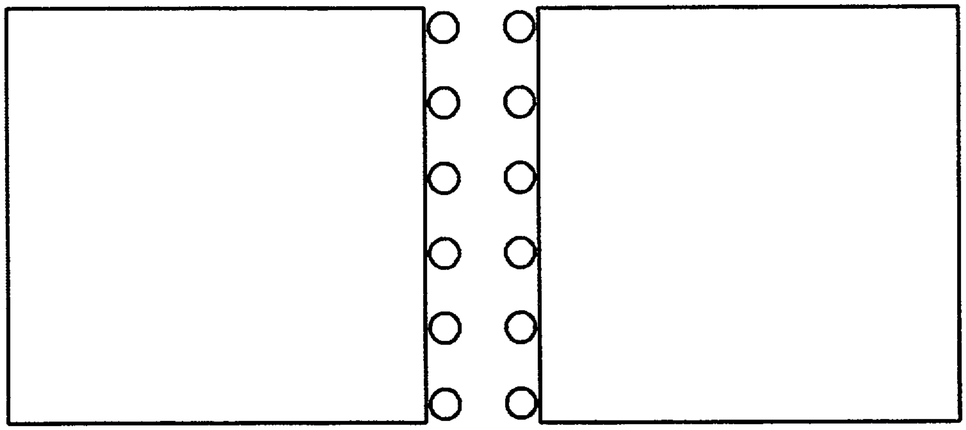


Figure F11.5.27 Two fuel assemblies and B_4C rods

water, which is in turn encased by depleted uranium. The depleted uranium is then contained in the outer steel cylinder of the shipping cask. A third HOLE is used to place the depleted uranium lid (Unit 13) on the shipping cask. This completes the shipping cask description of Fig. F11.5.19.

```
ARRAY    6 -38.6568 -14.807438 27.94
CYLINDER 3 1 47.625 447.04 16.51
HOLE     12   0.0 -26.1097 0.0
HOLE     12   0.0  26.1097 0.0
```

Array 6 is the global array for this problem and is therefore not preceded by a unit definition. It defines the coordinate system for the overall problem. The geometry data for this shipping cask are shown below. The plot data have been included for verification of the geometry description. However, the plot generated by this data is quite large and is therefore not included in this document.

```
READ GEOM
UNIT 1
CYLINDER 1 1 .41148 365.76 0.0
CYLINDER 2 1 .48133 365.76 0.0
CUBOID   3 1 .63754 -.63754 .63754 -.63754 365.76 0.0
UNIT 2
CUBOID   3 1 .63754 -.63754 .63754 -.63754 365.76 0.0
UNIT 3
CYLINDER 4 1 .584 365.76 0.0
CYLINDER 5 1 .635 365.76 0.0
CUBOID   3 1 .9912 -.9912 2.2352 -1.27 365.76 0.0
UNIT 4
CYLINDER 4 1 .584 365.76 0.0
CYLINDER 5 1 .635 365.76 0.0
```

```
CUBOID    3 1 .9912 -.9912 1.2702 -2.235 365.76 0.0
UNIT 5
CUBOID    3 1 .9912 -.9912 1.7526 -1.7526 365.76 0.0
UNIT 6
CYLINDER  4 1 .584 365.76 0.0
CYLINDER  5 1 .635 365.76 0.0
CUBOID    3 1 1.1875215 -1.1875215 1.883706 -1.883706 365.76 0.0
UNIT 7    ARRAY 1 -10.83818 -10.83818 0.0
CUBOID    3 1 11.112495 -11.112495 11.112495 -11.112495 365.76 0.0
CUBOID    8 1 11.302238 -11.302238 11.302238 -11.302238 365.76 0.0
UNIT 8    ARRAY 2 0 0 0
UNIT 9    ARRAY 3 0 0 0
UNIT 10   ARRAY 4 0 0 0
UNIT 11   ARRAY 5 0 0 0
UNIT 12   ARRAY 7 -24.979519 -11.302238 27.94
UNIT 13
CYLINDER  6 1 47.625 457.2 449.58
ARRAY     6  -38.6568 -14.807438 27.94
CYLINDER  3 1 47.625 447.04 16.51
HOLE      12    0.0 -26.1097 0.0
HOLE      12    0.0  26.1097 0.0
CYLINDER  7 1 48.895 447.04 13.335
CYLINDER  6 1 59.06 447.04 3.81
CYLINDER  7 1 63.01 462.28 0.0
HOLE      13    0.0 0.0 0.0
END GEOM
READ ARRAY
ARA=1 NUX=17 NUY=17 NUZ=1 FILL
39R1 2 2Q3 8R1 2 9R1 2 22R1 2 4Q3 38R1 2 4Q3
Q51 22R1 2 Q10 Q9 2Q3 39R1              END FILL
ARA=2 NUX=2 NUY=6 NUZ=1    FILL F6       END FILL
ARA=3 NUX=5 NUY=1 NUZ=1    FILL 7 8 7 8 7 END FILL
ARA=4 NUX=39 NUY=1 NUZ=1   FILL 3 5 2Q2 3 4 2Q2 5 4 3 2Q2 5 3 4 2Q2
5 4 3 2Q2 5 2Q2 3                   END FILL
ARA=5 NUX=39 NUY=1 NUZ=1   FILL 4 5 2Q2 4 3 2Q2 5 3 4 2Q2 5 4 3 2Q2
5 3 4 2Q2 5 2Q2 4                   END FILL
ARA=6 NUX=1 NUY=3 NUZ=1    FILL 11 9  10  END FILL
ARA=7 NUX=3 NUY=1 NUZ=1    FILL 7 8 7     END FILL
END ARRAY
READ PLOT
TTL=? SHIPPING CASK IF-300 X-Y SLICE  ?
XUL=-63 YUL=63 ZUL=180 XLR=63 YLR=-63 ZLR=180
UAX=1 VDN=-1 NAX=350
PLT=NO
END PLOT
```

F11.5.6.5 *Specification of the Global System*

KENO V.a geometry data must be correlated to an absolute coordinate system. In the past, the code was responsible for determining this coordinate system. KENO V.a allows the user to define the absolute coordinate system by specifying the global unit. If the user does not exercise this option, the code will assign the default as before. The use of the word GLOBAL in the geometry region data is used to specify the global unit. A global array is required only when a problem consists of a bare array and thus has no global unit. In that event, the user can specify the global array by entering GBL= in the array data.

In the geometry region data for a problem, the word GLOBAL preceding a UNIT NUMBER DEFINITION or an ARRAY PLACEMENT DESCRIPTION specifies that unit to be the global unit. Only one global specification should be used in a problem. If GLOBAL is entered more than once in the geometry region data, the last entry defines the global system. The specification of a global unit, whether manually or by default, overrides the manual specification of the global array in the array data. Entering the word GLOBAL in the geometry region data always overrides the use of GBL= in the array data.

By default, the global unit is defined to be the last ARRAY PLACEMENT DESCRIPTION that does not immediately follow a UNIT NUMBER DEFINITION. The associated surrounding geometry, if any, is the external reflector. The default global array is the array referenced by the global unit or, in the absence of a global unit, the largest array number specified in the array data. If an array is referenced by the global unit, it is always the first geometry region in the global unit (i.e., an ARRAY PLACEMENT DESCRIPTION is the first geometry region in that unit.) The array number of the ARRAY PLACEMENT DESCRIPTION specifies the array that is contained in the unit. This array may contain other arrays, nested to any level. However, the global array is the one that is considered to be in the global unit, just as each subsequent array within the global array is considered to be within the unit that contains it. If the global array specified in the array data (GBL=) is inconsistent with the specification of the global array in the geometry region data, an error message will be written and the problem will not run. Some examples of global specification follow.

EXAMPLE 1. Consider the 2 × 2 × 2 array of sample problem 1 shown in Fig. F11.D.1. The array consists of four identical units in each of two layers. The geometry for this problem is listed below. By default, the global array is array 1. The most negative point of the array is located at the origin so the array extends from x = 0.0 to x = 27.48, y = 0.0 to y = 27.48 and z = 0.0 to z = 26.02.

```
SAMPLE PROBLEM 1  CASE 2C8 BARE
READ GEOMETRY
CYLINDER 1 1 5.748 5.3825 -5.3825
CUBOID  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
END GEOMETRY
READ ARRAY  NUX=2 NUY=2 NUZ=2  END ARRAY
```

EXAMPLE 2, Case 1. Consider the 2 × 2 × 2 reflected array of sample problem 3 shown in Fig. F11.D.2. The array consists of four identical units in each of two layers. The array is reflected by 15.24 cm of paraffin. The geometry for this problem is listed below. By default, the global unit is the unit whose first geometry region is CORE. Note that the array number following the word CORE is defaulted to 1 if a zero is entered. Array 1 is contained within this unit and is thus the global array. In this geometry description, the most negative point of array 1 is located at (-23.48,-23.48,-22.75) relative to the origin of the global unit. Thus the array extends from x = -23.48 to x = +23.48, y = -23.48 to y = +23.48, and z = -22.75

to z = +22.75. The overall system extends from x = -38.72 to x = +38.72, y = -38.72 to y = +38.72 and z = -37.99 to z = +37.99.

```
SAMPLE PROBLEM 3  2C8  15.24 CM PARAFFIN REFL
READ ARRAY NUX=2 NUY=2 NUZ=2 END ARRAY
READ GEOM
CYLINDER  1 1 5.748 5.3825 -5.3825
CUBOID  0 1 11.74 -11.74 11.74 -11.74 11.375 -11.375
CORE  0 1 -23.48 -23.48 -22.75
CUBOID  2 2 26.48 -26.48 26.48 -26.48 25.75 -25.75
CUBOID  2 3 29.48 -29.48 29.48 -29.48 28.75 -28.75
CUBOID  2 4 32.48 -32.48 32.48 -32.48 31.75 -31.75
CUBOID  2 5 35.48 -35.48 35.48 -35.48 34.75 -34.75
CUBOID  2 6 38.72 -38.72 38.72 -38.72 37.99 -37.99
END GEOM
```

EXAMPLE 2, Case 2. The 2 × 2 × 2 reflected array of sample problem 3 above could also have been described by explicitly specifying the global unit as shown below. The geometry remains unchanged and the results will be identical. A third method would be to omit the "UNIT 2." Again the geometry and results will be unchanged.

```
SAMPLE PROBLEM 3  2C8  15.24 CM PARAFFIN REFL
READ ARRAY NUX=2 NUY=2 NUZ=2 END ARRAY
READ GEOM
CYLINDER  1 1 5.748 5.3825 -5.3825
CUBOID  0 1 11.74 -11.74 11.74 -11.74 11.375 -11.375
GLOBAL
UNIT 2
CORE  0 1 -23.48 -23.48 -22.75
CUBOID  2 2 26.48 -26.48 26.48 -26.48 25.75 -25.75
CUBOID  2 3 29.48 -29.48 29.48 -29.48 28.75 -28.75
CUBOID  2 4 32.48 -32.48 32.48 -32.48 31.75 -31.75
CUBOID  2 5 35.48 -35.48 35.48 -35.48 34.75 -34.75
CUBOID  2 6 38.72 -38.72 38.72 -38.72 37.99 -37.99
END GEOM
```

EXAMPLE 3, Case 1. Consider sample problem 15 illustrated in Fig. F11.D.6. This problem requires the use of two different units to describe the geometry. Unit 1 describes the portion of the sphere that extends into the Plexiglas collar, the Plexiglas collar and a cuboid of water tight fitting around them. Unit 2 describes the portion of the sphere that is above the Plexiglas collar, surrounded by a cuboid of water having an edge dimension the same as the diameter of the Plexiglas collar and a height equal to the partial sphere. These two units are stacked with Unit 2 on top of Unit 1. The global unit is the geometry external to the array (the CORE, CYLINDER and REPLICATE). The global unit is cylindrical in shape and has its origin at the center of the array (0,0,0). The most negative point in the array is positioned at (-12.7,-12.7,-7.092175). Thus the array extends from x = -12.7 to x = +12.7, y = -12.7 to y = +12.7, and z = -7.092175 to z = +7.092175.

```
SAMPLE PROBLEM 15  SMALL WATER REFLECTED SPHERE ON PLEXIGLAS COLLAR
READ GEOM
UNIT 1
HEMISPHE-Z 1 1 6.5537 CHORD -5.09066
CYLINDER  3 1 4.1275 -5.09066 -7.63065
CYLINDER  2 1 12.7   -5.09066 -7.63065
CUBOID  3 1 4P12.7 -5.09066 -7.63065
UNIT 2
HEMISPHE+Z 1 1 6.5537 CHORD 5.09066
CUBOID  3 1 4P12.7 6.5537 -5.09066
CORE  0 1 -12.7 -12.7 -7.092175
CYLINDER  3 1 17.97 2P7.0922
REPLICATE 3 2 3*3.0 5
END GEOM
READ ARRAY NUX=1 NUY=1 NUZ=2  FILL 1 2 END ARRAY
```

EXAMPLE 3, Case 2. Consider sample problem 15 described above. Again define Unit 1 to contain the portion of the sphere that extends into the Plexiglas collar, the Plexiglas collar and a cuboid of water snugly surrounding them. Unit 2 contains the portion of the sphere that is above the Plexiglas collar, surrounded by a cuboid of water having an edge dimension equal to the diameter of the Plexiglas collar and a height equal to the partial sphere. These two units are stacked with Unit 2 on top of Unit 1. The global unit is the geometry external to the array (the CORE, CYLINDER and REPLICATE). The global unit is cylindrical in shape and has its origin at the center of the sphere (0,0,0). The most negative point in the array is positioned at (-12.7,-12.7,-7.63065). Thus the array extends from x = -12.7 to x = +12.7, y = -12.7 to y = +12.7, and z = -7.63065 to z = +6.5537.

```
SAMPLE PROBLEM 15  SMALL WATER REFLECTED SPHERE ON PLEXIGLAS COLLAR
READ GEOM
UNIT 1
HEMISPHE-Z 1 1 6.5537 CHORD -5.09066
CYLINDER  3 1 4.1275 -5.09066 -7.63065
CYLINDER  2 1 12.7   -5.09066 -7.63065
CUBOID  3 1 4P12.7 -5.09066 -7.63065
UNIT 2
HEMISPHE+Z 1 1 6.5537 CHORD 5.09066
CUBOID  3 1 4P12.7 6.5537 -5.09066
CORE  0 1 -12.7 -12.7 -7.63065
CYLINDER  3 1 17.97 6.5537 -7.63065
REPLICATE 3 2 3*3.0 5
END GEOM
READ ARRAY NUX=1 NUY=1 NUZ=2  FILL 1 2 END ARRAY
```

EXAMPLE 3, Case 3. Consider sample problem 15 as described above. The problem geometry can remain basically unchanged, except the global unit is explicitly specified as shown below. The problem is identical to that of EXAMPLE 3, Case 2, and the results will be identical. The same problem could be run by omitting the "UNIT 3" and/or replacing "CORE 0" with "ARRAY 1."

```
SAMPLE PROBLEM 15  SMALL WATER REFLECTED SPHERE ON PLEXIGLAS COLLAR
READ GEOM
UNIT 1
HEMISPHE-Z 1 1 6.5537 CHORD -5.09066
CYLINDER  3 1 4.1275 -5.09066 -7.63065
CYLINDER  2 1 12.7   -5.09066 -7.63065
CUBOID  3 1 4P12.7 -5.09066 -7.63065
UNIT 2
HEMISPHE+Z 1 1 6.5537 CHORD 5.09066
CUBOID  3 1 4P12.7 6.5537 -5.09066
GLOBAL
UNIT 3
CORE  0 1 -12.7 -12.7 -7.63065
CYLINDER  3 1 17.97 6.5537 -7.63065
REPLICATE 3 2 3*3.0 5
END GEOM
READ ARRAY NUX=1 NUY=1 NUZ=2  FILL 1 2 END ARRAY
```

EXAMPLE 4, Case 1. Consider sample problem 19, illustrated in Fig. F11.D.3. The array of arrays geometry for this problem is shown below. Unit 1 defines a Plexiglas cylinder filled with uranyl nitrate solution. Unit 2 defines a uranium metal cylinder. Array 1 defines the 2 × 2 array of solution units at the back of the figure. Array 2 defines the 2 × 2 array of metal units at the front of the figure. Unit 3 contains array 1, and Unit 4 contains array 2. The global array is defined to be array 3 by specifying GBL=3 in the ARRAY DATA. Array 3 is composed of Unit 3 and Unit 4 and defines the overall system. The most negative point in the global array is at the origin of the global unit. The global array extends from x = 0.0 to x = 34.93, y = 0.0 to y = 43.5, z = 0.0 to z = 46.96. This is also the overall dimensions of the implied global array.

```
SAMPLE PROBLEM 19 4 AQUEOUS 4 METAL ARRAY OF ARRAYS
READ GEOM
UNIT 1
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 2P9.525
CUBOID  0 1 4P10.875 2P10.24
UNIT 2
CYLINDER  1 1 5.748 2P5.3825
CUBOID  0 1 4P6.59 2P6.225
UNIT 3
ARRAY 1 3*0.0
UNIT 4
ARRAY 2 3*0.0
REFLECTOR 0 1 2*0.0 2*8.57 2*8.03 1
END GEOM
READ ARRAY  ARA=1 NUX=1 NUY=2 NUZ=2  FILL F1 END FILL
ARA=2 NUX=1 NUY=2 NUZ=2 FILL F2 END FILL GBL=3 ARA=3 NUX=2 NUY=1 NUZ=1
FILL 4 3 END FILL
END ARRAY
```

EXAMPLE 4, Case 2. Sample problem 19 could also have been described by omitting GBL=3. The global array would have defaulted to array 3 because it is the largest array number in the array data.

EXAMPLE 4, Case 3. Still another way of specifying the data for sample problem 19 would be to omit GBL=3 in the array data and specify array 3 as the global array in the geometry data as shown below. The problem would still be identical to the previous descriptions although the geometry is specified in a slightly different manner.

```
SAMPLE PROBLEM 19 4 AQUEOUS 4 METAL ARRAY OF ARRAYS
READ GEOM
UNIT 1
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 2P9.525
CUBOID  0 1 4P10.875 2P10.24
UNIT 2
CYLINDER  1 1 5.748 2P5.3825
CUBOID  0 1 4P6.59 2P6.225
UNIT 3
ARRAY 1 3*0.0
UNIT 4
ARRAY 2 3*0.0
REFLECTOR 0 1 2*0.0 2*8.57 2*8.03 1
GLOBAL
ARRAY 3 3*0.0
END GEOM
READ ARRAY  ARA=1 NUX=1 NUY=2 NUZ=2  FILL F1 END FILL
ARA=2 NUX=1 NUY=2 NUZ=2 FILL F2 END FILL ARA=3 NUX=2 NUY=1 NUZ=1
FILL 4 3 END FILL
END ARRAY
```

EXAMPLE 4, Case 4. Consider the data specified in EXAMPLE 4, Case 1. It is possible to specify a portion of that data to be the global unit. This causes the code to ignore all of the data that are not referenced by the global unit. The 2 × 2 array of uranyl nitrate units can be run by specifying Unit 3 to be the global unit as shown below. Only the 2 × 2 array of solution units will be utilized by the problem.

```
SAMPLE PROBLEM 19 4 AQUEOUS 4 METAL ARRAY OF ARRAYS
READ GEOM
UNIT 1
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 2P9.525
CUBOID  0 1 4P10.875 2P10.24
UNIT 2
CYLINDER  1 1 5.748 2P5.3825
CUBOID  0 1 4P6.59 2P6.225
GLOBAL
UNIT 3
ARRAY 1IT 3
ARRAY 1 3*0.0
UNIT 4
ARRAY 2 3*0.0
REFLECTOR 0 1 2*0.0 2*8.57 2*8.03 1
END GEOM
READ ARRAY  ARA=1 NUX=1 NUY=2 NUZ=2  FILL F1 END FILL
ARA=2 NUX=1 NUY=2 NUZ=2 FILL F2 END FILL GBL=3 ARA=3 NUX=2 NUY=1 NUZ=1
FILL 4 3 END FILL
END ARRAY
```

EXAMPLE 4, Case 2. Sample problem 19 could also have been described by omitting GBL=3. The global array would have defaulted to array 3 because it is the largest array number in the array data.

EXAMPLE 4, Case 3. Still another way of specifying the data for sample problem 19 would be to omit GBL=3 in the array data and specify array 3 as the global array in the geometry data as shown below. The problem would still be identical to the previous descriptions although the geometry is specified in a slightly different manner.

```
SAMPLE PROBLEM 19 4 AQUEOUS 4 METAL ARRAY OF ARRAYS
READ GEOM
UNIT 1
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 2P9.525
CUBOID  0 1 4P10.875 2P10.24
UNIT 2
CYLINDER  1 1 5.748 2P5.3825
CUBOID  0 1 4P6.59 2P6.225
UNIT 3
ARRAY 1 3*0.0
UNIT 4
ARRAY 2 3*0.0
REFLECTOR 0 1 2*0.0 2*8.57 2*8.03 1
GLOBAL
ARRAY 3 3*0.0
END GEOM
READ ARRAY  ARA=1 NUX=1 NUY=2 NUZ=2  FILL F1 END FILL
ARA=2 NUX=1 NUY=2 NUZ=2 FILL F2 END FILL ARA=3 NUX=2 NUY=1 NUZ=1
FILL 4 3 END FILL
END ARR
```

EXAMPLE 4, Case 4. Consider the data specified in EXAMPLE 4, Case 1. It is possible to specify a portion of that data to be the global unit. This causes the code to ignore all of the data that are referenced by the global unit. The 2 × 2 array of uranyl nitrate units can be run by specifying Unit 3 to be the global unit as shown below. Only the 2 × 2 array of solution units will be utilized by the problem.

```
SAMPLE PROBLEM 19 4 AQUEOUS 4 METAL ARRAY OF ARRAYS
READ GEOM
UNIT 1
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 2P9.525
CUBOID  0 1 4P10.875 2P10.24
UNIT 2
CYLINDER  1 1 5.748 2P5.3825
CUBOID  0 1 4P6.59 2P6.225
GLOBAL
UNIT 3
ARRAY 1
```

Case 6 of EXAMPLE 4 is equivalent to the data shown below.

```
SAMPLE PROBLEM 19 4 AQUEOUS 4 METAL ARRAY OF ARRAYS
READ GEOM
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 2P9.525
CUBOID  0 1 4P10.875 2P10.24
END GEOM
```

F11.5.6.6 *Triangular-Pitched Arrays*

Triangular-pitched arrays can be described in KENO V.a geometry by properly defining the basic unit from which the array can be built. This includes close-packed triangular-pitched arrays. Two geometry configurations are described below.

EXAMPLE 1
BARE TRIANGULAR-PITCHED ARRAY

Figure F11.5.28 illustrates a small close-packed triangular-pitched array. Each rod in the array has a radius of 2.0 cm, and the pitch of the array is 4 cm. To create this array, describe five units as defined in Fig. F11.5.29.

Assume the rods described in the array are each 2.0 cm in radius and 100 cm tall. The rods are composed of mixture 1. The geometry descriptions for the first four units are given below.

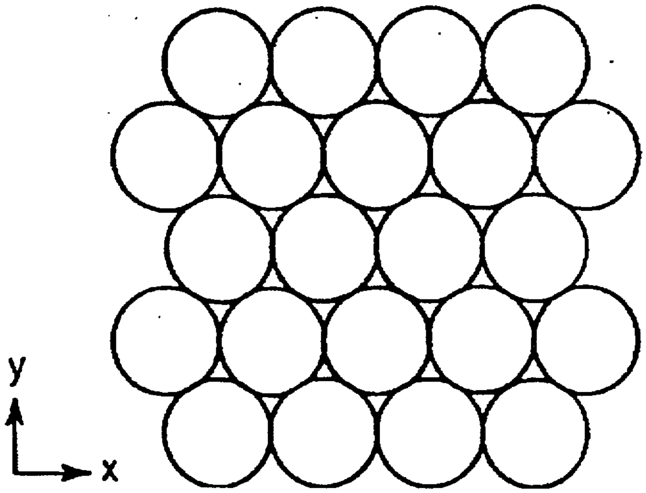


Figure F11.5.28 Bare triangular-pitched array

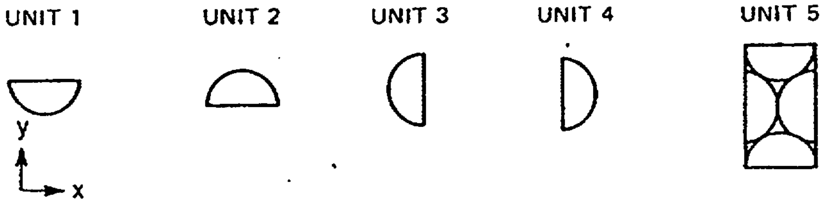


Figure F11.5.29 Units used to describe a bare triangular-pitched array

```
UNIT 1
ZHEMICYL-Y 1 1 2.0 50.0 -50.0

UNIT 2
ZHEMICYL+Y 1 1 2.0 50.0 -50.0

UNIT 3
ZHEMICYL-X 1 1 2.0 50.0 -50.0

UNIT 4
ZHEMICYL+X 1 1 2.0 50.0 -50.0
```

To describe Unit 5, define the origin of the unit to be at its center. Build one of the hemicylinders into the box and add the other three as holes. In this example, the +X hemicylinder is built into the box and the other hemicylinders are inserted as holes. (Because the +X hemicylinder is built into Unit 5, Unit 4 is not used in the problem.) The half dimension of the box in the X dimension is equal to the radius, 2.0 cm. The half dimension of the box in the Y direction is $\sqrt{3}/2$ times the pitch (0.866025 * 4.0) or 3.46411 cm. Unit 4 is described below.

```
UNIT 5
ZHEMICYL+X  1 1 2.0 50.0 -50.0 ORIGIN -2.0 0.0
CUBOID  0 1 2P2.0 2P3.46411 2P50.0
HOLE 1 0.0  3.46411  0.0
HOLE 2 0.0 -3.46411  0.0
HOLE 3 2.0           0.0
```

In the description of Unit 4, the ZHEMICYL+X places the hemicylinder at the left of the box. HOLE 1 places the top hemicylinder, HOLE 2 places the bottom hemicylinder, and HOLE 3 places the hemicylinder at the right of the box.

Next, define a Unit 6 that can be used to complete the lower rod of Unit 5. Define a Unit 7 that can be used to complete the upper rod of Unit 5. Define Unit 8 to complete the left rod of Unit 5, and define Unit 9 to complete the right rod of Unit 5. Unit 10 is defined to complete the corners of the overall array. The input data for these units are given below and are illustrated in Fig. F11.5.30.

```
UNIT 6
ZHEMICYL-Y   1 1 2.0 50.0 -50.0
CUBOID       0 1 2P2.0 0.0 -2.0 50.0 -50.0

UNIT 7
ZHEMICYL+Y   1 1 2.0 50.0 -50.0
CUBOID       0 1 2P2.0 2.0 0.0 2P50.0

UNIT 8
ZHEMICYL-X   1 1 2.0 50.0 -50.0
CUBOID       0 1 0.0 -2.0 2P3.46411 2P50.0

UNIT 9
ZHEMICYL+X   1 1 2.0 50.0 -50.0
CUBOID       0 1 2.0 0.0 2P3.46411 2P50.0

UNIT 10
CUBOID       0 1 2.0 0.0 2.0 0.0 2P50.0
```

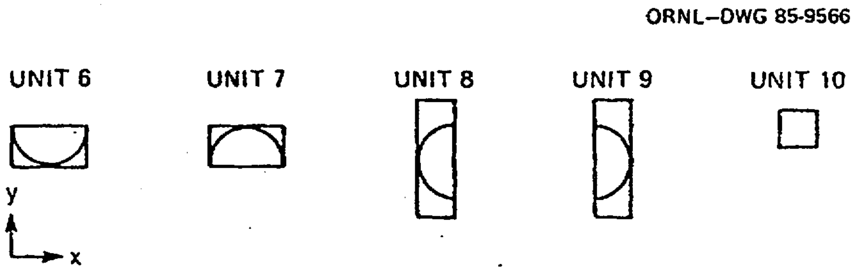


Figure F11.5.30 Units to complete the triangular-pitched array

Figure F11.5.31 shows the arrangement of the units to complete the array. The array data to describe the array are shown below.

ARA=1 NUX=6 NUY=4 NUZ=1 FILL 10 4R6 10 8 4R5 9 1Q6 10 4R7 10 END FILL

The bottom row of the array is described by the data entries 10 4R6 10. The second row of the array is described by the data entries 8 4R5 9. The third row is filled by repeating the previous six entries (1Q6). It could also have been described by entering 8 4R5 9. The top row of the array is described by the data entries 10 4R7 10.

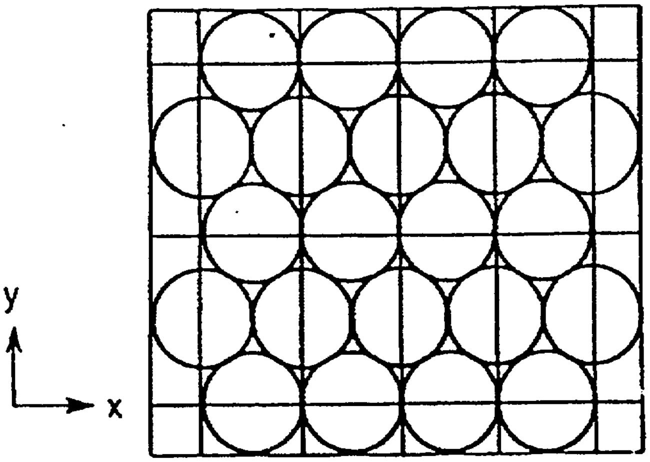


Figure F11.5.31 Completed array

EXAMPLE 2a.
TRIANGULAR-PITCHED ARRAY IN A CYLINDER

Figure F11.5.32 illustrates a close-packed triangular-pitched array in a cylinder. This array may be described by defining five basic units which are the same as those of Example 1 shown in Fig. F11.5.29.

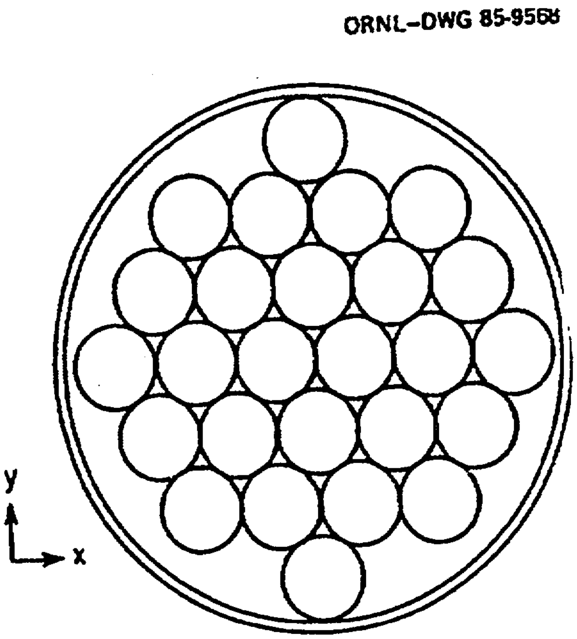


Figure F11.5.32 Triangular-pitched array within a cylinder

```
UNIT 1
ZHEMICYL-Y 1 1 2.0 50.0 -50.0

UNIT 2
ZHEMICYL+Y 1 1 2.0 50.0 -50.0

UNIT 3
ZHEMICYL-X 1 1 2.0 50.0 -50.0

UNIT 4
ZHEMICYL+X 1 1 2.0 50.0 -50.0
```

To describe Unit 5, define the origin of the unit to be at its center. Build one of the hemicylinders into the box and add the other three as holes. In this example, the +X hemicylinder is built into the box and the other hemicylinders are inserted as holes. The half dimension of the box in the X dimension is equal to the radius, 2.0 cm. The half dimension of the box in the Y direction is $\sqrt{3}/2$ times the pitch (0.866025 * 4.0) or 3.46411 cm. Unit 5 is described below.

```
UNIT 5
ZHEMICYL+X 1 1 2.0 50.0 -50.0 ORIGIN -2.0 0.0
CUBOID    0 1 2P2.0 2P3.46411 2P50.0
HOLE 1  0.0  3.46411 0.0
HOLE 2  0.0 -3.46411 0.0
HOLE 3  2.0  0.0     0.0
```

In the description of UNIT 5, the ZHEMICYL+X places the hemicylinder at the left of the box. HOLE 1 places the top hemicylinder, HOLE 2 places the bottom hemicylinder, and HOLE 3 places the hemicylinder at the right of the box.

To describe the base array of the problem, stack Units 5 in a 4 × 2 × 1 array as shown in Fig. F11.5.33. The input data for the array are the following:

ORNL–DWG 85-9569

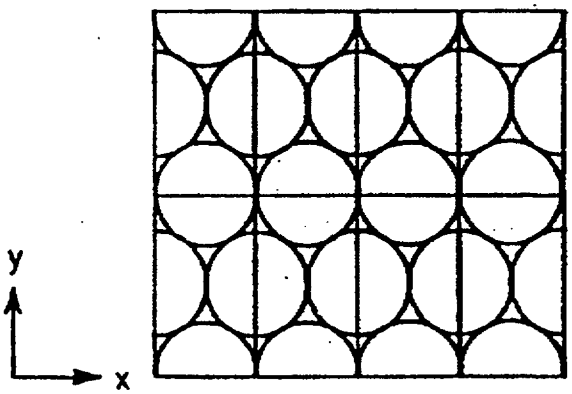


Figure F11.5.33 4 × 2 × 1 array to be placed within a cylinder

ARA=1 NUX=4 NUY=2 NUZ=1 FILL F5 END FILL

Next, place the array within the cylinder. This is done by placing the array in a unit, defined here to be Unit 6. The origin of the cylinder has been defined to be at the center of the array. The result of these data are shown in Fig. F11.5.34.

UNIT 6
ARRAY 1 -8.0 -6.92822 -50.0
CYLINDER 0 1 12.4 2P50.0
CYLINDER 2 1 12.65 2P50.0

Next, add the hemicylinders necessary to complete all of the half cylinders remaining in Fig. F11.5.34. This is done by placing four Units 1 at the appropriate positions along the bottom of the array, four Units 2 at the top of the array, two Units 3 at the left of the array, and two Units 4 at the right of the array. The input data are shown below, and the resultant configuration is shown in Fig. F11.5.35. In Unit 6, the first HOLE 1 places a Unit 1 under the lower-left unit of the array. The second HOLE 1 places a Unit 1 under the next array unit to the right of the first one. This procedure is repeated for the next two lower array units, thus completing the lower row of cylinders. Similarly, the first HOLE 2 places a Unit 2 above the upper-left unit of the array.

The second HOLE 2 places a Unit 1 to the right of the first one, etc., until the four cylinders at the top of the array are completed. The first HOLE 3 places a Unit 3 at the lower-left side of the array to complete that rod. The second HOLE 3 completes the rod above it. The first HOLE 4 completes the lower rod on the right side of the array. The second HOLE 4 completes the rod above it. The geometry data listed below result in the configuration shown in Fig. F11.5.35.

ORNL–DWG 85-9570

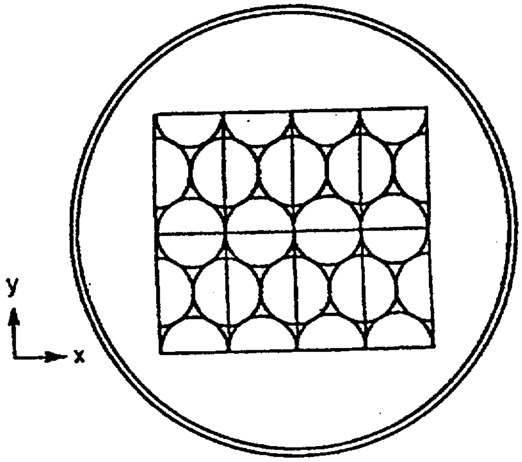


Figure F11.5.34 4 × 2 × 1 array within a cylinder

ORNL–DWG 85-9571

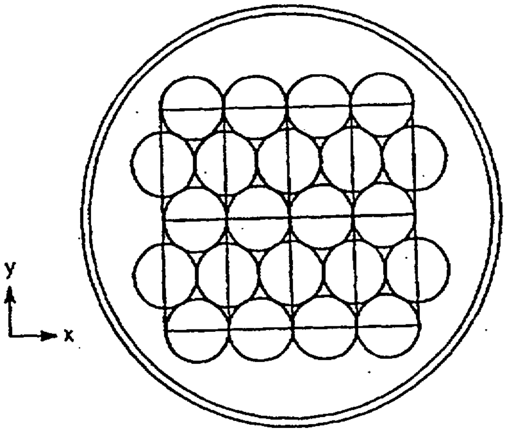


Figure F11.5.35 Partially completed triangular-pitched array in a cylinder

```
UNIT 6
ARRAY 1 -8.0 -6.92822 -50.0
CYLINDER 0 1 12.4  2P50.0
HOLE 1  -6.0 -6.92822 0.0
HOLE 1  -2.0 -6.92822 0.0
HOLE 1   2.0 -6.92822 0.0
HOLE 1   6.0 -6.92822 0.0
HOLE 2  -6.0  6.92822 0.0
HOLE 2  -2.0  6.92822 0.0
HOLE 2   2.0  6.92822 0.0
HOLE 2   6.0  6.92822 0.0
HOLE 3  -8.0 -3.46411 0.0
HOLE 3  -8.0  3.46411 0.0
HOLE 4   8.0 -3.46411 0.0
HOLE 4   8.0  3.46411 0.0
CYLINDER 2 1 12.65 2P50.0
```

To complete the desired configuration, define a cylinder, Unit 7 and place it at the four appropriate positions as shown below. The first HOLE 7 places the cylinder of Unit 7 at the left of the array, the second HOLE 7 places the cylinder at the top of the array, the third HOLE 7 places the cylinder at the right of the array, and the fourth HOLE 7 places the cylinder at the bottom of the array. The completed configuration is shown in Fig. F11.5.36. It is not necessary for Unit 7 to precede Unit 6. It is allowable to place Unit 7 after Unit 6 in the input data. Because the final configuration is defined in Unit 6, it must be designated as the global unit. The total geometry input for this example is listed below.

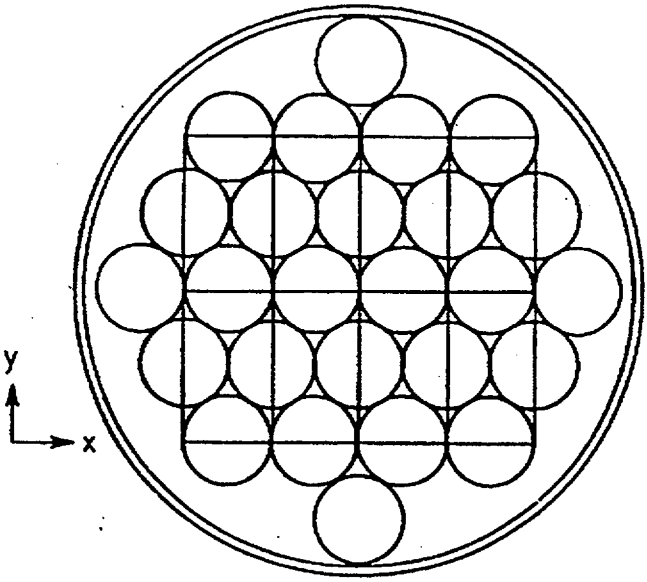


Figure F11.5.36 Completed triangular-pitched array in a cylinder

```
READ GEOM

UNIT 1
ZHEMICYL-Y 1 1 2.0 50.0 -50.0

UNIT 2
ZHEMICYL+Y 1 1 2.0 50.0 -50.0

UNIT 3
ZHEMICYL-X 1 1 2.0 50.0 -50.0

UNIT 4
ZHEMICYL+X 1 1 2.0 50.0 -50.0

UNIT 5
ZHEMICYL+X 1 1 2.0 50.0 -50.0 ORIGIN -2.0 0.0
CUBOID    0 1 2P2.0 2P3.46411 2P50.0
HOLE 1  0.0  3.46411 0.0
HOLE 2  0.0 -3.46411 0.0
HOLE 3  2.0  0.0     0.0

UNIT 7
CYLINDER 1 1 2.0 2P50.0
GLOBAL

UNIT 6
ARRAY 1 -8.0 -6.92822 -50.0
CYLINDER 0 1  12.4  2P50.0
HOLE 1  -6.0  -6.92822 0.0
HOLE 1  -2.0  -6.92822 0.0
HOLE 1   2.0  -6.92822 0.0
HOLE 1   6.0  -6.92822 0.0
HOLE 2  -6.0   6.92822 0.0
HOLE 2  -2.0   6.92822 0.0
HOLE 2   2.0   6.92822 0.0
HOLE 2   6.0   6.92822 0.0
HOLE 3  -8.0  -3.46411 0.0
HOLE 3  -8.0   3.46411 0.0
HOLE 4   8.0  -3.46411 0.0
HOLE 4   8.0   3.46411 0.0
HOLE 7 -10.0   0.0     0.0
HOLE 7   0.0  10.39233 0.0
HOLE 7  10.0   0.0     0.0
HOLE 7   0.0 -10.39233 0.0
CYLINDER 2 1  12.65 2P50.0
END GEOM
```

```
READ ARRAY

ARA=1 NUX=4 NUY=2 NUZ=1  FILL F5 END FILL
END ARRAY
```

EXAMPLE 2b.
ALTERNATIVE MOCKUP OF TRIANGULAR-PITCHED ARRAY IN A CYLINDER

Consider the triangular-pitched array shown in Fig. F11.5.32. Another method of describing this configuration is given below. Define four basic units as listed below. These are the same units shown in Fig. F11.5.29.

```
UNIT 1
ZHEMICYL-Y 1 1 2.0 50.0 -50.0

UNIT 2
ZHEMICYL+Y 1 1 2.0 50.0 -50.0

UNIT 3
ZHEMICYL-X 1 1 2.0 50.0 -50.0

UNIT 4
ZHEMICYL+X 1 1 2.0 50.0 -50.0
```

Unit 5 is the same as previously defined in Example 2a, and pictured in Fig. F11.5.29.

```
UNIT 5
ZHEMICYL+X 1 1 2.0 50.0 -50.0 ORIGIN -2.0 0.0
CUBOID     0 1 2P2.0 2P3.46411 2P50.0
HOLE 1  0.0  3.46411 0.0
HOLE 2  0.0 -3.46411 0.0
HOLE 3  2.0  0.0     0.0
```

To describe the basic array for the problem, stack Units 5 in a 4x2x1 as shown in Fig. F11.5.33. The input data for the array are the following:

```
ARA=1 NUX=4 NUY=2 NUZ=1  FILL F5 END FILL
```

Next, place the array (array 1) in Unit 6 and define Units 7 and 8 to be placed to the left and right of it (see Fig. F11.5.37). Unit 7 will complete the two rods at the left boundary of the array and will contain half of the far left rod in the completed configuration. In the description of Unit 7, the ZHEMICYL+X is half of the far right rod and is located with its cut face at the left boundary of a box that is as tall as the entire array of Fig. F11.5.33. The first HOLE 3 in Unit 7 completes the lower left rod of that array and the second HOLE 3 completes the upper left rod. Unit 8 is constructed in similar fashion to complete the two rods at the right of the array shown in Fig. F11.5.33. Unit 8 is the mirror image of Unit 7. Stack Units 6, 7, and 8 in an array (array 2) to achieve the configuration shown in Fig. F11.5.37. The data to accomplish this are listed below.

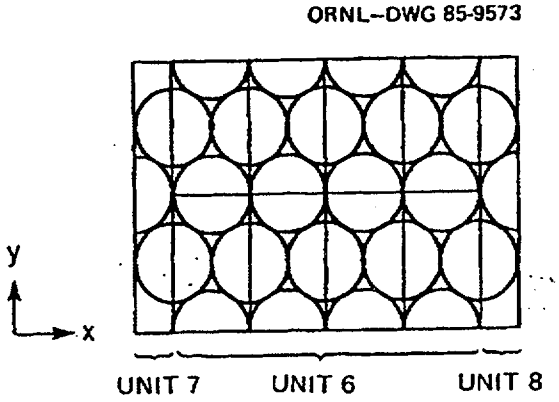


Figure F11.5.37 Array to be placed in a cylinder

```
UNIT 6
ARRAY 1 -8.0 -6.92822 -50.0
GLOBAL

UNIT 7
ZHEMICYL+X 1 1 2.0 50.0 -50.0
CUBOID    0 1 2.0 0.0 2P6.92822 2P50.0
HOLE 3 2.0 -3.46411 0.0
HOLE 3 2.0  3.46411 0.0

UNIT 8
ZHEMICYL-X 1 1 2.0 50.0 -50.0
CUBOID    0 1 0.0 -2.0 2P6.92822 2P50.0
HOLE 4 -2.0 -3.46411 0.0
HOLE 4 -2.0  3.46411 0.0

ARA=2 NUX=3 NUY=1 NUZ=1 FILL 7 6 8 END FILL
```

Next, place array 2 in the cylinder as shown in Fig. F11.5.38. The data are listed below.

```
UNIT 9
ARRAY 2 -10.0 -6.92822 -50.0
CYLINDER 0 1 12.4  2P50.0
CYLINDER 2 1 12.65 2P50.0
```

Now describe Units 10 and 11 to be placed above and below the array. These units are shown in Fig. F11.5.39. Unit 10 is described to complete the two central rods at the top of the array of Fig. F11.5.38. See Figs. F11.5.39 and F11.5.40 to assist in visualizing these units. The ZHEMICYL-Y is placed at the top of the unit to describe half of the rod at the very top of the array. The first HOLE 2 completes the left center rod at the top of the array pictured in Fig. F11.5.38. The second HOLE 2 completes the right center rod at the top of the array. Unit 11 is described in similar fashion. It is the mirror image of Unit 10 and is placed below the array of Fig. F11.5.38. The resultant configuration is shown in Fig. F11.5.39.

```
UNIT 10
ZHEMICYL-Y 1 1 2.0 2P50.0
CUBOID  0 1 2P4.0 0.0 -3.46411 2P50.0
HOLE 2 -2.0 -3.46411 0.0 2P50.0
HOLE 2  2.0 -3.46411 0.0

UNIT 11
ZHEMICYL+Y 1 1 2.0 2P50.0
CUBOID  0 1 2P4.0 3.46411 0.0 2P50.0
HOLE 1 -2.0 3.46411 0.0
HOLE 1  2.0 3.46411 0.0
```

Now place Units 10 and 11 above and below the array of Fig. F11.5.38 to obtain the configuration shown in Fig. F11.5.40.

ORNL–DWG 85-9574

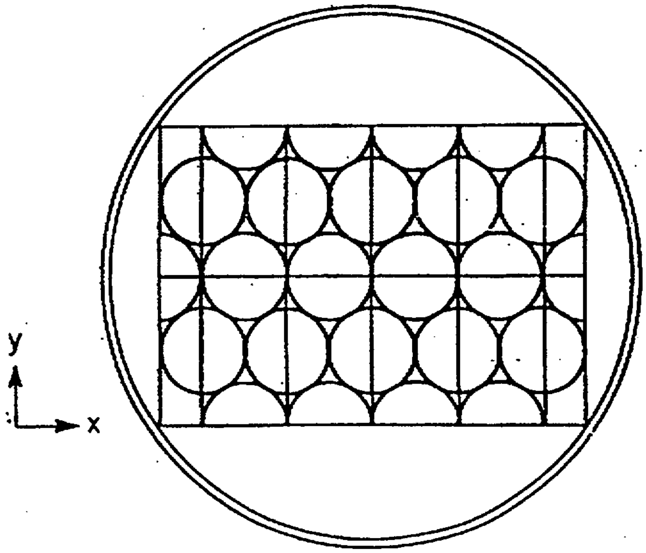


Figure F11.5.38 Partially completed array in a cylinder

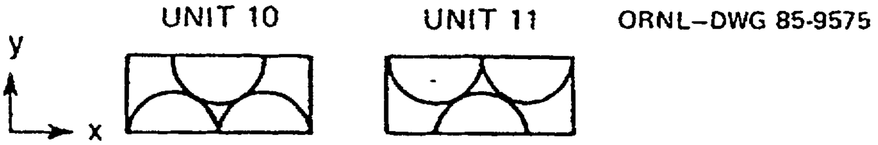


Figure F11.5.39 Description of Units 10 and 11 for Example 2b

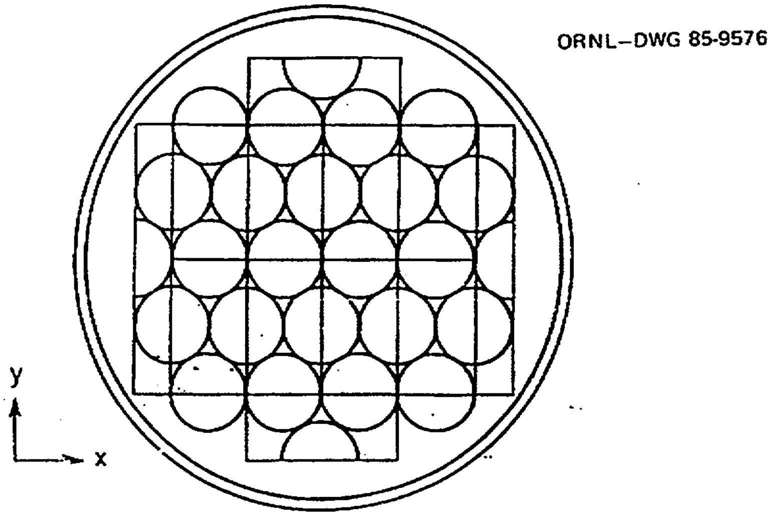


Figure F11.5.40 Partially completed triangular-pitched array in a cylinder

```
UNIT 9
ARRAY 2 -10.0 -6.92822 -50.0
CYLINDER 0 1 12.4  2P50.0
HOLE 10 0.0 10.39233 0.0
HOLE 11 0.0 -10.39233 0.0
CYLINDER 2 1 12.65 2P50.0
```

To complete the array the remaining half-cylinders must be entered as holes as shown below. The first HOLE 1 completes the half-rod at the lower left of Fig. F11.5.40. The second HOLE 1 completes the half-rod at the lower center, and the third HOLE 1 completes the half-rod at the lower right of Fig. F11.5.40. Similarly, the first HOLE 2 completes the half-rod at the upper left of Fig. F11.5.40. The second HOLE 2 completes the half-rod at the upper center, and the third HOLE 2 completes the half-rod at the upper right. The HOLE 3 completes the half-rod at the left of Fig. F11.5.40, and the HOLE 4 completes the half-rod at the right. The final geometry configuration is shown in Fig. F11.5.41. Unit 9 must be specified as the global unit because it defines the overall configuration.

```
GLOBAL
UNIT 9
ARRAY 2 -10.0 -6.92822 -50.0
CYLINDER 0 1 12.4  2P50.0
HOLE 10  0.0 10.39233 0.0
HOLE 11  0.0 -10.39233 0.0
HOLE 1  -6.0  -6.92822 0.0
HOLE 1   0.0 -10.39233 0.0
HOLE 1   6.0  -6.92822 0.0
HOLE 2  -6.0   6.92822 0.0
HOLE 2   0.0  10.39233 0.0
HOLE 2   6.0   6.92822 0.0
HOLE 3 -10.0  0.0     0.0
HOLE 4  10.0  0.0     0.0
CYLINDER 2 1 12.65 2P50.0
```

ORNL–DWG 85-9577

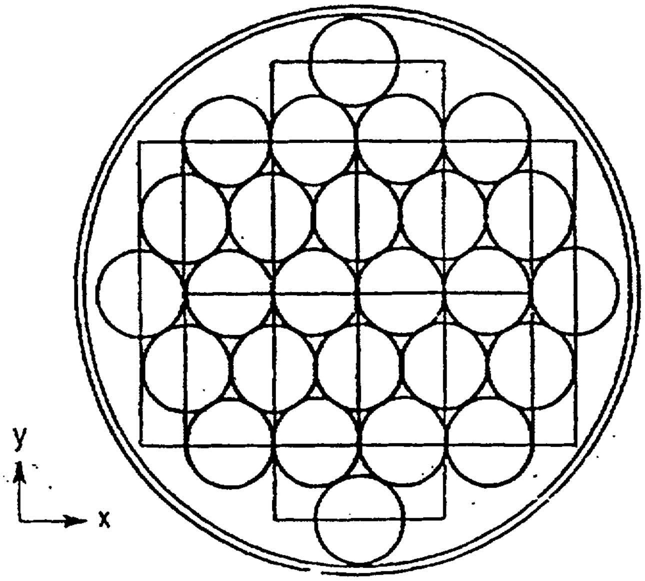


Figure F11.5.41 Final configuration of triangular-pitched array in a cylinder

The geometry data for Example 2b are given below.

```
READ GEOM

UNIT 1
ZHEMICYL-Y 1 1 2.0 50.0 -50.0

UNIT 2
ZHEMICYL+Y 1 1 2.0 50.0 -50.0

UNIT 3
ZHEMICYL-X 1 1 2.0 50.0 -50.0

UNIT 4
ZHEMICYL+X 1 1 2.0 50.0 -50.0
```

```
UNIT 5
ZHEMICYL+X 1 1 2.0 50.0 -50.0 ORIGIN -2.0 0.0
CUBOID     0 1 2P2.0 2P3.46411 2P50.0
HOLE 1  0.0  3.46411 0.0
HOLE 2  0.0 -3.46411 0.0
HOLE 3  2.0  0.0     0.0

UNIT 6
ARRAY 1 -8.0 -6.92822 -50.0

UNIT 7
ZHEMICYL+X 1 1 2.0 50.0 -50.0
CUBOID     0 1 2.0 0.0 2P6.92822 2P50.0
HOLE 3 2.0 -3.46411 0.0
HOLE 3 2.0  3.46411 0.0

UNIT 8
ZHEMICYL-X 1 1 2.0 50.0 -50.0
CUBOID     0 1 0.0 -2.0 2P6.92822 2P50.0
HOLE 3 0.0 -3.46411 0.0
HOLE 3 0.0  3.46411 0.0

UNIT 10
YZHEMICYL-Y 1 1 2.0 2P50.0
CUBOID  0 1 2P4.0 0.0 -3.46411 2P50.0
HOLE 2 -2.0 -3.46411 0.0
HOLE 2  2.0 -3.46411 0.0

UNIT 11
YZHEMICYL+Y 1 1 2.0 2P50.0
CUBOID  0 1 2P4.0 3.46411 0.0 2P50.0
HOLE 1 -2.0 3.46411 0.0
HOLE 1  2.0 3.46411 0.0
GLOBAL

UNIT 9
ARRAY 2 -10.0 -6.92822 -50.0
CYLINDER 0 1 12.4  2P50.0
HOLE 10  0.0   10.39233  0.0
HOLE 11  0.0  -10.39233  0.0
HOLE 1  -6.0   -6.92822  0.0
HOLE 1   0.0  -10.39233  0.0
HOLE 1   6.0   -6.92822  0.0
HOLE 2  -6.0    6.92822  0.0
HOLE 2   0.0   10.39233  0.0
HOLE 2   6.0    6.92822  0.0
HOLE 3 -10.0    0.0      0.0
HOLE 4 10.0   0.0      0.0
CYLINDER 2 1 12.65 2P50.0
END GEOM

READ ARRAY
ARA=1 NUX=4 NUY=2 NUZ=1  FILL F5 END FILL
ARA=2 NUX=3 NUY=1 NUZ=1 FILL 7 6 8 END FILL
END ARRAY
```

F11.5.7 ALTERNATIVE SAMPLE PROBLEM MOCKUPS

The geometry data for KENO V.a can often be described correctly in several ways. Some alternative geometry descriptions are given here for sample problem 12 and sample problem 13. (See Sect. F11.D.)

F11.5.7.1 *Sample Problem 12, First Alternative*

This mock-up maintains the same overall unit dimensions that were used in sample problem 12. In sample problem 12, the origin of Unit 1, the solution cylinder, is at the center of the unit; the origin of Units 2, 3, and 4, the metal cylinders, are at the center of the cylinders. In this mock-up, the unit numbers remain the same and the origin of each unit is at the center of the unit. In each unit the cylinder is offset by specifying the position of its centerline relative to the origin of the unit.

```
READ GEOM
UNIT 1
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 9.525 -9.525
CUBOID    0 1 10.875 -10.875 10.875 -10.875 10.24 -10.24
UNIT 2
CYLINDER  1 1 5.748 9.3975 -1.3975 ORIG 4.285 4.285
CUBOID    0 1 10.875 -10.875 10.875 -10.875 10.24 -10.24
UNIT 3
CYLINDER  1 1 5.748 9.3975 -1.3675 ORIG 4.285 -4.285
CUBOID    0 1 10.875 -10.875 10.875 -10.875 10.24 -10.24
UNIT 4
CYLINDER  1 1 5.748 1.3675 -9.3975 ORIG 4.285 4.285
CUBOID    0 1 10.875 -10.875 10.875 -10.875 10.24 -10.24
UNIT 5
CYLINDER  1 1 5.748 1.3675 -9.3975 ORIG 4.285 -4.285
CUBOID    0 1 10.875 -10.875 10.875 -10.875 10.24 -10.24
END GEOM
READ ARRAY  NUX=2 NUY=2 NUZ=2 FILL 2 1 3 1 4 1 5 1 END ARRAY
```

F11.5.7.2 *Sample Problem 12, Second Alternative*

In this mock-up, the outer boundaries of the system are made as close fitting as possible on all six faces. The origin of each unit is located at the center of the cylinder. Units 1, 3, 5, and 7 contain the metal cylinders. Units 2, 4, 6, and 8 contain the solution cylinders.

```
READ GEOM
UNIT 1
CYLINDER 1 1 5.748 5.3825 -5.3825
CUBOID  0 1 6.59 -5.748 6.59 -14.445 6.225 -13.54
UNIT 2
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 9.525 -9.525
CUBOID   0 1 10.16 -10.875 10.875 -10.16 10.24 -9.525
UNIT 3
CYLINDER  1 1 5.748 5.3825 -5.3825
CUBOID   0 1 6.59 -5.748 14.444 -6.59 6.225 -13.54
UNIT 4
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 9.525 -9.525
CUBOID   0 1 10.16 -10.875 10.16 -10.875 10.24 -9.525
UNIT 5
CYLINDER  1 1 5.748 5.3825 -5.3825
CUBOID   0 1 6.59 -5.748 6.59 -14.445 13.54 -6.225
UNIT 6
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 9.525 -9.525
CUBOID   0 1 10.16 -10.875 10.875 -10.16 9.525 -10.24
UNIT 7
CYLINDER  1 1 5.748 5.3825 -5.3825
CUBOID   0 1 6.59 -5.748 14.445 -6.59 13.54 -6.225
UNIT 8
CYLINDER  2 1 9.525 8.89 -8.89
CYLINDER  3 1 10.16 9.525 -9.525
CUBOID   0 1 10.16 -10.875 10.16 -10.875 9.525 -10.24
READ ARRAY  NUX=2 NUY=2 NUZ=2 FILL 6I1 8 END FILL END ARRAY
```

F11.5.7.3 *Sample Problem 13, Alternative*

This mock-up maintains the same overall unit dimensions that were used in sample problem 13, Sect. F11.D. In sample problem 13, the origin of Units 1, 2, and 3 is located at the center of the base of the uranium metal cuboid. In this mock-up, the origin of Units 1 and 2 is located at the center of the cylinder. In Unit 3, the origin is at the center of the unit.

```
READ GEOM
UNIT 1
CUBOID  1 1 0.2566 -12.4434 6.35 -6.35 3.81 -3.81
CYLINDER  0 1 13.97 3.81 -3.81
CYLINDER  1 1 19.05 3.81 -3.81
CUBOID  0 1 19.05 -19.05 19.05 -19.05 3.81 -3.81
UNIT 2
CUBOID  1 1 12.4434 -0.2566 6.35 -6.35 4.28 -4.28
CYLINDER  0 1 13.97 4.28 -4.28
CYLINDER  1 1 19.05 4.28 -4.28
CUBOID  0 1 19.05 -19.05 19.05 -19.05 4.28 -4.28
UNIT 3
CUBOID  1 1 12.4434 -0.2566 6.35 -6.35 1.308 -1.308
CUBOID  0 1 19.05 -19.05 19.05 -19.05 1.308 -1.308
END GEOM
READ ARRAY  NUX=1 NUY=1 NUZ=3  FILL 1 2 3 END ARRAY
```

F11.5.8 BIASING OR WEIGHTING DATA

Section F11.2.3 discusses the basis of weighting or biasing. The use of biasing data in reflected problems has been illustrated in Examples 9, 10, and 11 of Sect. F11.5.6. Section F11.4.7 discusses the input directions for entering biasing data.

Every geometry card requires a bias ID to associate that geometry region with a biasing or weighting function. A biasing or weighting function is a set of energy-dependent values of the average weight that are applicable in a given region. The default function for all bias IDs is constant through all energy groups, and is defined to be the default value of weight average which can be specified in the parameter data. A bias ID can be associated with a biasing function, other than default, by specifying it in the biasing input data. This function can be chosen from the weighting library or can be input from cards. Table F11.4.5 lists the materials and energy group structures for biasing functions available from the weighting library.

In general, the use of biasing should be restricted to external reflectors unless the user has generated correct biasing functions for other applications. Improper use of biasing functions can result in erroneous answers without giving any indication that they are invalid. Caution should be exercised in the generation and use of biasing functions.

Biasing functions are most applicable to thick external reflectors. Their use can significantly reduce the amount of computer time required to obtain answers in KENO V.a. If the user wishes to use a biasing function for a concrete reflector, for example, the following steps must be included in preparing the input data:

(1) The geometry region data must define the shape and dimensions of the reflector using the mixture ID for concrete and a sequence of bias ID's that associate the geometry region with the appropriate interval of the concrete weighting function. CAUTION: THE THICKNESS OF EACH REGION UTILIZING BIASING FUNCTIONS MUST MATCH OR VERY NEARLY MATCH THE INCREMENT THICKNESS OF THE WEIGHTING DATA. NO CHECK IS MADE ON THE REQUIREMENT. IT IS THE USER'S RESPONSIBILITY TO ENSURE CONSISTENCY.

(2) Biasing data must be entered. This must include the material ID for the reflector material (from Table F11.4.5 or as specified on cards) and a beginning and ending bias ID. The beginning bias ID is used to select the first set of energy-dependent average weights, and the subsequent sets of energy-dependent average weights are assigned consecutive IDs until the ending bias ID is reached.

Small deviations in reflector region thickness are allowed, such as using three generated regions with a thickness per region of 5.08 cm to generate a 15.24-cm-thick reflector of concrete, or using five generated regions with a thickness per region of 3.048 cm to generate a 15.24-cm-thick reflector of water. See Table F11.4.5 for a list of the increment thickness for each material in the weighting library. It is acceptable for the thickness of the last reflector region to be significantly different than the increment thickness. For example, a reflector card specifying five generated regions with a thickness per region of 3.0 cm could be followed by a reflector card specifying one region with a thickness per region of 0.24 cm. Assume material 2 is water and a 15.24-cm-thick cuboidal reflector of water is desired. The required reflector description and biasing data could be entered as follows:

*REFLECTOR 2 2 6*3.0 5*
*REFLECTOR 2 7 6*0.24 1*
READ BIAS ID=500 2 7 END BIAS

The same 15.24-cm-thick reflector can be described by including the extra 0.24 cm in the last region as shown below:

*REFLECTOR 2 2 6*3.0 4*
*REFLECTOR 2 6 6*3.24 1*
READ BIAS ID=500 2 6 END BIAS

Here the weighting functions associated with bias IDs 2, 3, 4, and 5 are defined by the first reflector card and each generated region has a thickness of 3.0 cm, corresponding exactly to the increment thickness for water in Table F11.4.5. Bias ID 6 is used for the last generated region which is 3.24 cm thick.

The following examples illustrate the use of biasing data. Suppose the user wishes to use the weighting function for water from Table F11.4.5 for bias IDs 2 through 6. The biasing input data would then be:

READ BIAS ID=500 2 6 END BIAS

The energy-dependent values of weight average for the first 3-cm interval of water will be used for weighting the geometry regions that specify a bias of ID of 2. The energy-dependent values of weight average for the second 3-cm interval of water will be used for geometry regions that specify a bias ID of 3, etc. Thus, the energy-dependent values of weight average for the fifth 3-cm interval of water will be used for geometry regions that specify a bias ID of 6. Geometry regions that use bias IDs other than 2, 3, 4, 5, and 6 will use the default value of weight average that is constant for all energies as a biasing function.

Several sets of biasing data can be entered at once. Assume the user wishes to use the weighting function for concrete from Table F11.4.5 for bias IDs 2 through 4 and the weighting function for water for bias IDs 5 through 7. The appropriate input data block is the following:

READ BIAS ID=301 2 4 ID=500 5 7 END BIAS

The energy-dependent values of weight average for the first 5-cm interval of concrete will be used for the geometry regions that specify a bias ID of 2, the energy-dependent values of weight average for the second 5-cm interval of concrete will be used for the geometry regions that specify a bias ID of 3, and the energy-dependent values of weight average for the third 5-cm interval of concrete will be used for the geometry regions that specify a bias ID of 4. The energy-dependent values of weight average for the first 3-cm interval of water will be used for geometry regions that specify a bias ID of 5, the values for the second 3-cm interval of water will be used for geometry regions that specify a bias ID of 6, and the values for the third 3-cm interval of water will be used for geometry regions that specify a bias ID of 7. The default value of weight average will be used for all bias IDs outside the range of 2 through 7.

If the biasing data block defines the same bias ID more than once, the value that is entered last supersedes previous entries. Assume the following data block is entered.

READ BIAS ID=400 2 7 ID=500 5 7 END BIAS

Then the data for paraffin (ID=400) will be used for bias IDs 2, 3, and 4, and the data for water (ID=500) will be used for bias IDs 5, 6, and 7.

EXAMPLE 1. Use of biasing data with a reflector card.

Assume a 5-cm-radius sphere of material 2 is reflected by a 20-cm thickness of material 1 (concrete). The concrete reflector is spherical and close fitting upon the sphere of material 2. The mixing table must specify material 1 and material 2. Material 1 must be defined as concrete. The geometry and biasing data should be entered as follows:

READ GEOM
SPHERE 2 1 5.0
REPLICATE 1 2 5.0 4
END GEOM
READ BIAS ID=301 2 5 END BIAS

In the above example, the replicate card will generate four spherical geometry regions, each 5.0 cm thick. The bias ID for the first generated region is 2; the second, 3; the third, 4; and the fourth, 5. The biasing data block specifies that the biasing function for material ID 301 (concrete) will be used from the weighting library. The bias ID to which the energy-dependent weighting function for the first 5.0-cm interval of concrete is applied is 2; the energy-dependent weighting function for the fourth 5-cm interval of concrete is applied to the fourth generated geometry region. This generated region has a bias ID of 5.

Example 1 can be described without using a reflector card as shown below. The cards that are generated by the reflector card in the previous set of data are identical to the last four spheres in this mock-up.

EXAMPLE 1. Use of biasing without a reflector card.

READ GEOM
SPHERE 2 1 5.0
SPHERE 1 2 10.0

```
SPHERE 1 3 15.0
SPHERE 1 4 20.0
SPHERE 1 5 25.0
END GEOM
READ BIAS  ID=301 2 5 END BIAS
```

F11.5.9 CHARACTER OR COLOR PLOTS

Plots are generated only if a plot data block has been entered for the problem and **PLT=NO** has not been entered in the parameter data or the plot data. Character plots are generated only if **SCR=NO** is entered in the plot data. See Sect. F11.4.11 for a description of plot data. When a plot is to be made, the user MUST correctly specify the upper left-hand corner of the plot with respect to the origin of the plot. The origin of a plot is defined as follows:

(1) SINGLE UNIT – The origin of the plot coincides with the origin of the geometry description.

(2) UNREFLECTED ARRAY – This is an array problem that does not have a global unit. The origin of the plot is located at the most negative point of the global array.

(3) REFLECTED ARRAY – The origin of the plot coincides with the origin of the ARRAY description in the EXTENDED GEOMETRY DESCRIPTION of the global unit.

Plots can represent mixture numbers, unit numbers or bias ID numbers. A title can be entered for each plot. If plot titles are omitted, the title of the KENO V.a case will be printed for each plot title until a plot title is entered. If a plot title is entered and a subsequent plot title is omitted, the last plot title prior to the omitted one will be used for the omitted one.

The upper-left and lower-right coordinates define the area (i.e., the slice and its location) for which the plot is to be made. The direction cosines across the plot and the direction cosines down the plot define the direction of the vector across the plot and the vector down the plot with respect to the geometry coordinate system. One of the simplest ways of generating a plot is to specify the desired coordinates of the upper-left and lower-right corners of the plot. Determine which plot axis is to be across the plot and which is to be down. The sign of the direction cosine should be consistent with the direction of that component when moving from the upper-left to lower-right corner. For example, to draw a plot of an x-z slice at y = 5.0 with x across the plot and z down the plot for a system whose x coordinates ranges from 0.0 to 10.0 and whose z coordinates range from 0.0 to 20.0, the upper-left coordinate could be XUL=0.0 YUL=5.0 ZUL=20.0 and the lower-right coordinates could be XLR=10.0 YLR=5.0 ZLR=0.0. Since x is to be plotted across the plot with x = 0.0 at the left and x = 10.0 at the right, only the x component of the direction cosines across the plot need be entered. It should be positive because going from 0.0 to 10.0 is moving in the positive direction. Thus, UAX=1.0 would be entered for the direction cosines across the plot. VAX and WAX could be omitted. Z is to be plotted down the plot with z = 20.0 at the top and z = 0.0 at the bottom. Therefore, only the z component of the direction cosines down the plot need to be defined. It should be negative because moving from 20.0 to 0.0 is moving in the negative direction. Thus, WDN = -1.0 would be entered for the direction cosines down the plot. UDN and VDN could be omitted. The sign of the direction cosines should be consistent with the coordinates of the upper-left and lower-right corners in order to get a plot.

It is not necessary that the plot be made for a slice orthogonal to one of the axes. Plots can be made of slices cut at any desired angle, but the user should exercise caution and be well aware of the distortion of shapes that can be introduced. (Nonorthogonal slices through cylinders plot as ellipses.)

The user can specify the horizontal and vertical spacing between points on the plot. It is usually advisable to enter one or the other. Entering both of them can cause distortion of the plot. DLX= is used to specify the horizontal spacing between points and DLD= is used to specify the vertical spacing between points. When only one of them is specified, the code calculates the correct value of the other such that the plot will not be distorted. In some instances, it can be desirable to distort a plot by entering both DLX and DLD. It might be desirable to compress one-dimension relative to another. If DLX=0.5 and DLD=5.0 are entered as data for a printer that prints 10 characters per inch across and 8 lines per inch down, the portion of the plot that is printed in the vertical direction will be reduced by a factor of 8 relative to the portion printed in the horizontal direction ((5.0*8)/(0.5*10)). That is, if the coordinates specify a perfect square, the plot will be a rectangle that is about 8 times as wide as it is tall. Color plots plot LPI pixels across per 10 pixels down. The portion of the plot that is printed in the vertical direction will be reduced by a factor of 10 relative to the portion plotted in the horizontal direction ((5.0*LPI)/(0.5*10)).

DLX or DLD can be specified by the user to be small enough to show the desirable detail in the plot. The plot is generated by starting at the upper left corner of the plot and generating a point every DLX across the plot; then moving down DLD and repeating the generation of the points across the plot.

NAX specifies the number of intervals that will be printed across the plot. It may be convenient for the user to specify the number of characters that can be printed across the plot (page) on the printer that will be used. Larger plots can be created by specifying multiples of this number. In that case, the plot must be taped together to see the overall plot. The plot will print one page wide and full length. Then the next page width and full length will be printed, etc., until the entire plot is completed.

NDN specifies the number of intervals that will be printed down the plot. If both NAX and NDN are entered, the plot may be distorted. If one of them is entered, the value of the other will be calculated so the plot will not be distorted.

LPI, for a character plot, is the number of lines per inch that are printed on a page. It should be entered to be consistent with the printer that will be used. The default value is 8 lines per inch. For a color plot, LPI is the number of pixels down per 10 pixels across. The default value is 10. LPI need be entered only once for a problem. It should be entered in the data for the first plot, so all the plots will be printed in the same manner.

When a plot is being made, the first character represents the coordinates of the upper-left corner. The value of DELV is added to the coordinate that is to be printed across the plot, and the next character is printed. DELV is added to that value to determine the location of the next character, that is, a point is determined every DELV across the plot and a character is printed for each point. When a line has been completed, a new line is begun DELU from the first line. This procedure is repeated until the plot is complete.

EXAMPLE 1. SINGLE UNIT WITH CENTERED ORIGIN

Consider two concentric cylinders in a cuboid. The inner cylinder is 5.2 cm in diameter. The outer cylinder has an inside diameter of 7.2 cm and an outside diameter of 7.6 cm. Both cylinders are 30 cm high. They are contained in a tight-fitting box whose wall thickness is 0.5 cm and whose top and bottom are each 1.0 cm thick. The inner cylinder is composed of mixture 1, the outer cylinder is made of mixture 4, and the box is made of mixture 2. The problem can be described with its origin at the center of the inner cylinder. The problem description for this arrangement is shown below:

```
=KENO5
SINGLE UNIT CONCENTRIC CYLINDERS IN CUBOID WITH ORIGIN AT CENTER
READ PARAM  RUN=NO LIB=41 END PARAM
READ MIXT  SCT=1  MIX=1 92500 4.7048-2 MIX=2 200 1.0 MIX=3 502 0.1
MIX=4 200 1.0
END MIXT
READ GEOM
UNIT 1
CYLINDER 1 1 2.6 2P15.0
CYLINDER 0 1 3.6 2P15.0
CYLINDER 4 1 3.8 2P15.0
CUBOID 0 1 4P3.8 2P15.0
CUBOID 2 1 4P4.3 2P16.0
END GEOM
```

CHARACTER PLOT DATA DESCRIPTION

```
READ PLOT   SCR=NO
TTL='X-Y SLICE AT Z MIDPOINT.  SINGLE UNIT CONCENTRIC CYLS.'
XUL=-4.6 YUL=4.6 ZUL=0.0 XLR=4.6 YLR=-4.6 ZLR=0.0
UAX=1.0 VDN=-1.0 NAX=130 NCH=' *-.X' END
PIC=UNIT NCH='01' END
END PLOT
END DATA
END
```

COLOR PLOT DATA DESCRIPTION

```
READ PLOT
TTL='X-Y SLICE AT Z MIDPOINT.  SINGLE UNIT CONCENTRIC CYLS'
XUL=-4.6 YUL=4.6   ZUL=0.0  XLR=4.6  YLR=-4.6   ZLR=0.0
UAX=1.0   VDN=-1.0   NAX=640   END
PIC=UNIT   END
END PLOT
END DATA
END
```

The plot data blocks included above are set up to draw a mixture map of an x-y slice taken at the half height (z=0.0) and a unit map for the same slice. In the above examples, the geometry dimensions extend from x = -4.3 to x = 4.3, from y = -4.3, to y = 4.3, and from z = -8.0 to z = 8.0. An x-y slice is be printed at the half-height (z = 0.0). The desired plot data sets the upper left-hand corner of the plot to be x = -4.6 and y = 4.6. The lower right-hand corner of the plot is specified as x = 4.6 and y = -4.6. These data are entered by specifying the upper left-hand corner as XUL=-4.6 YUL=4.6 ZUL=0.0 and the lower right-hand corner as XLR=4.6 YLR=-4.6 ZLR=0.0. It is desired to print x across the plot and y down the plot. Therefore, the x direction cosine is specified across the plot, in the direction from x = -4.6 to x = 4.6 as UAX=1.0. The y direction cosine is specified down the plot, from y = 4.6 to y = -4.6 as VDN=-1.0.

For a character plot, question marks will be printed for points outside the range of the problem geometry description. By setting the plot dimension slightly larger than the geometry dimension, a border of questions marks will be printed around the specified plot. This verifies that the outer boundaries of the geometry are contained within the plot dimensions. It was desirable for the character plot to be one page wide (130 characters) so the number of characters across the page was specified as NAX=130. An arbitrary choice was made to print a blank for a void, a * for mixture 1, a - for mixture 2, and a . for mixture 4. Mixture 3 was not used in the problem, so a character did not have to be entered for it in the character string. Thus, a character string of NCH=’ *-.’ would have been sufficient but a string of NCH=’ *-.X’ was entered. Since only three mixtures were used, only the first four characters were utilized. The blank represents a void, the * represents the smallest mixture number used in the problem (mixture 1), the - represents the next smallest mixture number used in the geometry description (mixture 2) and the . represents the largest mixture number used in the geometry (mixture 4). The resultant character plots and associated data are shown in Figs. F11.5.42 and F11.5.43. A second character plot covering the same area shows a unit map rather than a mixture map. This unit map and associated data are shown in Figs. F11.5.44 and F11.5.45.

For a color plot, a black border will be printed for points outside the range of the problem geometry description. By setting the plot dimension slightly larger than the geometry dimension, a black border will be printed around the specified plot. This verifies that the outer boundaries of the geometry are contained within the plot dimensions. NAX is the number of pixels across for a color plot. The size of the computer screen will determine the maximum number of pixels to be viewed. A recommended range for NAX is between 400 and 640 pixels. NCH is not used in color plots. The resultant color plots and associated data are shown in Figs. F11.5.42-a and F11.5.43-a. A second color plot covering the same area shows a unit map rather than a mixture map. This unit map and associated data are shown in Figs. F11.5.44-a and F11.5.45-a.

```
x-y slice at z midpoint.  single unit concentric cyls.

                                   mixture map

mixture   0 1 2 4
 symbol     * - .
     upper  left          lower right
     coordinates          coordinates
x    -4.6000e+00           4.6000e+00
y     4.6000e+00          -4.6000e+00
z     0.0000e+00           0.0000e+00
         u axis       v axis
         (down)       (across)
x          .00000      1.00000
y        -1.00000       .00000
z          .00000       .00000
nu=  86    nv= 130    delu= 1.0615e-01    delv= 7.0769e-02
```

Figure F11.5.42 Associated data for single unit mixture map

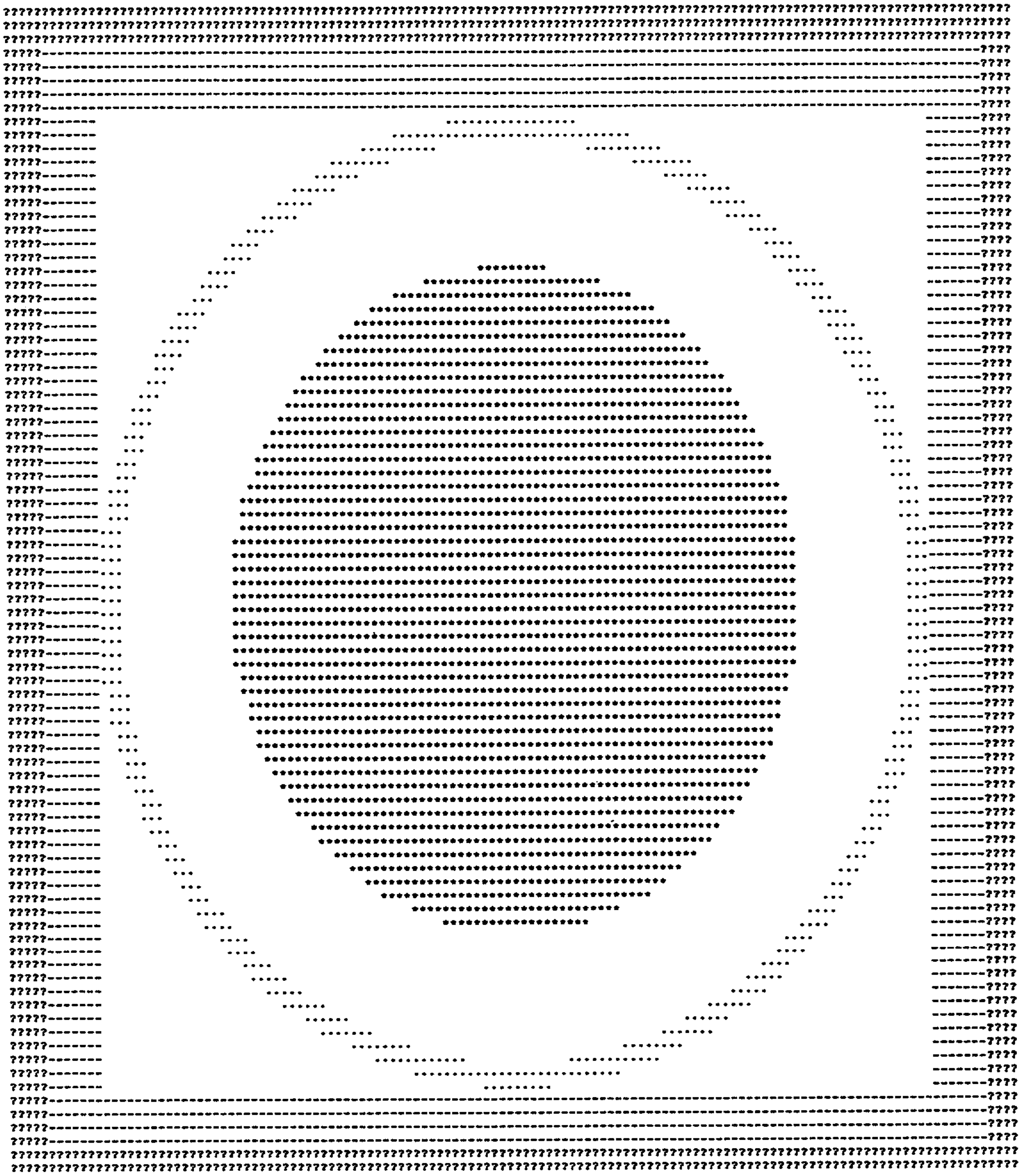


Figure F11.5.43 Mixture map character plot of single unit with centered origin

```
x-y slice at z midpoint.  single unit concentric cyls.
                                             unit map
   unit  1
 symbol  1
      upper  left          lower right
      coordinates          coordinates
x     -4.6000e+00           4.6000e+00
y      4.6000e+00          -4.6000e+00
z      0.0000e+00           0.0000e+00
          u axis      v axis
          (down)      (across)
x           .00000     1.00000
y         -1.00000      .00000
z           .00000      .00000
nu=   86   nv=  130     delu= 1.0698e-01     delv= 7.0769e-02
```

Figure F11.5.44 Associated data for single unit map

```
??????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????
??????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????
??????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
?????111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111????
??????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????
??????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????
```

Figure F11.5.45 Unit map character plot for single unit with centered origin

Figure F11.5.45 shows a block of 1's surrounded by a border of ?'s. This indicates that the entire slice specified in the plot data was part of Unit 1. For this problem the entire volume is Unit 1.

```
                                    mixture map

mixture   0 1 2 4
 symbol     1 2 3
      upper  left           lower right
      coordinates           coordinates
x     -4.6000e+00            4.6000e+00
y      4.6000e+00           -4.6000e+00
z      0.0000e+00            0.0000e+00

          u axis      v axis
          (down)      (across)
x         0.00000      1.00000
y        -1.00000      0.00000
z         0.00000      0.00000

nu=  640   nv=  640     delu= 1.4375e-02     delv= 1.4375e-02     lpi=  10.000
```

Figure F11.5.46 Associated data for single unit mixture map

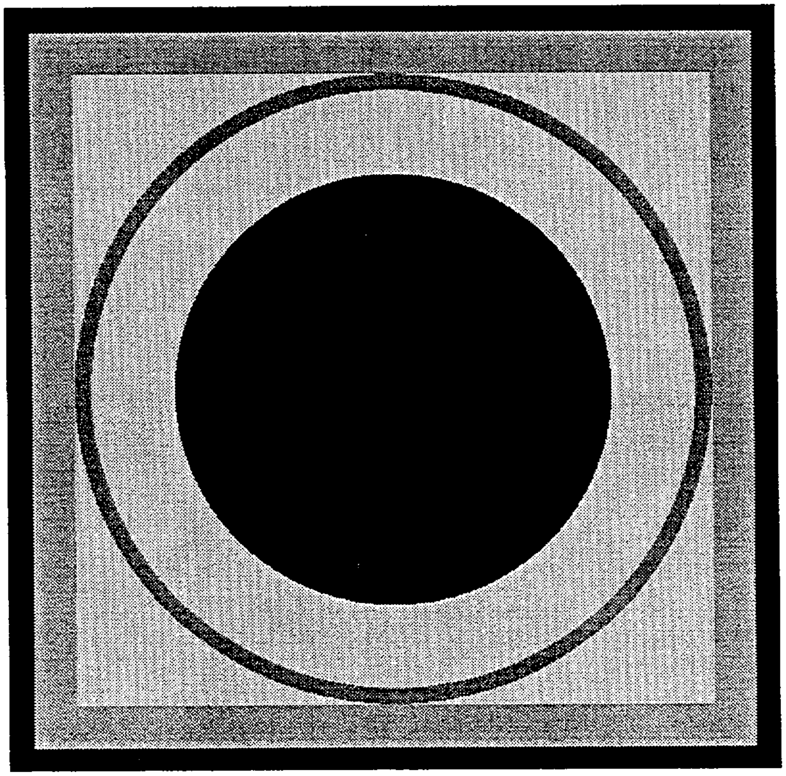


Figure F11.5.47 Mixture map color plot for single unit with centered origin

```
x-y slice at z midpoint. single unit concentric cyls

                                                    unit map

   unit  1
  symbol  1
      upper  left          lower right
      coordinates          coordinates
x     -4.6000e+00           4.6000e+00
y      4.6000e+00          -4.6000e+00
z      0.0000e+00           0.0000e+00
          u axis       v axis
          (down)       (across)
x         0.00000       1.00000
y        -1.00000       0.00000
z         0.00000       0.00000
nu=  640   nv=  640     delu= 1.4375e-02     delv= 1.4375e-02     lpi=  10.000
```

Figure F11.5.48 Associated data for single unit map

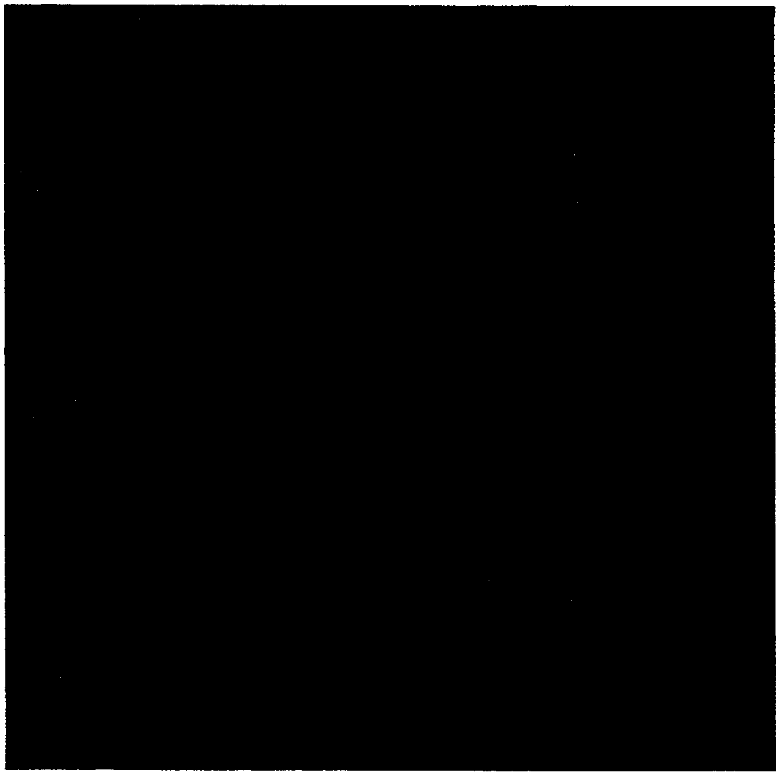


Figure F11.5.49 Unit map color plot for single unit with centered origin

EXAMPLE 2. SINGLE UNIT WITH OFFSET ORIGIN

The physical problem is the same as that described in Example 1: two concentric cylinders in a cuboid. The dimensions are exactly the same. The difference is in the choice of the origin. In this geometry description, the origin was specified as the most negative point of the unit. Thus, the cylinders have to have an origin specified to center them in the cuboid and the cuboid extends from 0.0 to 8.6 in x and y and from 0.0 to 32 in z as shown in the problem description below.

```
=KENO5
SINGLE UNIT CONCENTRIC CYLINDERS IN CUBOID WITH ORIGIN AT CORNER
READ PARAM  RUN=NO LIB=41 END PARAM
READ MIXT  SCT=1  MIX=1 92500 4.7048-2 MIX=2 200 1.0 MIX=3 502 0.1
MIX=4 200 1.0
END MIXT
READ GEOM
UNIT 1
CYLINDER 1 1 2.6 31.0 1.0 ORIGIN 4.3 4.3
CYLINDER 0 1 3.6 31.0 1.0 ORIGIN 4.3 4.3
CYLINDER 4 1 3.8 31.0 1.0 ORIGIN 4.3 4.3
CUBOID 0 1 8.1 0.5 8.1 0.5 31.0 1.0
CUBOID 2 1 8.6 0.0 8.6 0.0 32.0 0.0
END GEOM
READ PLOT
TTL='X-Y SLICE AT Z MIDPOINT.  SINGLE UNIT CONCENTRIC CYLS.'
XUL=-0.3 YUL=8.9 ZUL=16.0 XLR=8.9 YLR=-0.3 ZLR=16.0
UAX=1.0 VDN=-1.0 NAX=640 END
PIC=UNIT  END
END PLOT
END DATA
END
```

The plot data included above will draw a mixture map of an x-y slice taken at the half-height (z = 16.0). It will also draw a unit map of the same slice. The plot dimensions extend 0.3 cm beyond the problem dimensions to provide a black border around the plot. The associated plot data specification for the mixture map is shown in Fig. F11.5.50, the mixture map is shown in Fig. F11.5.51, and the associated plot data for the unit map is shown in Fig. F11.5.52. The unit map is identical to Fig. F11.5.45 and is not included. Note that Fig. F11.5.51 is identical to Fig. F11.5.43.

```
                 x-y slice at z midpoint.  single unit concentric cyls.

                                      mixture map

mixture  0 1 2 4
 symbol    1 2 3
      upper  left          lower right
      coordinates          coordinates
x      -3.0000e-01          8.9000e+00
y       8.9000e+00         -3.0000e-01
z       1.6000e+01          1.6000e+01
          u axis      v axis
          (down)     (across)
x         0.00000     1.00000
y        -1.00000     0.00000
z         0.00000     0.00000
nu=  640   nv=  640     delu= 1.4375e-02      delv= 1.4375e-02      lpi=  10.000
```

Figure F11.5.50 Associated data for mixture map of single unit with offset origin

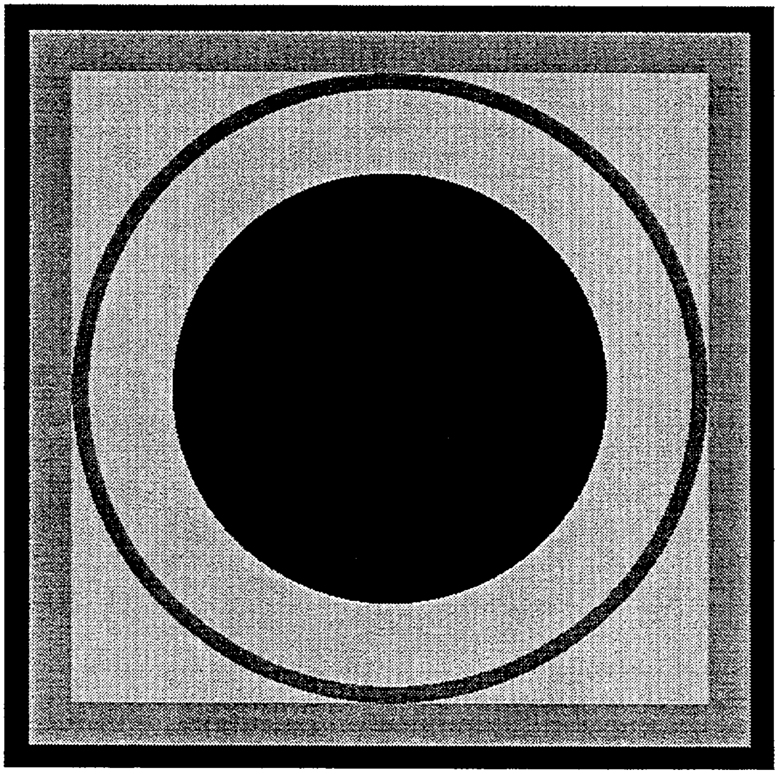


Figure F11.5.51 Mixture map of single unit with offset origin

```
                    x-y slice at z midpoint.  single unit concentric cyls.

                                                   unit map

   unit  1
 symbol  1
       upper  left          lower right
       coordinates          coordinates
x      -3.0000e-01           8.9000e+00
y       8.9000e+00          -3.0000e-01
z       1.6000e+01           1.6000e+01
            u axis      v axis
            (down)     (across)
x          0.00000      1.00000
y         -1.00000      0.00000
z          0.00000      0.00000
nu=  640   nv=  640     delu= 1.4375e-02     delv= 1.4375e-02     lpi=  10.000
```

Figure F11.5.52 Associated data for unit map of single unit with offset origin

EXAMPLE 3. A 2×2×2 UNREFLECTED ARRAY OF CONCENTRIC CYLINDERS IN CUBOIDS

The physical representation of this example is a 2 × 2 × 2 array of the configuration described in Example 1 of this section. The input data description for this array is given below:

```
2x2x2 BARE ARRAY OF CONCENTRIC CYLINDERS IN CUBOID
READ PARAM  RUN=NO LIB=41 END PARAM
READ MIXT  SCT=1  MIX=1 92500 4.7048-2 MIX=2 200 1.0 MIX=3 502 0.1
MIX=4 200 1.0
END MIXT
READ GEOM
UNIT 1
CYLINDER 1 1 2.6 2P15.0
CYLINDER 0 1 3.6 2P15.0
CYLINDER 4 1 3.8 2P15.0
CUBOID 0 1 4P3.8 2P15.0
CUBOID 2 1 4P4.3 2P16.0
END GEOM
READ ARRAY NUX=2 NUY=2 NUZ=2 END ARRAY
READ PLOT
TTL='X-Y SLICE AT HALF HEIGHT OF BOTTOM LAYER.'
XUL=-0.3 YUL=17.5 ZUL=16.0 XLR=17.5 YLR=-0.3 ZLR=16.0
UAX=1.0 VDN=-1.0 NAX=640  END
TTL='X-Z SLICE THROUGH FRONT ROW, Y=12.9.'
XUL=-1.0 YUL=12.9 ZUL=65.0 XLR=18.2 YLR=12.9 ZLR=-1.0
UAX=1.0 WDN=-1.0 NAX=320 END
END PLOT
END DATA
END
```

As stated at the beginning of Sect. F11.5.9, the origin of the plot is located at the most negative point of the array. Each individual unit in the array is 8.6 cm wide in x and y and is 32 cm high in z. Since the array has two units stacked in each direction, the array is 17.2 cm wide in x and y and is 64 cm high. Therefore, the array exists from x = 0.0 to x = 17.2, from y = 0.0 to y = 17.2 and from z = 0.0 to z = 64.0.

The first color plot is to generate an x-y slice through the array at the half height of the first layer as shown in Fig. F11.5.53. This occurs at z = 16.0 cm. It is desirable to define the outer boundaries of the array. This is achieved by setting the boundaries of the plot larger than the array. In this case, the boundaries were arbitrarily set 0.3 cm larger than the array, resulting in a black border around the array. If the plot were to exclude everything external to the array, the following coordinates could have been entered XUL=0.0 YUL=17.2 ZUL=16.0 XLR=17.2 YLR=0.0 ZLR=16.0. This would have eliminated the black border. The existing plot was made using XUL=-0.3 YUL=17.5 ZUL=16.0 XLR=17.5 YLR=-0.3 ZLR=16.0.

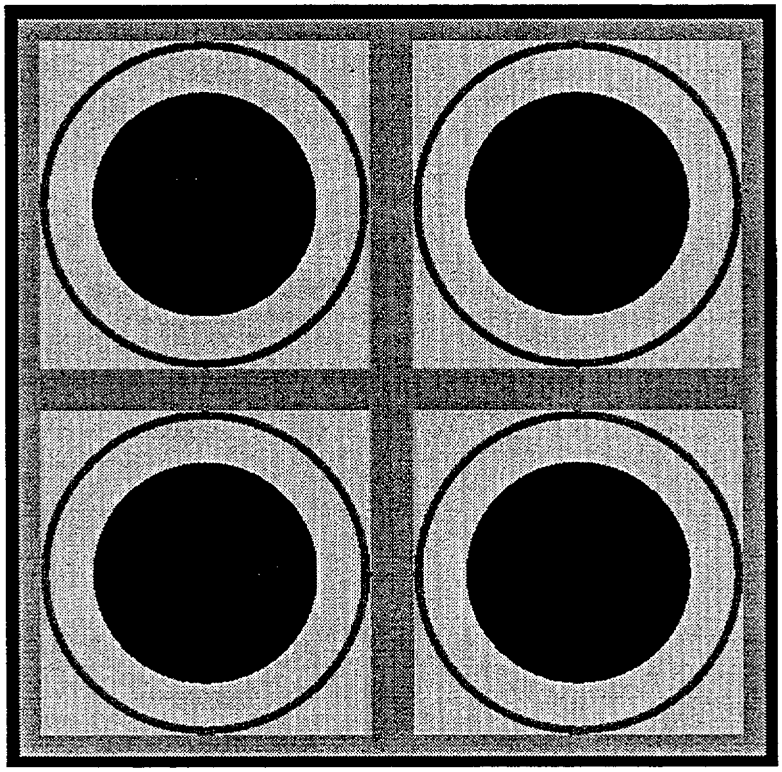


Figure F11.5.53 x-y plot of 2 × 2 × 2 bare array

The second color plot is to generate an x-z slice through the center of the front row of the array. In order to obtain a black border, the coordinates of x and z were arbitrarily set 1.0 cm larger than the boundaries of the array. The center of the front row occurs at y = 12.9. The coordinates of the plot were: XUL=-1.0 ZUL=65.0 YUL=12.9 XLR=18.2 ZLR=-1.0 YLR=12.9. The resultant mixture map is shown in Fig. F11.5.54.

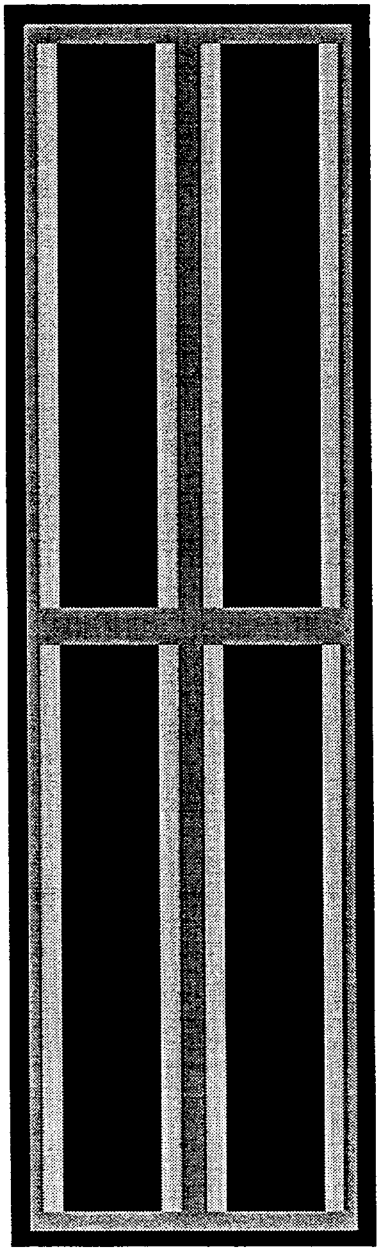


Figure F11.5.54 x-z plot of 2 × 2 × 2 bare array

EXAMPLE 4. A 2 × 2 × 2 REFLECTED ARRAY WITH THE ORIGIN AT THE MOST NEGATIVE POINT OF THE ARRAY

The array is the array described in Example 3 of this section with a 6-in. concrete reflector on all faces. The input data description for this array is given below.

```
2x2x2 REFLECTED ARRAY OF CONCENTRIC CYLINDERS IN CUBOID
READ PARAM  RUN=NO LIB=41 END PARAM
READ MIXT  SCT=1  MIX=1 92500 4.7048-2 MIX=2 200 1.0 MIX=3 301 1.0
MIX=4 200 1.0
END MIXT
READ GEOM
UNIT 1
CYLINDER 1 1 2.6 2P15.0
CYLINDER 0 1 3.6 2P15.0
CYLINDER 4 1 3.8 2P15.0
CUBOID 0 1 4P3.8 2P15.0
CUBOID 2 1 4P4.3 2P16.0
GLOBAL
UNIT 2
ARRAY 1  3*0.0
REFLECTOR 3 2 6*5.0 3
REFLECTOR 3 5 6*0.24 1
END GEOM
READ BIAS ID=301 2 5 END BIAS
READ ARRAY ARA=1 NUX=2 NUY=2 NUZ=2 FILL F1 END FILL END ARRAY
READ PLOT
TTL='X-Y SLICE AT HALF HEIGHT OF BOTTOM LAYER.INCLUDES REFL.'
XUL=-16.24 YUL=33.44 ZUL=16.0 XLR=33.44 YLR=-16.24 ZLR=16.0
UAX=1.0 VDN=-1.0 NAX=640  END
TTL='X-Y SLICE AT HALF HEIGHT OF BOTTOM LAYER, INCLUDE 3 CM OF REFL.'
XUL=-3.0 YUL=20.2 ZUL=16.0 XLR=20.2 YLR=-3.0 ZLR=16.0
UAX=1.0 VDN=-1.0 NAX=640 END
TTL='X-Z SLICE THROUGH FRONT ROW, Y=12.9. INCLUDE REFLECTOR'
XUL=-16.24 YUL=12.9 ZUL=80.24 XLR=33.44 YLR=12.9 ZLR=-16.24
UAX=1.0 WDN=-1.0 NAX=640 END
TTL='X-Z SLICE THROUGH FRONT ROW, Y=12.9. INCLUDE 3 CM OF REFLECTOR'
XUL=-3.0 YUL=12.9 ZUL=67.0 XLR=20.2 YLR=12.9 ZLR=-3.0
UAX=1.0 WDN=-1.0 NAX=640 END
END PLOT
END DATA
END
```

The ARRAY record specifies the array number and the coordinates of the most negative point of the array to be (0.0,0.0,0.0). Thus the reflected array extends from - 15.24 cm to +32.44 cm in x and y and from - 15.24 to +79.24 in z.

The first color plot for this example is to show an x-y slice through the array and reflector at the half-height of the bottom layer. A black border is used to verify that the entire reflector has been shown. This is accomplished by arbitrarily setting the plot boundaries 1 cm beyond the reflector boundaries. The coordinates used for this plot are: XUL = - YUL = 33.44 ZUL = 16.0 XLR = 33.44 YL = -16.24 ZLR = 16.0. The plot data description is shown in Fig. F11.5.55, and the plot is shown in Fig. F11.5.56.

```
                      x-y slice at half height of bottom layer.includes refl.

                                           mixture map

mixture   0 1 2 3 4
 symbol     1 2 3 4
      upper  left          lower right
      coordinates          coordinates
x     -1.6240e+01           3.3440e+01
y      3.3440e+01          -1.6240e+01
z      1.6000e+01           1.6000e+01
          u axis      v axis
          (down)     (across)
x         0.00000     1.00000
y        -1.00000     0.00000
z         0.00000     0.00000
nu=  640   nv=  640    delu= 7.7625e-02     delv= 7.7625e-02     lpi=  10.000
```

Figure F11.5.55 Plot data for x-y slice of example 4

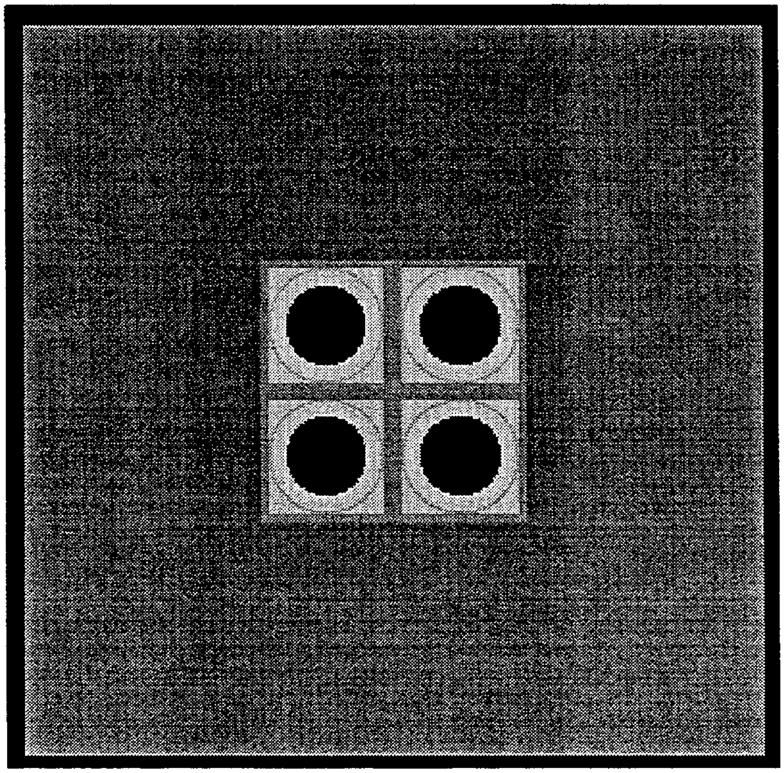


Figure F11.5.56 x-y plot of 2 × 2 × 2 reflected array

The next color plot is the same as the previous plot except the plot includes only the first 3 cm of the reflector. This results in the plot being large enough to show more detail. The coordinates used for this plot are: XUL=-3.0 YUL=20.2 ZUL=16.0 XLR=20.2 YLR=-3.0 ZLR=16.0. This plot data description is given in Fig. F11.5.57, and the plot is shown in Fig. F11.5.58.

```
                 x-y slice at half height of bottom layer, include 3 cm of refl.
                                          mixture map

mixture   0 1 2 3 4
 symbol     1 2 3 4
      upper  left            lower right
      coordinates            coordinates
x     -3.0000e+00             2.0200e+01
y      2.0200e+01            -3.0000e+00
z      1.6000e+01             1.6000e+01
          u axis        v axis
          (down)        (across)
x        0.00000        1.00000
y       -1.00000        0.00000
z        0.00000        0.00000
nu=  640   nv=  640     delu= 3.6250e-02     delv= 3.6250e-02     lpi=  10.000
```

Figure F11.5.57 Plot data for enlarged x-y slice of example 4

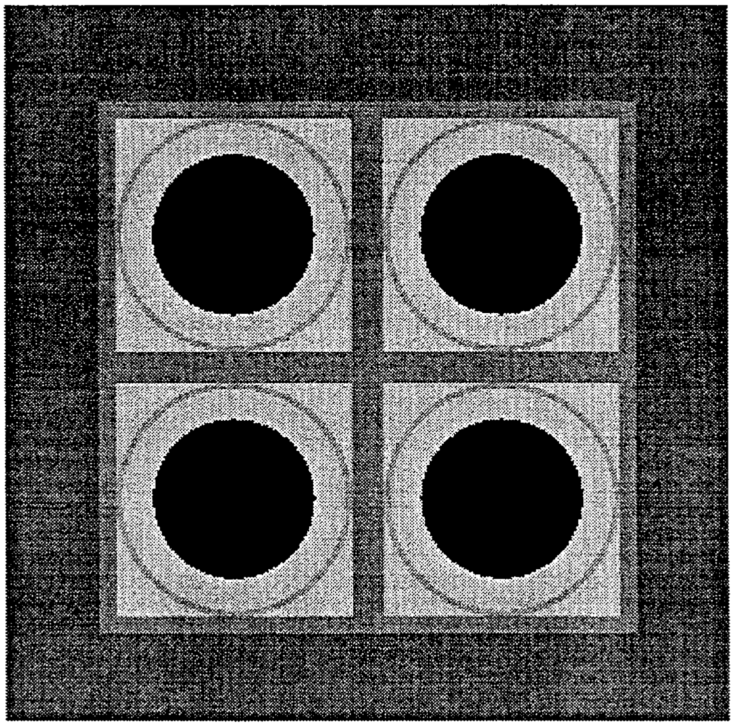


Figure F11.5.58 Enlarged x-y plot of 2 × 2 × 2 reflected array.

The third color plot for this example is an x-z slice through the center of the front row. An extra 1 cm is included in the coordinates to provide a black border around the plot. The coordinates are: XUL=-16.24 YUL=12.9 ZUL=80.24 XLR=33.44 YLR=12.9 ZLR=-16.24. The resultant plot data and plot are shown in Figs. F11.5.59 and F11.5.60.

```
                  x-z slice through front row, y=12.9. include reflector

                                      mixture map

mixture   0 1 2 3 4
 symbol     1 2 3 4
      upper  left           lower right
      coordinates           coordinates
x     -1.6240e+01            3.3440e+01
y      1.2900e+01            1.2900e+01
z      8.0240e+01           -1.6240e+01
          u axis       v axis
          (down)       (across)
x         0.00000       1.00000
y         0.00000       0.00000
z        -1.00000       0.00000
nu= 1242   nv=   640     delu= 7.7625e-02     delv= 7.7625e-02      lpi=  10.000
```

Figure F11.5.59 Plot data for x-z slice of example 4

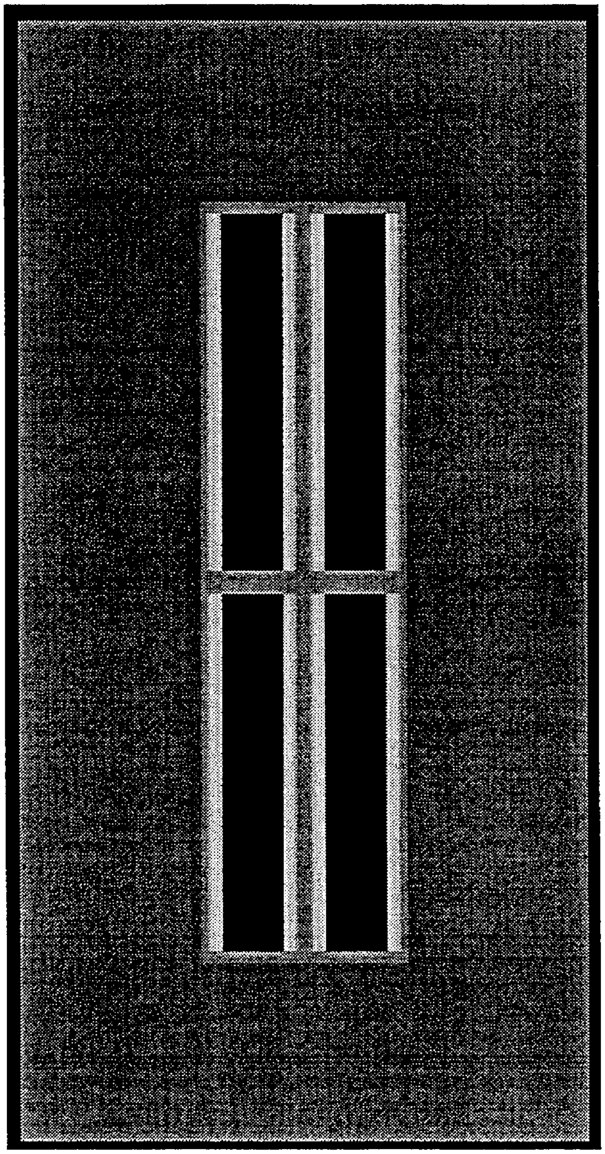
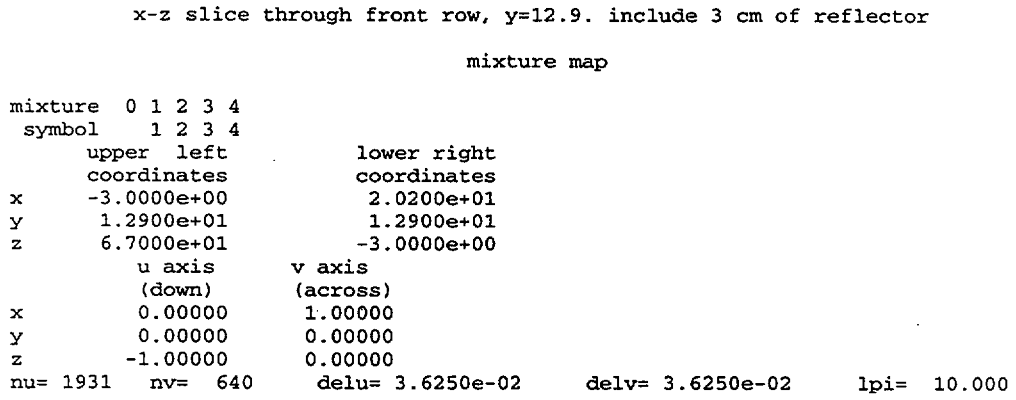


Figure F11.5.60 x-z plot for 2 × 2 × 2 reflected array

The last color plot for this example is the same as the previous one, except only 3 cm of the reflector is included in the plot. The plot data and associated plot are shown in Figs. F11.5.61 and F11.5.62.



```
                x-z slice through front row, y=12.9. include 3 cm of reflector

                                         mixture map

mixture  0 1 2 3 4
 symbol    1 2 3 4
      upper  left            lower right
      coordinates            coordinates
x     -3.0000e+00             2.0200e+01
y       1.2900e+01             1.2900e+01
z       6.7000e+01            -3.0000e+00
          u axis       v axis
          (down)      (across)
x         0.00000      1.00000
y         0.00000      0.00000
z        -1.00000      0.00000
nu= 1931   nv=  640    delu= 3.6250e-02     delv= 3.6250e-02     lpi=  10.000
```

Figure F11.5.61 Plot data for enlarged x-z slice of example 4

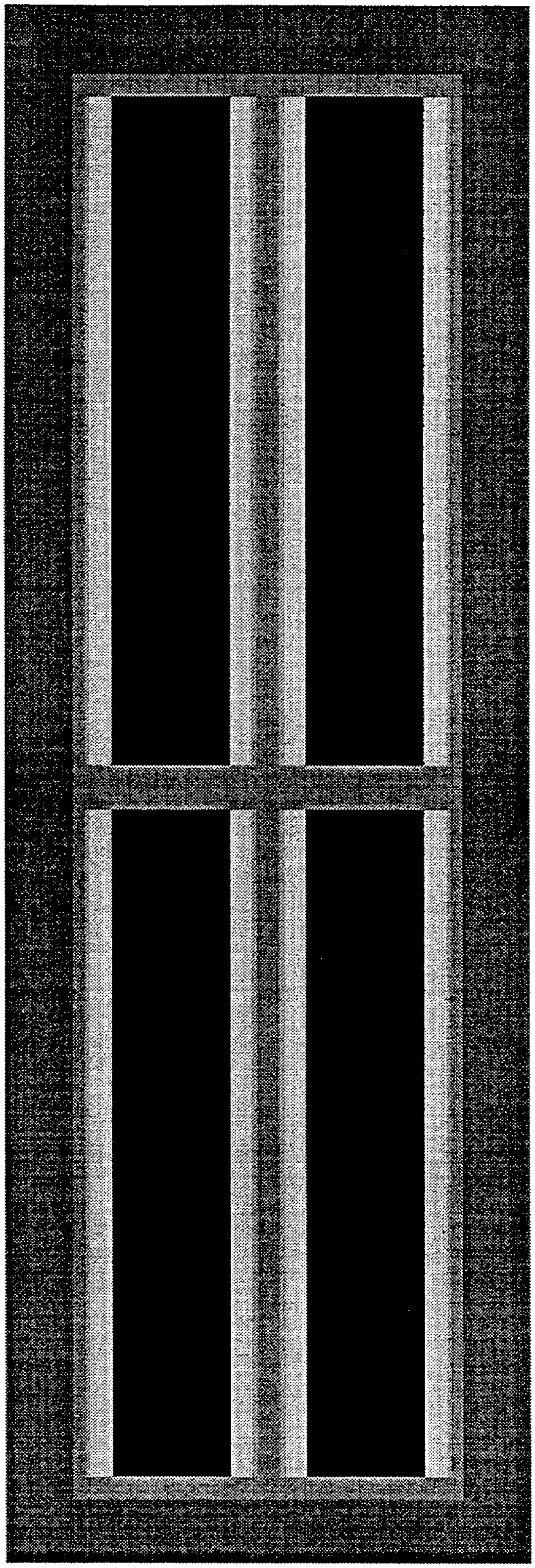


Figure F11.5.62 Enlarged x-z plot of 2 × 2 × 2 reflected array

EXAMPLE 5. A 2 × 2 × 2 REFLECTED ARRAY WITH THE ORIGIN CENTERED IN THE ARRAY

This example is physically identical to Example 4. The difference is in the specification of the origin. The bare array is 17.2 cm wide in x and y and 64 cm high. The origin (0,0,0) can be placed at the exact center of the array by specifying the most negative point of the array as x = -8.6, y = -8.6 and z = -32.0. This is done using the ARRAY description. Because the origin is located at a different position, the coordinates of the plots will also be different. The input data description for this example is given below.

```
2x2x2 REFLECTED ARRAY OF CONCENTRIC CYLINDERS IN CUBOID
READ PARAM  RUN=NO LIB=41 TME=0.5 END PARAM
READ MIXT  SCT=1  MIX=1 92500 4.7048-2 MIX=2 200 1.0 MIX=3 301 1.0
MIX=4 200 1.0
END MIXT
READ GEOM
UNIT 1
CYLINDER 1 1 2.6 2P15.0
CYLINDER 0 1 3.6 2P15.0
CYLINDER 4 1 3.8 2P15.0
CUBOID 0 1 4P3.8 2P15.0
CUBOID 2 1 4P4.3 2P16.0
GLOBAL
UNIT 2
ARRAY 1 1 2*-8.6 -32.0
REFLECTOR 3 2 6*5.0 3
REFLECTOR 3 5 6*0.24 1
END GEOM
READ BIAS ID=301 2 5 END BIAS
READ ARRAY ARA=1 NUX=2 NUY=2 NUZ=2 FILL F1 END FILL  END ARRAY
READ PLOT
TTL=’X-Y SLICE AT HALF HEIGHT OF BOTTOM LAYER.INCLUDES REFL.’
XUL=-24.84 YUL=24.84 ZUL=-8.0 XLR=24.84 YLR=-24.84 ZLR=-8.0
UAX=1.0 VDN=-1.0 NAX=640  END
TTL=’X-Y SLICE AT HALF HEIGHT OF BOTTOM LAYER, INCLUDE 3 CM OF REFL.’
XUL=-11.6 YUL=11.6 ZUL=-8.0 XLR=11.6 YLR=-11.6 ZLR=-8.0
UAX=1.0 VDN=-1.0 NAX=640  END
TTL=’X-Z SLICE THROUGH FRONT ROW. Y=4.3  INCLUDE REFLECTOR’
XUL=-24.84 YUL=4.3 ZUL=48.24 XLR=24.84 YLR=4.3 ZLR=-48.24
UAX=1.0 WDN=-1.0 NAX=640 END
TTL=’X-Z SLICE THROUGH FRONT ROW, Y=4.3  INCLUDE 3 CM OF REFLECTOR’
XUL=-11.6 YUL=4.3 ZUL=35.0 XLR=11.6 YLR=4.3 ZLR=-35.0
UAX=1.0 WDN=-1.0 NAX=640 END
END PLOT
END DATA
END
```

The first color plot for this example covers identically the same area as the first plot for Example 4. The plot data for this plot and the actual plot are given in Figs. F11.5.63 and F11.5.64.

```
                    x-y slice at half height of bottom layer.includes refl.
                                         mixture map

mixture  0 1 2 3 4
 symbol    1 2 3 4
      upper  left          lower right
      coordinates          coordinates
x     -2.4840e+01           2.4840e+01
y      2.4840e+01          -2.4840e+01
z     -8.0000e+00          -8.0000e+00
          u axis      v axis
          (down)      (across)
x         0.00000      1.00000
y        -1.00000      0.00000
z         0.00000      0.00000
nu=  640   nv=  640    delu= 7.7625e-02     delv= 7.7625e-02     lpi=  10.000
```

Figure F11.5.63 Plot data for x-y slice of example 5

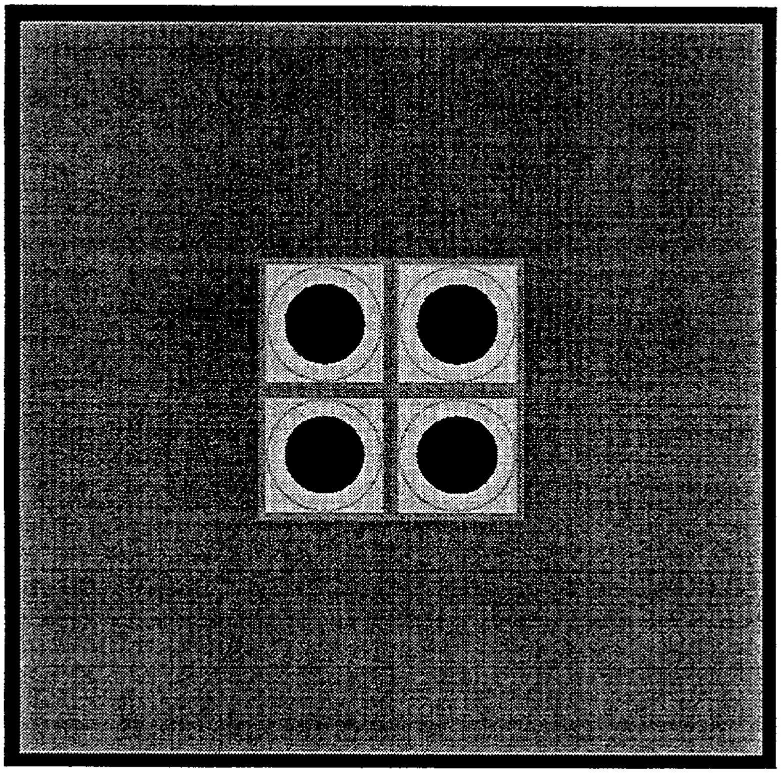


Figure F11.5.64 x-y plot of 2 × 2 × 2 reflected array with centered origin

The Example 5 plot data and associated plots for an enlarged x-y plot, an x-z plot and an enlarged x-z plot are given in Figs. F11.5.65 through F11.5.70.

```
              x-y slice at half height of bottom layer, include 3 cm of refl.

                                       mixture map

mixture  0 1 2 3 4
 symbol    1 2 3 4
      upper  left           lower right
      coordinates           coordinates
x     -1.1600e+01            1.1600e+01
y      1.1600e+01           -1.1600e+01
z     -8.0000e+00           -8.0000e+00
          u axis       v axis
          (down)       (across)
x         0.00000       1.00000
y        -1.00000       0.00000
z         0.00000       0.00000
nu=   640    nv=   640     delu= 3.6250e-02      delv= 3.6250e-02      lpi=  10.000
```

Figure F11.5.65 Plot data for an enlarged x-y slice of example 5

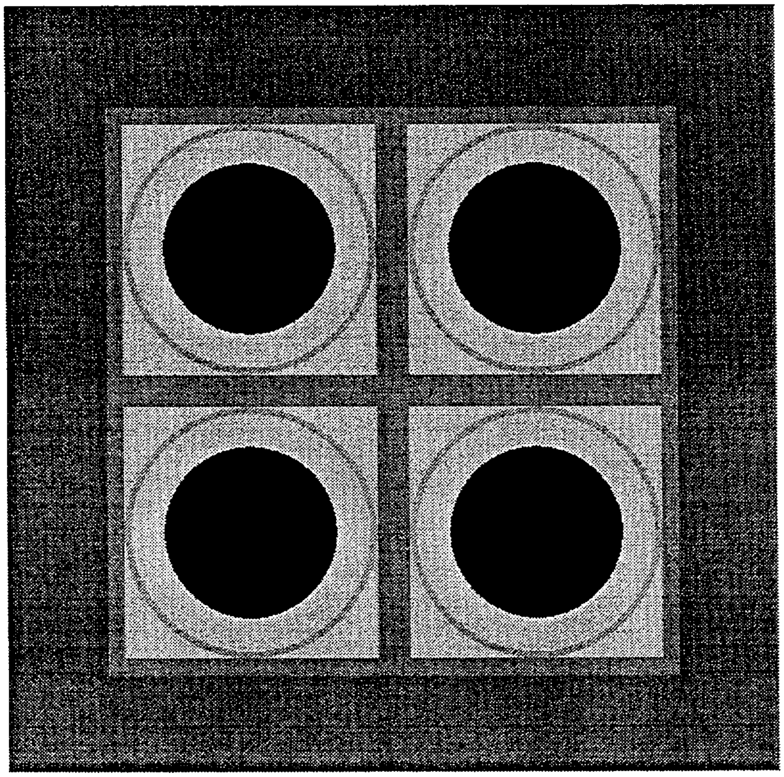


Figure F11.5.66 Enlarged x-y plot of 2 × 2 × 2 reflected array with centered origin

```
                 x-z slice through front row. y=4.3  include reflector

                                  mixture map

mixture   0 1 2 3 4
 symbol     1 2 3 4
      upper  left          lower right
      coordinates          coordinates
x     -2.4840e+01            2.4840e+01
y      4.3000e+00            4.3000e+00
z      4.8240e+01           -4.8240e+01
          u axis      v axis
          (down)      (across)
x        0.00000       1.00000
y        0.00000       0.00000
z       -1.00000       0.00000
nu= 1242   nv=  640     delu= 7.7625e-02     delv= 7.7625e-02     lpi=  10.000
```

Figure F11.5.67 Plot data for x-z slice of example 5

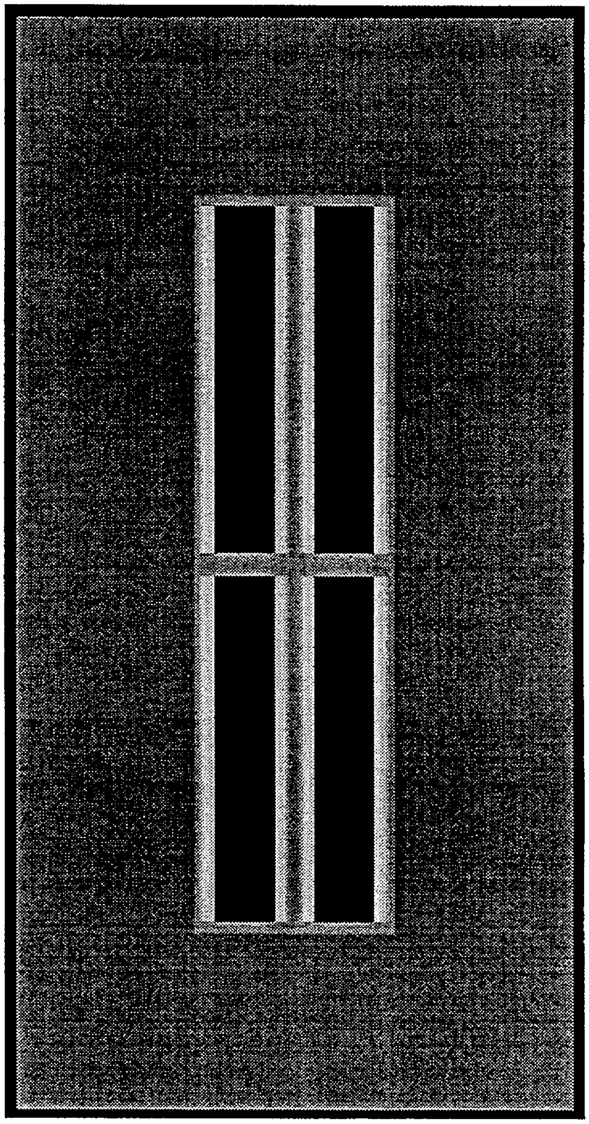


Figure F11.5.68 x-z plot of reflected 2 × 2 × 2 array with centered origin

```
                x-z slice through front row, y=4.3  include 3 cm of reflector
                                        mixture map

                                     mixture  0 1 2 3 4
                                      symbol    1 2 3 4
                                 upper  left           lower right
                                 coordinates           coordinates
                            x    -1.1600e+01             1.1600e+01
                            y     4.3000e+00             4.3000e+00
                            z     3.5000e+01            -3.5000e+01
                                         u axis       v axis
                                         (down)      (across)
                                x       0.00000       1.00000
                                y       0.00000       0.00000
                                z      -1.00000       0.00000
nu= 1931    nv=   640     delu= 3.6250e-02      delv= 3.6250e-02      lpi=  10.000
```

Figure F11.5.69 Plot data for enlarged x-z slice of example 5

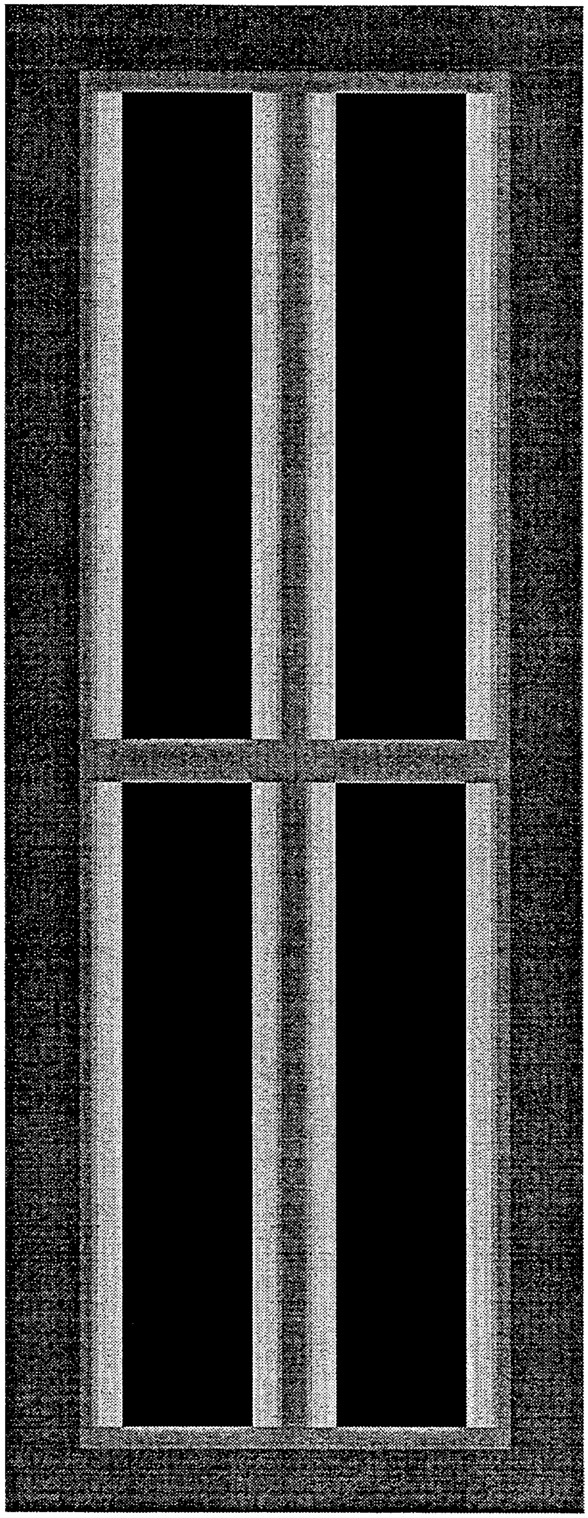


Figure F11.5.70 Enlarged x-z plot of reflected 2 × 2 × 2 array with centered origin

EXAMPLE 6. NESTED HOLES

Refer to the nested hole description of Sect. F11.5.6.2. This example is one that involves a reasonably complicated placement of units or box types. Therefore, it might be useful to the user to generate both a mixture map and a unit map for the problem. The resultant mixture map is shown in Fig. F11.5.71, and the unit map is shown in Fig. F11.5.73. The data description for Example 6 follows.

```
=KENO5
NESTED HOLES SAMPLE
READ PARAM  RUN=NO LIB=41  END PARAM
READ MIXT  SCT=1  MIX=1 92500 4.7048-2 MIX=2 200 1.0 MIX=3 502 0.1
MIX=4 200 1.0
END MIXT
READ GEOM
UNIT 1
CYLINDER 1 1 0.1 2P15.0
UNIT 2
CUBOID 2 1 2P0.1 2P0.05 2P15.0
UNIT 3
CUBOID 2 1 2P0.05 2P0.1 2P15.0
UNIT 4
CYLINDER 1 1 0.1 2P15.0
CYLINDER 3 1 0.5 2P15.0
HOLE 1  0.0 -0.4 0.0
HOLE 1  0.4  0.0 0.0
HOLE 1  0.0  0.4 0.0
HOLE 1 -0.4  0.0 0.0
HOLE 2 -0.2  0.0 0.0
HOLE 2  0.2  0.0 0.0
HOLE 3  0.0 -0.2 0.0
HOLE 3  0.0  0.2 0.0
UNIT 5
CYLINDER 1 1 0.5 2P15.0
UNIT 6
CYLINDER 2 1 0.2 2P15.0
UNIT 7
CYLINDER 2 1 0.2 2P15.0
CYLINDER 0 1 1.3 2P15.0
HOLE 5 0.707107 2*0.0
HOLE 6 0.707107 0.707107 0.0
HOLE 4 0.0 0.707107 0.0
HOLE 6 -0.707107 0.707107 0.0
HOLE 5 -0.707107 0.0 0.0
```

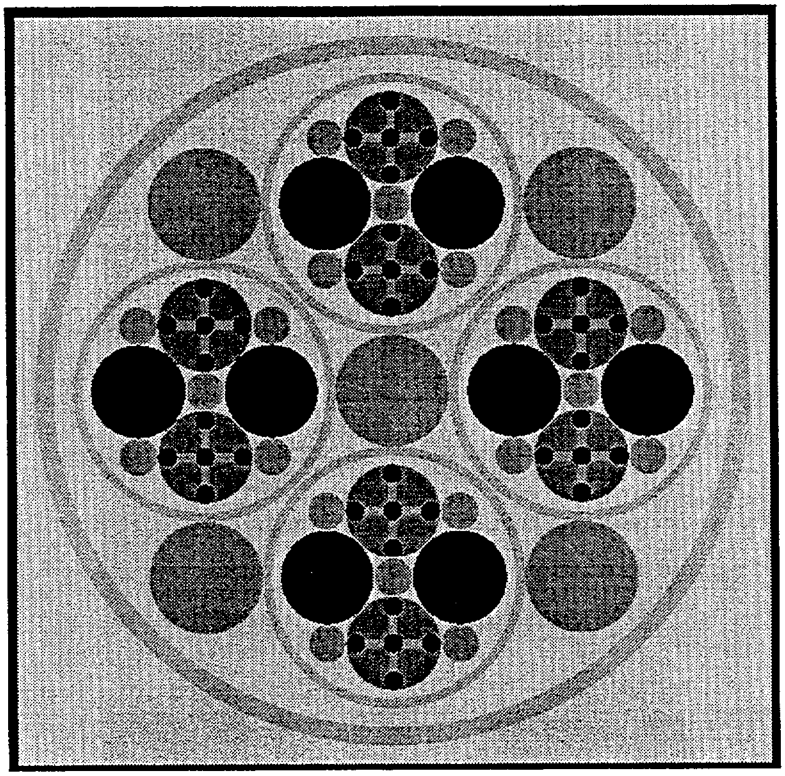


Figure F11.5.71 Mixture map of nested holes problem

```
                 x-y slice at z midpoint.  nested holes.  unit map

                 0                                        unit map

  unit   1 2 3 4 5 6 7 8 9
symbol   1 2 3 4 5 6 7 8 9


      overall system coordinates:
      xmin= 0.00000e+00   xmax= 8.00000e+00   ymin= 0.00000e+00   ymax= 8.00000e+00
      zmin= 0.00000e+00   zmax= 3.20000e+01
      upper  left          lower right
      coordinates          coordinates
x     -1.0000e-01           8.1000e+00
y      8.1000e+00          -1.0000e-01
z      1.6000e+01           1.6000e+01
          u axis       v axis
          (down)       (across)
x         0.00000       1.00000
y        -1.00000       0.00000
z         0.00000       0.00000
nu=   640   nv=   640     delu= 1.2813e-02     delv= 1.2813e-02     lpi=  10.000
```

Figure F11.5.72 Plot data for unit map of nested holes

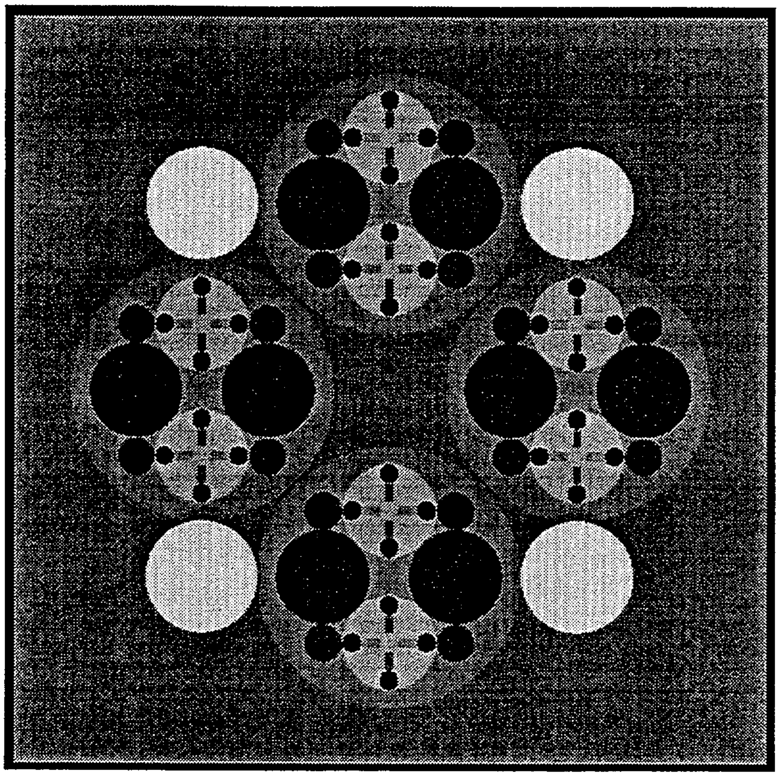


Figure F11.5.73 Unit map of nested holes problem

```
HOLE 6 -0.707107 -0.707107 0.0
HOLE 4 0.0 -0.707107 0.0
HOLE 6 0.707107 -0.707107 0.0
CYLINDER 4 1 1.4 2P15.0
UNIT 8
CYLINDER 2 1 0.6 2P15.0
UNIT 9
CYLINDER 2 1 0.6 2P15.0
CYLINDER 0 1 3.6 2P15.0
HOLE 7 2.0 0.0 0.0
HOLE 8 2*2.0 0.0
HOLE 7 0.0 2.0 0.0
HOLE 8 -2.0 2.0 0.0
HOLE 7 -2.0 2*0.0
HOLE 8 2*-2.0 0.0
HOLE 7 0.0 -2.0 0.0
HOLE 8 2P2.0 0.0
```

```
CYLINDER 4 1 3.8 2P15.0
CUBOID 0 1 4P4.0 2P16.0
END GEOM
READ ARRAY ARA=1 NUX=1 NUY=1 NUZ=1  FILL 9 END ARRAY
READ PLOT
TTL='X-Y SLICE AT Z MIDPOINT.  NESTED HOLES'
XUL=-0.1 YUL=8.1 ZUL=16.0 XLR=8.1 YLR=-0.1 ZLR=16
UAX=1.0 VDN=-1.0 NAX=640  END
TTL='X-Y SLICE AT Z MIDPOINT.  NESTED HOLES.  UNIT MAP'
PIC=UNIT  END
END PLOT
END DATA
END
```

The plot data description for unit map of nested holes are shown in Fig. F11.5.72. The unit map is shown in Fig. F11.5.73. The user can utilize this map to verify the correct placement of the units. Note that the unit map plots the units that are present at the deepest nesting level for the 2-D slice. It does not show any detail within a unit. The apparent detail within Unit 7 are Units 1, 2, 3, 4, 5, and 6 that were placed there via the hole option. In the legend of the plot, the material number actually refers to the unit number.

EXAMPLE 7. LARGE STORAGE ARRAY

The storage array described in Sect. F11.5.6.3 and Fig. F11.5.15 is such a sparse array that the mixture map had to be very large in order to show the detail of the shelves and uranium buttons. The mixture maps for this configuration were not presented in Sect. F11.5.6.3, but the data description was listed so the user could generate them. It may be useful to generate a unit map for this kind of problem. The input data for generating unit maps for this storage array is given below.

```
READ PLOT  PIC=UNIT
TTL='X-Z SLICE THROUGH STORAGE ARRAY ROOM AT Y=30.48 WITH Z ACROSS AND X DOWN'
XUL=624.84 YUL=30.48 ZUL=-45.72 XLR=-30.48 YLR=30.48 ZLR=381.0
WAX=1.0 UDN=-1.0 NAX=320  END
TTL='X-Y SLICE THROUGH STORAGE ARRAY ROOM AT Z=0.3175 WITH X ACROSS AND Y
DOWN'
XUL=-30.48 YUL=1341.1 ZUL=0.3175 XLR=624.84 YLR=-30.48 ZLR=0.3175
UAX=1.0 VDN=-1.0 NAX=320 END
END PLOT
```

The z direction which extends from -45.72 cm to 381.0 cm is plotted in 320 pixels across the plot. The plot data and unit map for an x-z slice through the array at y=30.48 cm is given in Figs. F11.5.74 and F11.5.75. This unit map was created with z across the plot and x down the plot.

```
         x-z slice through storage array room at y=30.48 with z across and x down

                                      unit map

     unit  1 2 3 4 5 6 7
   symbol  1 2 3 4 5 6 7
        upper  left          lower right
        coordinates          coordinates
 x       6.2484e+02          -3.0480e+01
 y       3.0480e+01           3.0480e+01
 z      -4.5720e+01           3.8100e+02
            u axis      v axis
            (down)      (across)
 x        -1.00000      0.00000
 y         0.00000      0.00000
 z         0.00000      1.00000
 nu=  491   nv=  320     delu= 1.3335e+00      delv= 1.3335e+00      lpi=  10.000
```

Figure F11.5.74 Plot data for x-z slice of storage array

The plot data and unit map for an x-y slice through the shelf are given in Figs. F11.5.76 and F11.5.77. This unit map was created with x across the plot and y down the plot. This shows 5 rows of shelves in the x direction.

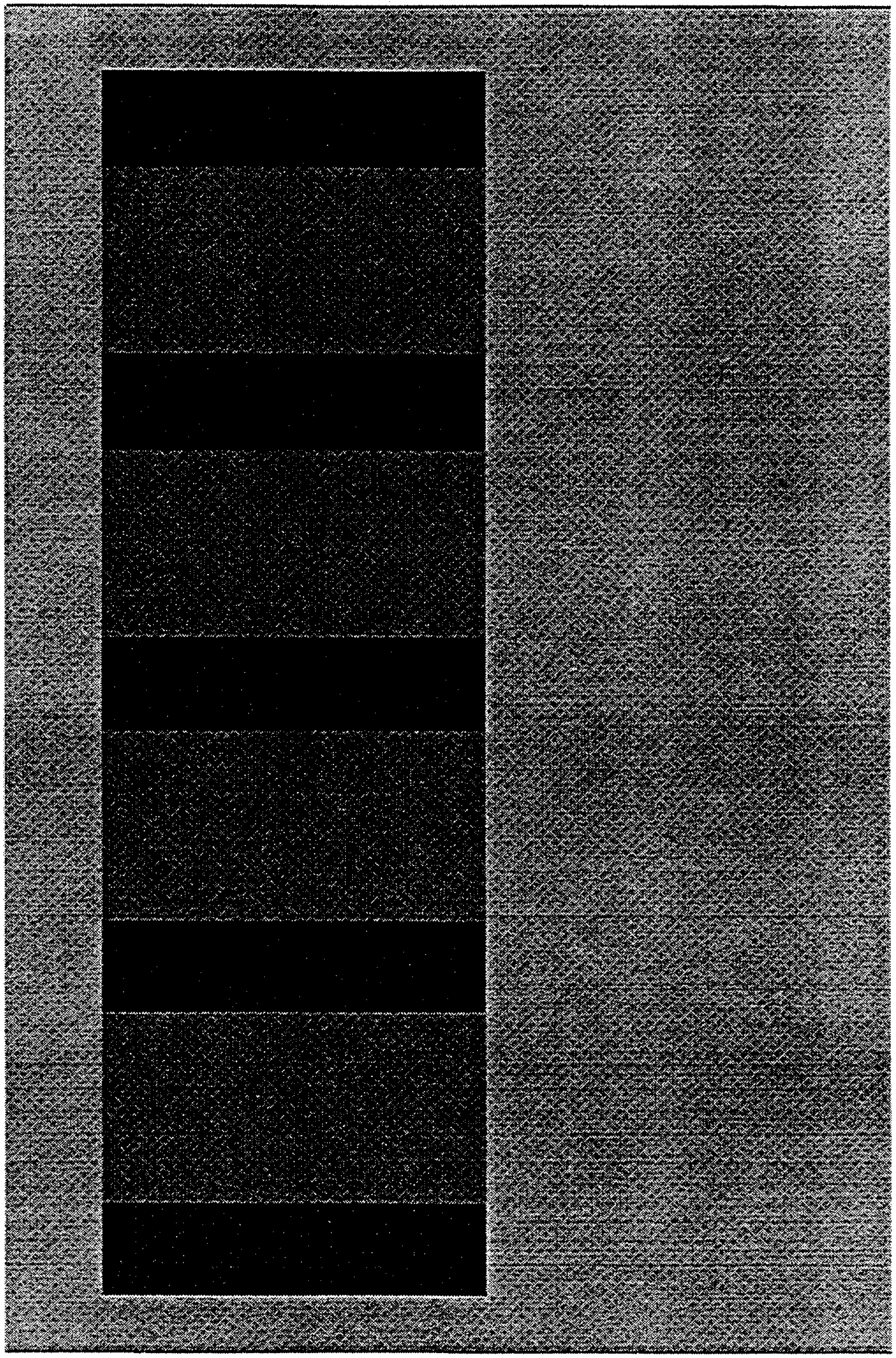


Figure F11.5.75 x-z plot of storage array

```
     x-y slice through storage array room at z=0.3175 with x across and y down
                                          unit map

   unit   1 2 3 4 5 6 7
 symbol   1 2 3 4 5 6 7
        upper  left          lower right
        coordinates          coordinates
x     -3.0480e+01            6.2484e+02
y      1.3411e+03           -3.0480e+01
z      3.1750e-01            3.1750e-01
           u axis      v axis
           (down)      (across)
x          0.00000      1.00000
y         -1.00000      0.00000
z          0.00000      0.00000
nu=   669    nv=   320     delu= 2.0479e+00     delv= 2.0479e+00     lpi=  10.000
```

Figure F11.5.76 Plot data for x-y slice of storage array

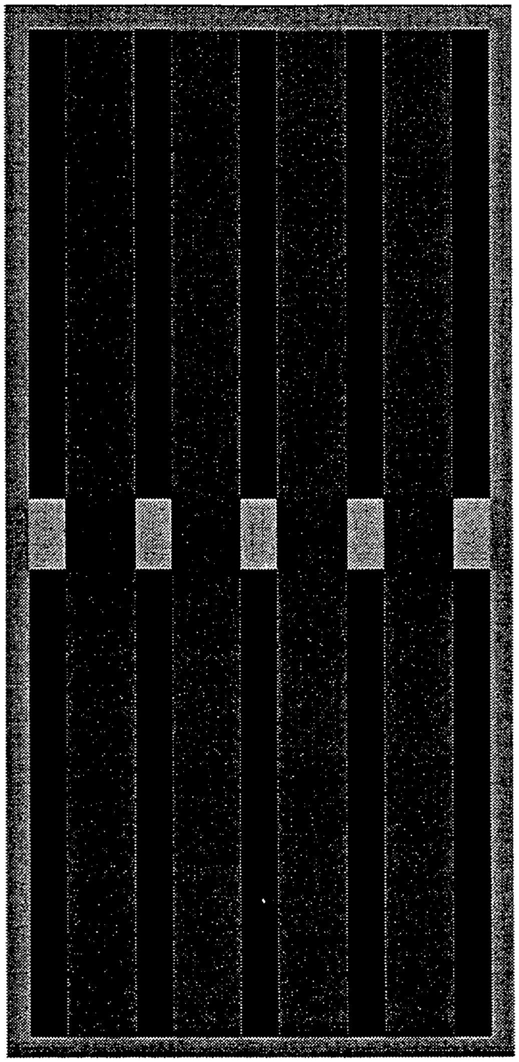


Figure F11.5.77 x-y plot of storage array

F11.5.10 RESTART CAPABILITIES

Restart data can be written and used for restarting a problem. This type of data is saved by specifying a file definition card in the job-control language on the unit associated with parameter WRS= when it is written, and RST= when it is read. Most input data can be changed *only* if the problem is restarted with the first generation. However, certain parameter data can be changed if the problem is restarted at a generation greater than 1.

Parameters that can be changed when a problem is restarted at a generation greater than 1 include: RND=, TME=, TBA=, GEN=, RES=, LNG=, BEG=, AMX=, XAP=, XS1=, XS2=, PKI=, P1D=, CKU=, CKP=, CKH=, CKA=, FMU=, FMP=, FMH=, FMA=, BUG=, TRK=, PWT=, PGM=, RUN=, PLT=, NB8=, NL8=, and PAX=. All the logical unit numbers can be changed.

If NSK= is changed, it will cause the fluxes, fission densities, leakage, absorptions, and fissions to be incorrect.

FLX=, FDN=, FAR=, MKU=, MKP=, MKH= and MKA= can be changed subject to the following restrictions:

1. If the original problem written to a particular restart file specified YES, and the restarted problem specifies NO, the data will be calculated but not printed. A warning message will be printed.

2. If the original problem written to a particular restart file specified NO, and the restarted problem specifies YES, an error message is printed and the problem is terminated.

The parameters RUN= and PLT= differ from other parameters because they can be set in the parameter data and overridden in the plot data. Their values are stored in a common block that is written out if a problem is to be restarted. Therefore, when a problem is run, the value entered in the plot data will override the value that was entered in the parameter data. However, if the problem is restarted, the value from the restart file will be used unless it is overridden by entering additional data.

If WRS= is defined in the parameter data and RES= is not entered, a full restart data file is not written. The input data are written on the restart data file but the calculated data are not. When the restart data file is written in this manner, it can only be used to restart a problem at the first generation. Any desired data can be overridden when a problem is restarted at the first generation. This is accomplished by reading in the desired data.

To write a complete restart data file, both WRS= and RES= must be specified in the parameter data. However, WRS= is defaulted to 35 if a value greater then zero is entered for RES=. In this case, a file definition card for Unit 35 must be supplied by the user in order to save the restart data.

A problem that is restarted can also write a restart data file. These data are written on the same data file if no entry for WRS= is made in the parameter data. It can be written on a different unit if WRS= is so specified in the parameter data and the proper file definition card is included in the job control language. If a restarted problem does not have RES=0 specified in the parameter data, it will continue writing calculated restart data. Therefore, the user can run a long problem a little bit at a time by allowing the restarted problem to write restart data and then restarting the problem with those data. This can be done in sequence until the desired number of generations have been completed.

For example, consider a problem that is to run 500,000 histories (500 generations of 1000 histories per generation) and the amount of computer time available at any given time is quite limited. On the first pass, GEN=500 NPG=1000 RES=500 WRS=35 should be included in the parameter data and a unit specification for Unit 35 should be included in the job control language when the problem is run. Note that 35 is an arbitrary number chosen by the user. KENO V.a will automatically pull the job before it runs out of time or I/Os. A restart data file will be written on Unit 35 for the last generation that was completed. If the user wishes to restart the problem by reading the data from Unit 35 and writing a new restart file on the same unit, the parameter data for the second and all subsequent passes would be: READ PARAM BEG=500 RST=35 END PARAM.

If the user wishes to run additional histories to improve convergence, BEG should be set to one greater than the last generation saved on the restart file. For example, if a problem was originally run with GEN=100, RES=100 WRS=35, then the restart problem should specify BEG=101 GEN=200 RST=35, in order to run an additional 100 generations.

The random sequence can be changed when a problem is restarted by entering a different random number in the parameter data. The starting random number is acceptable for changing the random sequence of a problem that is to be restarted at a generation greater than 1.

Note that if a problem is restarted from one unit and restart data are written to another unit, the new problem description becomes the "original problem" for the new restart dataset.

Parameter data can be changed by entering new values for the desired options. This has the effect of inputting additional data within the parameter data block as explained in Sect. F11.5.3. However, all other data blocks that are to be changed must be entered as an entire data block and will completely replace that data block from the restart problem. When a problem is restarted and some of the data are overridden, the original restart data blocks are retained on the unit defined by WRS=, if it is the same as the unit defined by RST=. If the unit number associated with WRS= is different from the unit number associated with RST=, the resultant data blocks are written on the unit number associated with WRS=. For example, if the original problem is a single unit problem with RST=95 and WRS=95, and an array data block is entered to change it to an array problem, the original single unit problem data blocks remain on Unit 95. However, if the single unit problem is restarted with RST=95 and WRS=96, and an array data block is entered to change the problem to an array problem, the original single unit data blocks remain on Unit 95 and the resultant array problem data blocks are retained on Unit 96.

When a problem is restarted, the only way to change data in a data block is to reenter a new data block. For example, if in the original problem an array data block was entered as: READ ARRAY NUX=11 NUY=5 NUZ=3 END ARRAY, and the problem is to be restarted at the first generation as an 11 × 5 × 6 array, the following data block must be entered: READ ARRAY NUX=11 NUY=5 NUZ=6 END ARRAY.

Section F11.2.6 contains some information pertaining to the restart capability. Other information may be found in Sect. F11.4.3. The structure of the restart file is listed in Table F11.5.2, and the variables referenced in that table are listed in Table F11.5.3.

F11.5.11 RANDOM SEQUENCE

The random-number package utilized by KENO V.a always starts with the same seed and thus always reproduces the same sequence of random numbers. The current random number is printed at various places in the KENO V.a printout, and any of them except the one printed in the parameter table can be used to activate a different random sequence. The user can rerun a problem with a different random sequence by simply entering a hexidecimal random number, other than the starting random number, in the parameter data. For example, by entering RND=A10C1893E6D5 in the parameter data, the problem will be run with a different random sequence.

Table F11.5.2 Structure of RESTART file

Record No.	Contents
(1)	TITLE(20), NBA, NPB, NSKIP, NRSTRT, NBANK, NFBNK, NXBNK, NXFBK, NUMX1D, TMAX, TBTCH, RNDNUM, LOG(36) (This record contains the title and parameter data.)
(2)	MT(LMT) LMT = 5 + NUMX1D (This record contains the identifiers of the 1-D cross-section arrays.)
(3 to NUMPT+2)	NDX = 1, NUMPT NUMPT = 11 (a) NDX,NREC (This record contains the index for each type of data and the number of records of data associated with each type of data.) (b) NREC - 1 record of data (These records contain the data associated with the specified type of data.)
	NDX = 1 (geometry data) (b1) LLNGTH, KMAX, KREFM, NGBLU, NBOXT, MAXMIX, MAXIMP, NUMHOL, NUCOM, EXRFL (b2) MAT(KREFM), IMP(KREFM), IGEOM(KREFM), XX(7,KREFM), KBNDS1(NBOXT), KBNDS2(NBOXT), IFH(KREFM), ILH(KREFM), KHOLE(NUMHOL), LHOLU(NUMHOL), HOLX(NUMHOL), HOLY(NUMHOL), HOLZ(NUMHOL), ICOMC(NBOXT), UCOMNT(33,NUCOM)
	NDX = 2 (array or unit orientation data) (b1) LLNGTH, NGLOBL, MAXARA, NACOM, LSGUN, MBOX, LFIL (b2) NBXMAX(MAXARA), NBYMAX(MAXARA), NBZMAX(MAXARA), ITYPE(MAXARA), LPT(MAXARA), LNG(MAXARA), ICOMA(MAXARA), ACOMNT(33,NACOM) I = 1, MAXARA (The following records are written for each array if multiple units are entered in the geometry data.) (b3) IF (LPT(I).GT.0) LBA(NBXMAX(I),NBYMAX(I),NBZMAX(I))
	NDX = 3 (mixing table data)

Table F11.5.2 (continued)

Record No.	Contents
	(b1) LLNGTH, NMIX, NSCT, MIX, MIXT, NPL, PBXS
	(b2) MIXTUR(NMIX), NUC(NMIX), DEN(NMIX)
	NDX = 4 (extra data)
	(b1) LLNGTH, LNEXTR
	(b2) D(LLNGTH)
	NDX = 5 (weighting function by energy group and importance reg.)
	(b1) LLNGTH, NIMP
	(b2) WTAVG(NGP,NIMP)
	NDX = 6 (start data)
	(b1) LLNGTH, NTYPST, TFX, TFY,TFZ, NBXS, NBYS, NBZS, KFIS, LFIN, NBOXST, FRACT, FISVOL, XSM, XSP, YSM, YSP, ZSM, ZSP, RFLKEY, LPRT6, LPSTP
	(b2) IF (NTYPST .EQ. 6) NUBANK(LFIN,LBANK0)
	NDX = 7 (albedo data)
	(b1) LLNGTH, NALB, NANG, NG, INTR(6), RNAMES(2,6), IDALB(6), NBXL(6), LNXX
	IF (NALB .GT. 0)
	(b2) NABS(3,NALB), LABS(3,NALB)
	(b3) MAL(3,NANG,NG)
	(b4) EALB(NG+1) (group boundaries)
	I = 1,NALB
	(b5(i)) PLIM(NANG), CPOL(NANG), SPOL(NANG)

Table F11.5.2 (continued)

Record No.	Contents
	(b6(i)) ALB(LENG) (LENG is the length of the albedos for a given angle.)
	(b7(i)) A(LENG,NANG)
	NDX = 8 (mixed cross sections)
	(b1) LLNGTH, NMAT, NGP, MAXANG
	(b2) LXS(MANG,MATT) MANG = 2*NSCT + 4
	I = 1, MATT for each existing mixture
	(b3(i)) ID(50) the mixture information record
	(b4(i)) X1D(NN1D,NGP+1) NN1D = ID(28) no. of 1-D's
	(b5(i)) MWA(3,NGP)
	(b6(i)) P0(LNG) LNG = MWA(2,NGP)
	J = 1, NSCT
	(b7(i,j)) ANG(LNG)
	(b8(i,j)) PRB(LNG)
	NDX = 9 (energies and inverse velocities)
	(b1) LLNGTH
	(b2) E(NGP+1), VINV(NGP)
	NDX = 10 (plot data)
	(b1) LLNGTH, NUMPLT
	I = 1, NUMPLT
	(b2(i)) XL, YL, ZL, XR, YR, ZR, VX, VY, VZ, UX, UY, UZ, DELV, DELU, NV, NU, LPIC, NSTOR(49)
	NDX = 11 (biasing data)

Table F11.5.2 (continued)

Record No.	Contents
	(b1) LLNGTH, NUMIDS, NCS, NTSETS
	IF ((NUMIDS + NCS).GT.0),
	(b2) ID(NUMIDS), IBGN(NUMIDS), IEND(NUMIDS), WTTITL(3,NUMIDS), NCID(NCS), NCSETS(NCS), CRDTTL(3,NCS), NCTHK(NTSETS), NUMINC(NTSETS), NGPWTS(NTSETS), IPTWT(NTSETS)
	I = 1, NCS
	J = 1, NSETS(I)
	(b3(i,j)) WTAVG(NGPWTS(j,i),NUMINC(j,i))
(14 to END)	calculated data as listed below
	(a) IGEN, RND, NPB, NGP, KMAX, LBANK, NBANK, DLIF(15), LIF(20), LOJIC(36), EFFK(I),I=1,IGEN) (This record contains parameter data, COMMON /LIFETM/ part of COMMON /LOGIC/ and k-effectives by generation.)
	(b) NUBANK(LBANK,NBANK)
	(c) IF(LOJIC(4)) FISDEN(KMAX,3) (If fission densities are calculated, the fission densities are written on the restart file.)
	(d) IF(LOJIC(10)) TP(MATDIM,MATDIM,3), SNP(MATDIM), SP(MATDIM) (If matrix data by position are calculated, the matrix by position data are written on the restart file.)
	(e) IF(LOJIC(7)) TU(NBOXT, NBOXT,3), SNU(NBOXT), SU(NBOXT) (If matrix data by unit are calculated, the matrix by unit data are written on the restart file.)
	(f) IF(LOJIC(13)) TH(NUMHOL,NUMHOL,3), SNH(NUMHOL), SH(NUMHOL) (If matrix data by hole are calculated, the matrix by hole data are written on the restart file.)

Table F11.5.2 (continued)

Record No.	Contents
	(g) IF(LOJIC(17)) TA(NUMARA,NUMARA,3), SNA(NUMARA), SA(NUMARA) (If matrix data by array are calculated, the matrix by array data are written on the restart file.)
	(h) 1 to NGP (the following records are written for each energy group.
	(h1) IGROUP, FLEAK(3)
	(h2) FMABS(LREG,3), FMFIS(LREG,3)
	(h3) IF(LOJIC(3)) FLUX(KMAX,3)
	Repeat (a through h) until LOJIC(34) is TRUE (i.e., until the last generation is completed).

Table F11.5.3 Key of RESTART file variables

TITLE	80-character KENO V.a problem title
NBA	Number of generations in the KENO V.a problem
NPB	Number of histories per generation
NSKIP	Number of generations to be skipped in averaging k-effective
NRSTRT	Number of generations between writing restart data
NBANK	Number of positions in the neutron bank
NFBNK	Number of positions in the fission bank
NXBNK	Number of extra entries in the neutron bank
NXFBK	Number of extra entries in the fission bank
NUMX1D	Number of extra 1-D cross sections
TMAX	Time allowed to execute the problem
TBTCH	Time allowed for each generation
RNDNUM	Random number with which the problem will be started
LOG	COMMON /LOGIC/
LMT	Number of 1-D cross-section identifiers. LMT = 5 + NUMX1D
MT	Array containing the 1-D cross-section identifiers
NUMPT	Number of kinds of data that are written on the restart file
NDX	Index for the type of data
NREC	Number of records for the specified type of data
NDX = 1	Geometry data
LLNGTH	The length of record b2
KMAX	Number of geometry regions used
KREFM	Number of geometry cards read in the geometry data
NGBLU	The global unit number
NBOX	Largest unit or boxtype number in the geometry data
NBOXT	NBOX + the number extra units generated by KENO V.a
MAXMIX	Largest mixture number encountered in the geometry data
MAXIMP	Largest biasing number encountered in the geometry data
NUMHOL	Number of holes in the geometry data
NUCOM	The number of units having comments in the geometry region data
EXRFL	Logical flag. Value is TRUE if a reflector is present
MAT	Array containing the mixtures used in the geometry
IMP	Array containing the bias IDs used in the geometry
IGEOM	Array containing the shape identifiers used in the geometry
XX	Array of geometry dimensions
KBNDS1	Array of the first geometry region number in each unit
KBNDS2	Array of the last geometry region number in each unit
IFH	First hole number encountered in each geometry region
ILH	Last hole number encountered in each geometry region
KHOLE	Region number that contains the hole
LHOLU	Unit that is placed in the hole

Table F11.5.3 (continued)

HOLX	X coordinate of the origin of the unit in the hole with respect to the unit that contains the hole
HOLY	Y coordinate of the origin of the unit in the hole with respect to the unit that contains the hole
HOLZ	Z coordinate of the origin of the unit in the hole with respect to the unit that contains the hole
ICOMC	The index into the comment array for a unit
UCOMNT	Comments associated with units specified in the geometry region data
NDX = 2	Array or unit orientation data
LLNGTH	Length of the b2 record
NGLOBL	Global array number
MAXARA	Largest array number encountered in the array data
NACOM	The number of arrays having comments in the array data
LSGUN	Logical flag. Value is TRUE if there is no array present
MBOX	A logical variable that is set .TRUE. if multiple box types are specified
NBXMAX	Number of units in the X direction of each array
NBYMAX	Number of units in the Y direction of each array
NBZMAX	Number of units in the Z direction of each array
ITYPE	Pitch indicator, 1 for square pitched arrays
LPT	Pointer to locate the beginning of each unit orientation array
LNG	Length of each unit orientation array
ICOMA	The index into the comment array for arrays
LBA	Contains the unit orientation arrays for all arrays
MAXARA	The largest array number
NACOM	Number of array comments
ACOMNT	Comments
NDX = 3	Mixing table data
LLNGTH	Length of b2 record
NMIX	Number of entries in the mixing table
NSCT	Number of scattering angles
MIX	Number of different mixtures to be mixed
MIXT	Largest mixture number to be mixed
NPL	Order of Legendre coefficients + 1
MIXTUR	Array of mixture numbers used in the mixing table
NUC	Array of the nuclide ID numbers used in the mixing table
DEN	Array of the number densities used in the mixing table
NDX = 4	Extra data
LLNGTH	Length of b2 record
LENGTH	D(LENGTH) is the data contained in the b2 record

Table F11.5.3 (continued)

LNXTR	Length of the extra data
NDX = 5	Weighting function
LLNGTH	Length of b2 record
NIMP	Number of biasing regions
WTAVG	Average weight by energy group and biasing region
NDX = 6	Start data (initial source distribution)
LLNGTH	Length of b2 record
NTYPST	Start type to define the initial source distribution
TFX	X coordinate of neutron starting point
TFY	Y coordinate of neutron starting point
TFZ	Z coordinate of neutron starting point
NBXS	X index of unit's position in the global array
NBYS	Y index of unit's position in the global array
NBZS	Z index of unit's position in the global array
KFIS	Mixture whose fission spectrum is used for initial source
LFIN	Last neutron to be started at the specified point
NBOXST	Unit in which neutrons will be started
FRACT	Fraction of initial source to be started as a spike
FISVOL	Fraction of global system that contains fissile material
XSM	-X dimension of cuboid in which neutrons will be started
XSP	+X dimension of cuboid in which neutrons will be started
YSM	-Y dimension of cuboid in which neutrons will be started
YSP	+Y dimension of cuboid in which neutrons will be started
ZSM	-Z dimension of cuboid in which neutrons will be started
ZSP	+Z dimension of cuboid in which neutrons will be started
RFLKEY	Logical variable. Set TRUE if neutrons can be started in reflector
LPRT6	Logical variable. Set TRUE to print start type 6 data
LPSTP	Logical variable. Set TRUE to print initial source points
NUBANK	The neutron bank. The variables that are used from the neutron bank are X(NPB),Y(NPB),Z(NPB),NBX(NPB),NBY(NPB),NBZ(NPB)
LFIN	The last neutron for which start type 6 data was entered
LBANK	The number of positions per neutron in the neutron bank
NPB	Number of histories per generation
X	X coordinate of neutron starting point for start type 6
Y	Y coordinate of neutron starting point for start type 6
Z	Z coordinate of neutron starting point for start type 6
NBX	X index of unit's position in the global array for start type 6
NBY	Y index of unit's position in the global array for start type 6
NBZ	Z index of unit's position in the global array for start type 6

Table F11.5.3 (continued)

NDX = 7	Albedo data
LLNGTH	Length of b2 record
NALB	Number of different differential albedos to be used
NANG	Number of angles available in the albedo function
NG	Number of energy groups available in the albedo function
INTR	Type of boundary condition for each face
RNAMES	Name of boundary condition
IDALB	Index to the correct set of albedo data
NBXL	Specifies the face where the history will reenter
LNXX	Logical flag. Set TRUE for either specular or differential reflection
NABS	Pointers into the albedo data for each albedo used
LABS	Length of the albedo data for each albedo used
MAL	Pointer array for albedos
EALB	Energy boundaries for albedos
PLIM	Incident polar angle bins
CPOL	Cosine of returning polar angle
SPOL	Sine of returning polar angle
ALB	Probabilities for the returning energy group
A	Probabilities for the returning angle
LENG	Length of albedo data for a given angle
NDX = 8	Mixed cross sections
LLNGTH	Length of b2 record
NMAT	Number of mixtures used in the problem
NGP	Number of energy groups from the cross-section file
MAXANG	Number of scattering angles
LXS	Length of each cross-section set
MANG	2*NSCT + 4
ID	Head record from mixed cross-section file
X1D	1-D cross sections
NN1D	Number of 1-D cross sections (ID(28))
MWA	Pointer array for cross sections
P0	Group-to-group transfer probabilities
ANG	Scattering angles
PRB	Probabilities for scattering at angle ANG
LNG	Length of the P0 arrays
NDX = 9	Energies and inverse velocities
LLNGTH	Length of b2 record
NGP	Number of energy groups
E	Energy group bounds
VINV	Inverse velocities

Table F11.5.3 (continued)

NDX = 10	Plot data
LLNGTH	Length of b2 record
NUMPLT	Number of plots to be generated
XL	X coordinate of the upper left corner of the plot
YL	Y coordinate of the upper left corner of the plot
ZL	Z coordinate of the upper left corner of the plot
XR	X coordinate of the lower right corner of the plot
YR	Y coordinate of the lower right corner of the plot
ZR	Z coordinate of the lower right corner of the plot
VX	X component of the direction cosine for the across axis of the plot
VY	Y component of the direction cosine for the across axis of the plot
VZ	Z component of the direction cosine for the across axis of the plot
UX	X component of the direction cosine for the down axis of the plot
UY	Y component of the direction cosine for the down axis of the plot
UZ	Z component of the direction cosine for the down axis of the plot
DELV	Horizontal spacing between points on the plot
DELU	Vertical spacing between points on the plot
NV	Number of characters across the plot
NU	Number of characters down the plot
LPIC	Plot type indicator. LPIC=1 mixture, LPIC=2 unit, LPIC=3 bias ID
NSTOR	An array that contains PTITL and TABLE. The first 33 words are PTITL and the last 16 are TABLE.
NDX = 11	Biasing data
LLNGTH	Length of b2 record
NUMIDS	Number of bias IDs requested
NCS	Number of bias IDs entered from cards
NTSETS	Total number of group structures for all IDs
ID	ID number or set of weights to be read from WTS file
IBGN	Beginning importance region number assigned to ID
IEND	Ending importance region number assigned to ID
WTTITL	Title associated with the ID
NCID	ID to be read from cards
NCSETS	Number of group structures to be read from cards
CRDTTL	Title associated with the ID read from cards
NCTHK	Thickness per increment from cards
NUMINC	Number of increments from cards
NGPWTS	Number of energy groups for this set of weights
IPTWT	Pointer into the weights from cards
WTAVG	Weighting function by energy group and importance region

Table F11.5.3 (continued)

Calculated data	
IGEN	Generation for which this set of restart data was written
RND	Last random number that was used
NPB	Number of generations
NGP	Number of histories per generation
KMAX	Number of geometry regions used
LBANK	Number of positions per neutron in the neutron bank
NBANK	Number of positions in the neutron bank
DLIF	The double precision portion of COMMON/LIFETM/
LIF	The single precision portion of COMMON /LIFETM/
LOJIC	An array equivalenced to COMMON /LOGIC/
EFFK	Array containing k-effectives by generation
NUBANK	The neutron bank, NBANK by LBANK
FISDEN	Array containing the fission densities
TP	Fission production matrix by array position
SNP	Eigenvector of the fission production matrix by array position
SP	Source vector by array position
TU	Fission production matrix by unit
SNU	Eigenvector of the fission production matrix by unit
SU	Source vector by unit
TH	Fission production matrix by hole
SNH	Eigenvector of the fission production matrix by hole
SH	Source vector by hole
TA	Fission production matrix by array
SNA	Eigenvector of the fission production matrix by array
SA	Source vector by array
IGROUP	Current energy group
FLEAK	Leakage fraction
FMABS	Absorption fraction
FMFIS	Total fission production
FLUX	Flux
LOJIC(4)	Key indicating whether to calculate fission densities Specified by entering FDN= in the parameter data
LOJIC(10)	Key indicating whether to calculate matrix data by position Specified by entering MKP= in the parameter data
LOJIC(7)	Key indicating whether to calculate matrix data by unit Specified by entering MKU= in the parameter data
LOJIC(13)	Key indicating whether to calculate matrix data by hole Specified by entering MKH= in the parameter data
LOJIC(17)	Key indicating whether to calculate matrix data by array Specified by entering MKA= in the parameter data
LOJIC(3)	Key indicating whether to calculate fluxes Specified by entering FLX= in the parameter data

F11.5.12 MATRIX K-EFFECTIVE

Matrix k-effective calculations provide an alternative method of calculating the k-effective of the system. Cofactor k-effectives and source vectors are additional information that can be provided when the matrix k-effective is calculated. The necessary source and fission weight data are collected during the neutron tracking procedure in subroutine TRACK. This information is converted to a FISSION PRODUCTION MATRIX which is the number of next generation neutrons produced at J by a neutron born at I. The principal eigenvalue of the fission probability matrix is the matrix k-effective. KENO V.a offers four alternatives when calculating matrix k-effective as discussed below:

(1) If MKP=YES is specified in the parameter data, the fission production matrix is collected by array position or position index. The position index is used to reference a given location in a 3-D lattice. For a 2 × 2 × 2 array there are nine unique position indices as shown below. Position zero contains everything outside the global array.

POSITION INDEX	X	POSITION Y	Z
0	0	0	0
1	1	1	1
2	2	1	1
3	1	2	1
4	2	2	1
5	1	1	2
6	2	1	2
7	1	2	2
8	2	2	2

The fission production matrix is the number of next generation neutrons produced at index J by a neutron born at index I. This matrix is used to calculate the matrix k-effective, cofactor k-effectives and the source vector by position index. Because the size of the fission probability matrix is the square of the array size (for a 4 × 4 × 4 array there are 4096 entries), it can use vast amounts of computer memory.

(2) If MKU=YES is specified in the parameter data, the fission production matrix is collected by unit or box type. It is the number of next generation neutrons produced in unit or box type J by a neutron born in unit or box type I. This matrix is used to calculate the matrix k-effective, cofactor k-effectives and source vector by unit or box type.

(3) If MKH=YES is specified in the parameter data, the fission production matrix is collected by hole number. Matrix information can be collected at either the highest hole nesting level (first level of nesting) or the deepest hole nesting level. HHL=YES specifies that the matrix information will be collected at the first nesting level. By default, the matrix information is collected at the deepest nesting level. The fission production matrix is the number of next generation neutrons produced in hole J by a neutron born in hole I. This matrix is used to calculate the matrix k-effective, cofactor k-effectives and the source vector by hole.

(4) If MKA=YES is specified in the parameter data, the fission production matrix is collected by array number. It can be collected at the highest array level (first level of nesting) or at the deepest array level. HAL=YES specifies that the matrix information will be collected at the first nesting level. By default, the matrix information is collected at the deepest nesting level. The fission production matrix is the number of next generation neutrons produced in array J by a neutron born in array I. This matrix is used to calculate the matrix k-effective, cofactor k-effectives and the source vector by array.

The user can simultaneously utilize all methods of calculating the matrix k-effective. The results are labeled in the printout. Matrix k-effectives cannot be calculated for a single unit problem. If the user wishes to do so, the geometry description must have a cube or cuboid as its outer region and the problem description should include READ ARRAY END ARRAY. These two actions convert the single unit problem into a 1 × 1 × 1 array.

A cofactor k-effective is the eigenvalue of the fission production matrix, reduced by the row and column that references the specified unit or position index. The difference between the k-effective for the system and the cofactor k-effective for a unit or position index is an indication of the in situ k-effective of that unit or the contribution that unit makes to the k-effective of the system. The cofactor k-effective of a unit devoid of fissile material should approximate the k-effective of the system.

F11.5.13 DEVIATIONS

When a deviation is calculated by KENO V.a, it is the standard deviation of the mean. This assumes a large sample having a normal distribution. If results do not fall in this category, the deviations cannot be assumed to be correct. KENO V.a serially groups generations together to reduce the effects of serial correlations. The grouping is varied such that a minimum of 30 groups is maintained (if possible) and the grouping that produces the maximum deviation is used.

F11.5.14 GENERATION TIME AND LIFETIME

The generation time and lifetime calculations utilize the average velocity. The validity of these calculations is determined by how accurately the average velocity represents the spectrum over the range of the energy group. The lifetime and generation time calculated by KENO V.a are not kinetics parameters. The lifetime is the average lifespan of a neutron (in seconds) from the time it is born until it is absorbed or leaks from the system. The generation time is the average time (in seconds) between successive neutron generations.

F11.5.15 REFERENCES

1. S. K. Fraley, *Users Guide for ICE-II,* ORNL/CSD/TM-9/R1, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., July 1977.

2. J. T. Thomas, "Critical Three-Dimensional Arrays of U(93.2)-Metal Cylinders," *Nucl. Sci. Eng.* **52**, 350 (November 1973).

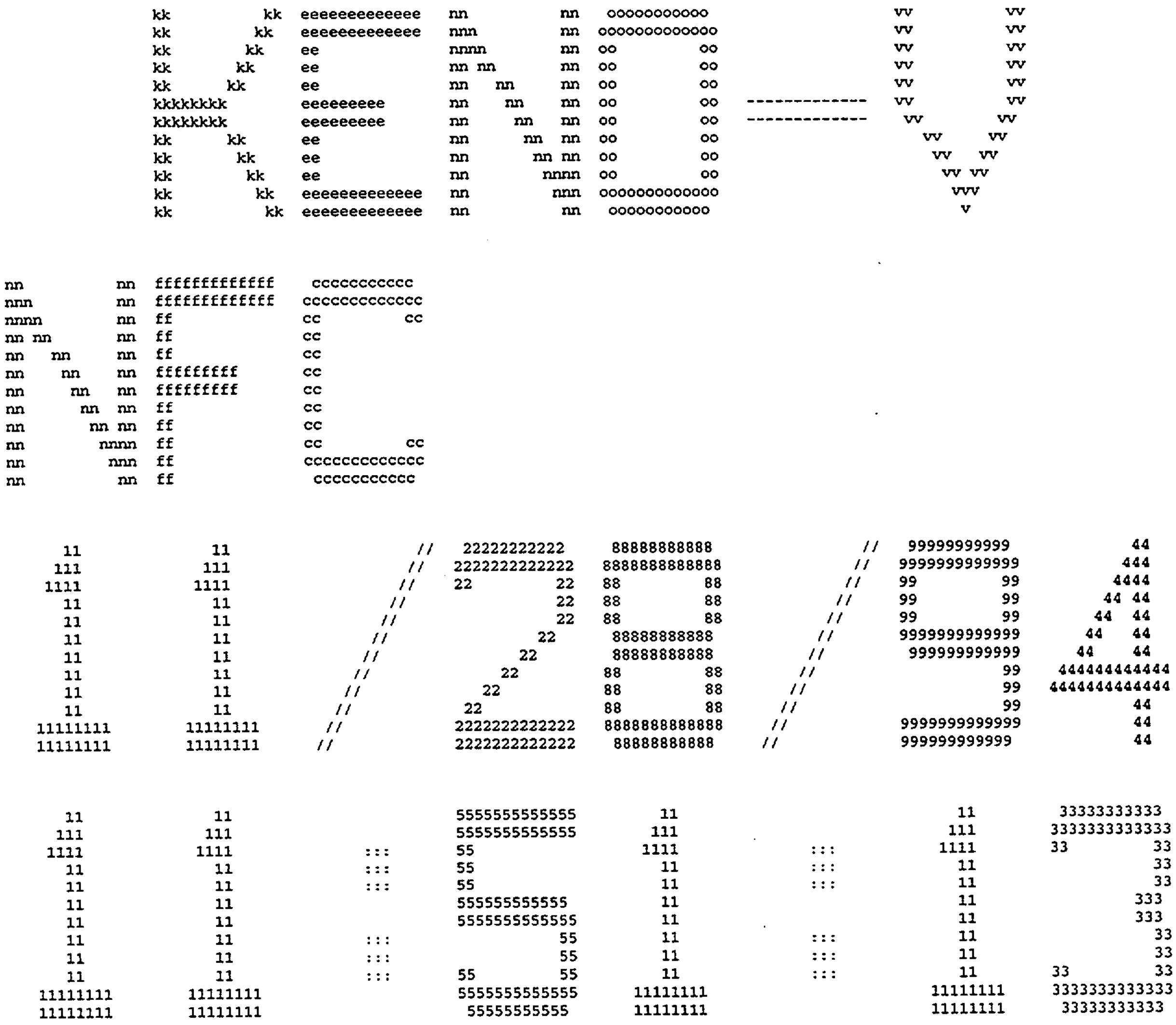
3. J. T. Thomas, *Critical Three-Dimensional Arrays of Neutron-Interacting Units, Part II,* ORNL/TM-868, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., July 1964.

F11.6 DESCRIPTION OF OUTPUT

This section contains a brief description and explanation of the KENO V.a output. Portions of the printout will not be printed for every problem. Some printout is optional and is so noted in this section.

F11.6.1 HEADER PAGE

The first page of print from KENO V.a is the header page. A sample header page is shown in Fig. F11.6.1.



```
              kk        kk  eeeeeeeeeeeee  nn         nn   ooooooooooo                      vv         vv
              kk       kk   eeeeeeeeeeeee  nnn        nn  ooooooooooooo                     vv         vv
              kk      kk    ee             nnnn       nn  oo         oo                     vv         vv
              kk     kk     ee             nn nn      nn  oo         oo                     vv         vv
              kk    kk      ee             nn  nn     nn  oo         oo                     vv         vv
              kkkkkkkk      eeeeeeeee      nn   nn    nn  oo         oo   -------------     vv         vv
              kkkkkkkk      eeeeeeeee      nn    nn   nn  oo         oo   -------------      vv       vv
              kk    kk      ee             nn     nn  nn  oo         oo                       vv     vv
              kk     kk     ee             nn      nn nn  oo         oo                        vv   vv
              kk      kk    ee             nn       nnnn  oo         oo                         vv vv
              kk       kk   eeeeeeeeeeeee  nn        nnn  ooooooooooooo                          vvv
              kk        kk  eeeeeeeeeeeee  nn         nn   ooooooooooo                            v

nn         nn  fffffffffffff   ccccccccccc
nnn        nn  fffffffffffff  ccccccccccccc
nnnn       nn  ff             cc         cc
nn nn      nn  ff             cc
nn  nn     nn  ff             cc
nn   nn    nn  fffffffff      cc
nn    nn   nn  fffffffff      cc
nn     nn  nn  ff             cc
nn      nn nn  ff             cc
nn       nnnn  ff             cc         cc
nn        nnn  ff             ccccccccccccc
nn         nn  ff              ccccccccccc

     11          11                 //   22222222222     88888888888              //   99999999999             44
    111         111                //   2222222222222   8888888888888            //   9999999999999           444
   1111        1111               //    22         22   88         88           //    99         99          4444
     11          11              //                22   88         88          //     99         99         44 44
     11          11             //                 22   88         88         //      99         99        44  44
     11          11            //                22      88888888888         //       9999999999999      44   44
     11          11           //               22        88888888888        //         999999999999     44    44
     11          11          //              22         88         88      //                    99     444444444444
     11          11         //             22           88         88     //                     99    4444444444444
     11          11        //            22             88         88    //                      99             44
   11111111    11111111   //            2222222222222   8888888888888   //          9999999999999             44
   11111111    11111111  //             2222222222222    88888888888   //           999999999999              44

     11          11                   5555555555555       11                    11       33333333333
    111         111                   5555555555555      111                   111      3333333333333
   1111        1111         :::       55                1111        :::       1111      33         33
     11          11         :::       55                  11        :::         11                 33
     11          11         :::       55                  11        :::         11                 33
     11          11                   555555555555        11                    11                333
     11          11                   5555555555555       11                    11                333
     11          11         :::                  55       11        :::         11                 33
     11          11         :::                  55       11        :::         11                 33
     11          11         :::       55         55       11        :::         11      33         33
   11111111    11111111               5555555555555     11111111              11111111  3333333333333
   11111111    11111111                55555555555      11111111              11111111   33333333333
```

Figure F11.6.1 Sample KENO V.a header page

The first line of block letters prints KENO V.a. The second line of block letters gives the job name from the job control language. The third line gives the date the job was run as month/day/year. The last line prints the time execution of the problem was begun (hour/minute/second) in terms of a 24-hour clock with midnight being 2400 hours. The header page is printed from subroutine MESAGE.

F11.6.2 PROGRAM VERIFICATION INFORMATION

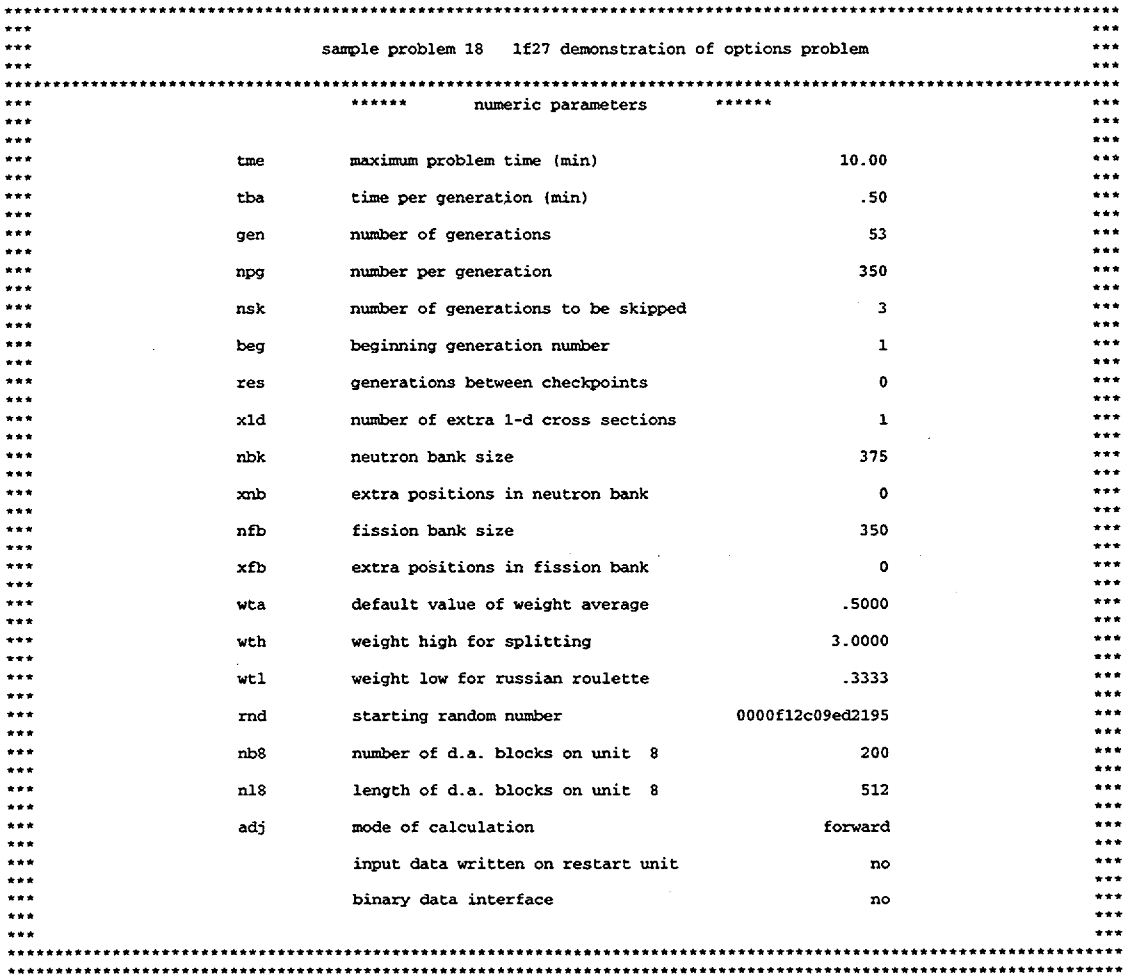
Program verification information, see Fig. F11.6.2, is printed after the header page. It lists the name of the program, the date the load module was created, the library that contains the load module, the computer code name from the configuration control table, and the revision number. The job name, date, and time of execution are also printed. This information may be used for quality assurance purposes.

```
 ssssssssss     cccccccccc      aaaaaaaa     ll             eeeeeeeeeeee
ssssssssssss   cccccccccccc    aaaaaaaaaa    ll             eeeeeeeeeeee
ss        ss   cc        cc   aa        aa   ll             ee
ss             cc             aa        aa   ll             ee
ss             cc             aa        aa   ll             ee
sssssssssss    cc             aaaaaaaaaaaa   ll             eeeeeeeee
 sssssssssss   cc             aaaaaaaaaaaa   ll             eeeeeeeee
          ss   cc             aa        aa   ll             ee
          ss   cc             aa        aa   ll             ee
ss        ss   cc        cc   aa        aa   ll             ee
ssssssssssss   cccccccccccc   aa        aa   llllllllllll   eeeeeeeeeeee
 ssssssssss     cccccccccc    aa        aa   llllllllllll   eeeeeeeeeeee



**************************************************************************
**************************************************************************
**************************************************************************
*****                                                                *****
*****                program verification information                *****
*****                                                                *****
*****               code system:    scale  version:    4.3           *****
*****                                                                *****
**************************************************************************
**************************************************************************
*****                                                                *****
*****                                                                *****
*****            program:  kenova                                    *****
*****                                                                *****
*****      creation date:  11/28/94                                  *****
*****                                                                *****
*****            library:  /scale4.3/bin                             *****
*****                                                                *****
*****                                                                *****
*****    production code:  kenova                                    *****
*****                                                                *****
*****            version:  3.0                                       *****
*****                                                                *****
*****            jobname:  nfc                                       *****
*****                                                                *****
*****  date of execution:  11/28/94                                  *****
*****                                                                *****
*****  time of execution:  11:51:14                                  *****
*****                                                                *****
*****                                                                *****
**************************************************************************
**************************************************************************
**************************************************************************
```

Figure F11.6.2 Sample program verification table

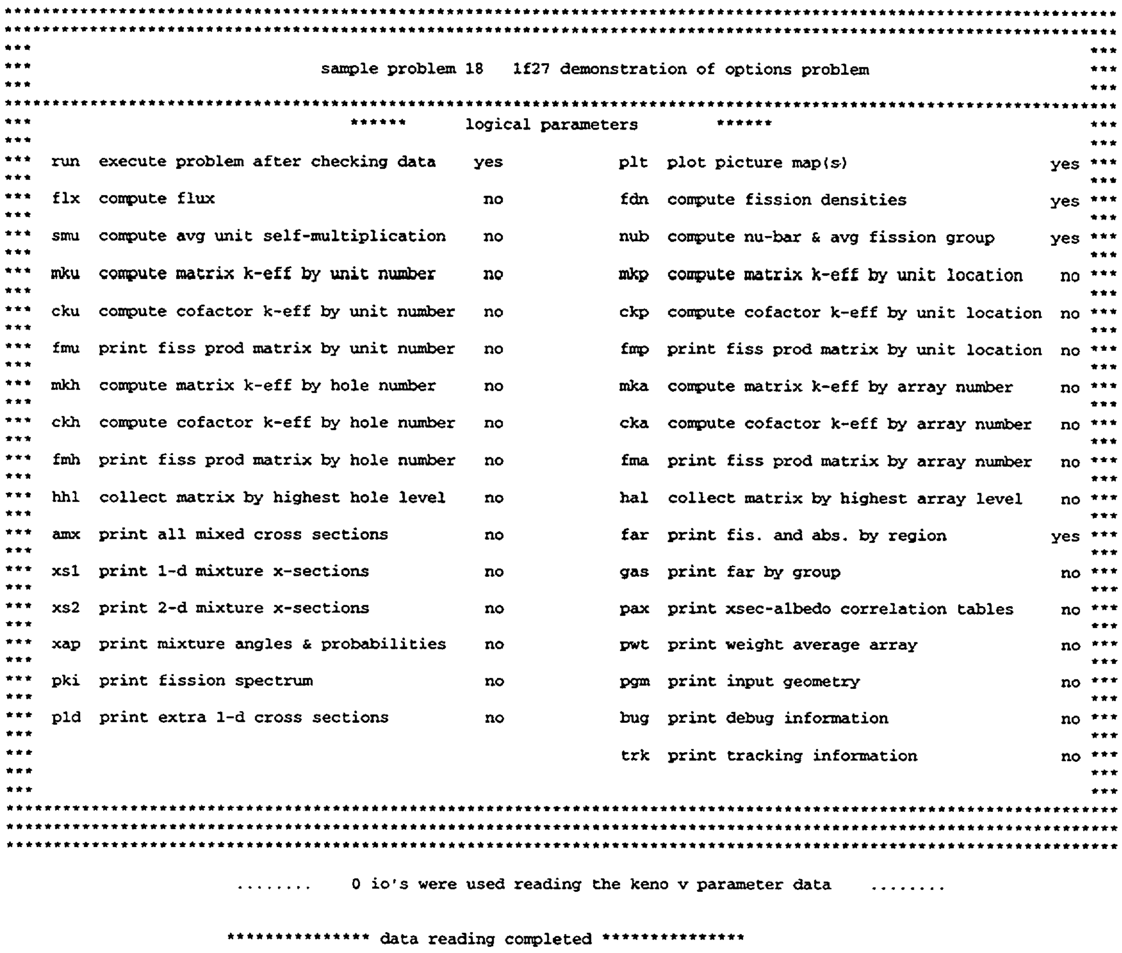
F11.6.3 TABLES OF PARAMETER DATA

The first two tables printed by KENO V.a list the numeric parameters and logical parameters that are used in the problem. They should always be examined by the user to verify that the parameter data block was input as desired. Examples of these tables are shown in Figs. F11.6.3 and F11.6.4.



```
*********************************************************************************************************************
***                                                                                                               ***
***                               sample problem 18   1f27 demonstration of options problem                       ***
***                                                                                                               ***
*********************************************************************************************************************
***                                 ******       numeric parameters       ******                                  ***
***                                                                                                               ***
***                                                                                                               ***
***                      tme        maximum problem time (min)                        10.00                       ***
***                                                                                                               ***
***                      tba        time per generation (min)                           .50                       ***
***                                                                                                               ***
***                      gen        number of generations                                53                       ***
***                                                                                                               ***
***                      npg        number per generation                               350                       ***
***                                                                                                               ***
***                      nsk        number of generations to be skipped                   3                       ***
***                                                                                                               ***
***                      beg        beginning generation number                           1                       ***
***                                                                                                               ***
***                      res        generations between checkpoints                       0                       ***
***                                                                                                               ***
***                      xld        number of extra 1-d cross sections                    1                       ***
***                                                                                                               ***
***                      nbk        neutron bank size                                   375                       ***
***                                                                                                               ***
***                      xnb        extra positions in neutron bank                       0                       ***
***                                                                                                               ***
***                      nfb        fission bank size                                   350                       ***
***                                                                                                               ***
***                      xfb        extra positions in fission bank                       0                       ***
***                                                                                                               ***
***                      wta        default value of weight average                   .5000                       ***
***                                                                                                               ***
***                      wth        weight high for splitting                        3.0000                       ***
***                                                                                                               ***
***                      wtl        weight low for russian roulette                   .3333                       ***
***                                                                                                               ***
***                      rnd        starting random number                 0000f12c09ed2195                       ***
***                                                                                                               ***
***                      nb8        number of d.a. blocks on unit  8                    200                       ***
***                                                                                                               ***
***                      nl8        length of d.a. blocks on unit  8                    512                       ***
***                                                                                                               ***
***                      adj        mode of calculation                             forward                       ***
***                                                                                                               ***
***                                 input data written on restart unit                   no                       ***
***                                                                                                               ***
***                                 binary data interface                                no                       ***
***                                                                                                               ***
***                                                                                                               ***
*********************************************************************************************************************
*********************************************************************************************************************
```

Figure F11.6.3 Sample table of numeric parameter data



```
*************************************************************************************************************
*************************************************************************************************************
***                                                                                                       ***
***                              sample problem 18   1f27 demonstration of options problem                ***
***                                                                                                       ***
*************************************************************************************************************
***                                 ******      logical parameters        ******                          ***
***                                                                                                       ***
***  run  execute problem after checking data    yes            plt  plot picture map(s)              yes ***
***                                                                                                       ***
***  flx  compute flux                            no            fdn  compute fission densities        yes ***
***                                                                                                       ***
***  smu  compute avg unit self-multiplication    no            nub  compute nu-bar & avg fission group   yes ***
***                                                                                                       ***
***  mku  compute matrix k-eff by unit number     no            mkp  compute matrix k-eff by unit location   no ***
***                                                                                                       ***
***  cku  compute cofactor k-eff by unit number   no            ckp  compute cofactor k-eff by unit location  no ***
***                                                                                                       ***
***  fmu  print fiss prod matrix by unit number   no            fmp  print fiss prod matrix by unit location  no ***
***                                                                                                       ***
***  mkh  compute matrix k-eff by hole number     no            mka  compute matrix k-eff by array number    no ***
***                                                                                                       ***
***  ckh  compute cofactor k-eff by hole number   no            cka  compute cofactor k-eff by array number  no ***
***                                                                                                       ***
***  fmh  print fiss prod matrix by hole number   no            fma  print fiss prod matrix by array number  no ***
***                                                                                                       ***
***  hhl  collect matrix by highest hole level    no            hal  collect matrix by highest array level   no ***
***                                                                                                       ***
***  amx  print all mixed cross sections          no            far  print fis. and abs. by region       yes ***
***                                                                                                       ***
***  xs1  print 1-d mixture x-sections            no            gas  print far by group                   no ***
***                                                                                                       ***
***  xs2  print 2-d mixture x-sections            no            pax  print xsec-albedo correlation tables    no ***
***                                                                                                       ***
***  xap  print mixture angles & probabilities    no            pwt  print weight average array           no ***
***                                                                                                       ***
***  pki  print fission spectrum                  no            pgm  print input geometry                 no ***
***                                                                                                       ***
***  pld  print extra 1-d cross sections          no            bug  print debug information              no ***
***                                                                                                       ***
***                                                             trk  print tracking information           no ***
***                                                                                                       ***
***                                                                                                       ***
*************************************************************************************************************
*************************************************************************************************************
*************************************************************************************************************

                        ........    0 io's were used reading the keno v parameter data    ........

                       *************** data reading completed ***************
```

Figure F11.6.4 Sample table of logical parameter data

```
                         sample problem 18   1f27 demonstration of options problem

                                                   geometry description input

unit      1

region

  1 cylinder     1  1  radius =  9.5200       +z=  8.7804       -z= -8.7804

  2 cylinder     0  1  radius =  9.5200       +z=  8.9896       -z= -8.7804

  3 cylinder     2  1  radius =  10.160       +z=  9.6296       -z= -9.4204

  4 cuboid       4  1      +x =  18.450     -x = -18.450      +y =  18.450     -y = -18.450      +z =  17.895      -z = -17.685


unit      2

region

  1 array        1  1      -x = 0.00000e+00 -y = 0.00000e+00  -z = 0.00000e+00


unit      3

region

  1 array        2  1      -x = 0.00000e+00 -y = 0.00000e+00  -z = 0.00000e+00


unit      4

region

  1 array        3  1      -x = 0.00000e+00 -y = 0.00000e+00  -z = 0.00000e+00


unit      5

region

  1 array        4  1      -x = 0.00000e+00 -y = 0.00000e+00  -z = 0.00000e+00


unit      6

region

  1 cuboid       4  1      +x =  55.350     -x = -55.350      +y =  55.350     -y = -55.350      +z =  53.370      -z = -53.370

  6 replicate    3  2  5 region(s), thickness/face=  3.0000      3.0000      3.0000      3.0000      3.0000      3.0000

  7 replicate    3  7  1 region(s), thickness/face=  .24000      .24000      .24000      .24000      .24000      .24000

                           *************** data reading completed ***************
```

Figure F11.6.5 Example of unprocessed geometry input data

The title of the problem is printed at the top of each table. The first table, shown in Fig. F11.6.3, lists the numeric parameter data. It contains a triple column that lists the applicable parameter keyword used to input the data, a brief explanation of its meaning, and the associated data value. The last two entries in this table are slightly different than the others because a keyword is not listed with them. The INPUT DATA WRITTEN ON RESTART UNIT is set YES if a unit number is provided for writing restart data (WRS=). The BINARY DATA INTERFACE is set YES if a unit number is provided for reading restart data (RST=).

The second table, shown in Fig. F11.6.4, lists the logical parameter data. It contains two triple columns, each listing the applicable parameter keyword, a brief explanation of the parameter's function, and the value associated with the parameter. Messages concerning the parameter data may be printed at the bottom of the table. If the problem is one that is being restarted, the title of the parent case is printed at the bottom of the table.

If the restart title or messages are not printed, the bottom section of the table is omitted. If the user desires to change some of the data in these tables, the appropriate parameter keyword must be entered in the parameter data followed by an equal sign and the desired value. Following this table is a statement affirming completion of the parameter input and a statement of the number of I/Os used in reading the parameter data. At this point, the unprocessed input geometry may be printed as described in Sect. F11.6.4. These data are followed by a statement affirming the completion of the data reading.

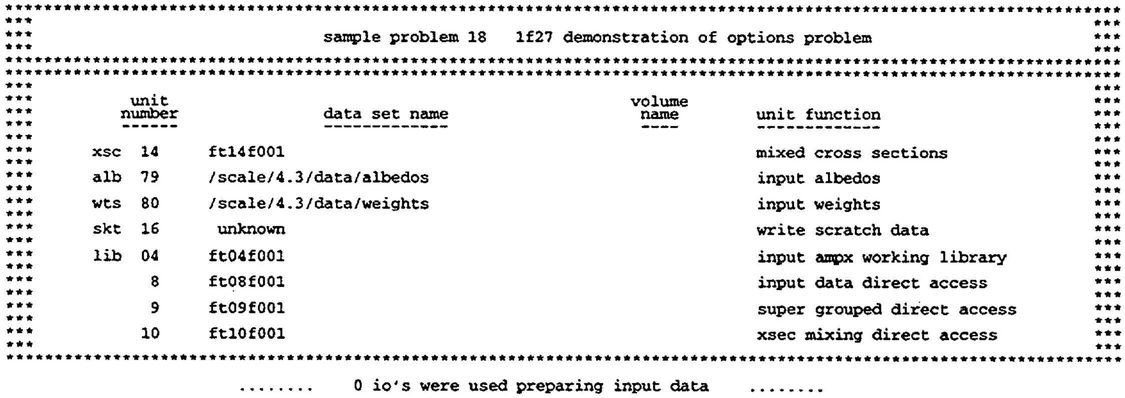
F11.6.4 UNPROCESSED GEOMETRY INPUT DATA

This printout is optional and is usually used to locate code difficulties, to show all the geometry input data when only part of it is used in the problem, or to show the order in which units were entered. It is considered debug information and is printed only if PGM=YES is specified in the parameter input data as described in Sect. F11.4.3. Standard KENO V.a use does not require printing these data because the processed geometry that is used in the problem is always printed. See Sects. F11.6.15 and F11.6.16 for examples of the standard printed KENO V.a geometry data. An example of the data printed in the unprocessed geometry input data is shown in Fig. F11.6.5. The unprocessed geometry is printed by subroutine KENOG.

When the unprocessed geometry input is printed, the problem title is located at the top of the page, followed by the heading "GEOMETRY DESCRIPTION INPUT." The region-dependent geometry information is then printed. If the problem contains a unit orientation array, the problem title is printed again, followed by the unit orientation. This is followed by a statement affirming the completion of the data input.

F11.6.5 TABLE OF DATA SETS USED IN THE PROBLEM

This table is the third table of data printed by KENO V.a. It should be carefully scrutinized to verify the desired data set name is associated with the proper unit number and volume. An example of this table is shown in Fig. F11.6.6.



```
*****************************************************************************************************
***                                                                                               ***
***                         sample problem 18   1f27 demonstration of options problem             ***
***                                                                                               ***
*****************************************************************************************************
*****************************************************************************************************
***                                                                                               ***
***        unit                                          volume                                   ***
***       number            data set name                 name       unit function                ***
***       ------            -------------                 ----       -------------                ***
***                                                                                               ***
***   xsc  14     ft14f001                                           mixed cross sections         ***
***                                                                                               ***
***   alb  79     /scale/4.3/data/albedos                            input albedos                ***
***                                                                                               ***
***   wts  80     /scale/4.3/data/weights                            input weights                ***
***                                                                                               ***
***   skt  16      unknown                                           write scratch data           ***
***                                                                                               ***
***   lib  04     ft04f001                                           input ampx working library   ***
***                                                                                               ***
***         8     ft08f001                                           input data direct access     ***
***                                                                                               ***
***         9     ft09f001                                           super grouped direct access  ***
***                                                                                               ***
***        10     ft10f001                                           xsec mixing direct access    ***
***                                                                                               ***
*****************************************************************************************************

                ........    0 io's were used preparing input data    ........
```

Figure F11.6.6 Sample table of data sets used in the problem

This table, printed from subroutine DATAIN, lists unit numbers that are specified in the parameter data or are defaulted in the code, and information pertinent to them. This information is given in the following order, left to right: (1) the keyword used in the parameter data to define the unit number, (2) the unit number, (3) the data set name, (4) the name of the volume on which the data set resides, and (5) the type of data contained on the data set. This table can be useful for quality assurance purposes. Information for units whose default values have not been overridden are printed even though they may not be used in the problem. Information for every unit specified in the parameter data is also printed. Units 8, 9, and 10 are the direct-access devices and their unit numbers are fixed within the code. When this table is printed, units 9 and 10 have not yet been defined. This causes their data set names to be listed as FT08F001 or as "UNKNOWN" on some systems. If KENO V.a is run as part of a CSAS sequence, this table will include two entries for unit 95, one for binary input data, and one for read restart data. This table is followed by a statement of the number of I/Os used preparing the input data and writing it on the direct-access data sets.

F11.6.6 MIXING TABLE DATA

These data are printed by subroutine **PRTMIX.** If **LIB=** is entered in the KENO parameter data and a mixing table data block is provided to KENO, mixing table data will be printed. It is not considered optional because it cannot be suppressed if the necessary data are present. Sample mixing table data are shown in Fig. F11.6.7.

The data printed in this table include the problem title, the heading "MIXING TABLE," the number of scattering angles, and the cross-section message threshold. The number of scattering angles can be set using the "XSEC PARAMETER" **SCT=**, and the cross-section message threshold can be set using the "XSEC PARAMETER" **EPS=**. For each mixture specified in the problem, the mixture number and associated density is printed, followed by columns of data for each nuclide in the mixture. The column headings, printed below each mixture and density specification, are: "**NUCLIDE**," the nuclide ID number from the working-format library; "**ATOM-DENS.**," the number density of the nuclide in units of atoms/barn-cm.; "**WGT. FRAC.**," the weight fraction of the nuclide (the sum of all of the weight fractions in a mixture should sum to 1.0); "**ZA**," 1000 X Z (the atomic number of the nuclide from the periodic table) plus A (the atomic mass of the nuclide); "**AWT**," the atomic weight of the nuclide; and "**NUCLIDE TITLE**," the nuclide title from the working-format cross-section library used by KENO V.a to run the problem. The pertinent data for each nuclide in the mixture are listed below the column headings. Zeros will be printed for "**WGT. FRAC.**," "**ZA**," and "**AWT**" if the working-format cross-section library does not include the data required to provide that information. After listing the mixing table data are for each mixture, a table of the nuclide IDs and their associated titles are listed in the order they occur on the working-format cross-section library. After these data are printed, a message or messages may be printed if bad moments were encountered during cross-section processing. Then the number of IOs used in cross-section processing is printed. Zero I/Os indicate that the code does not have a method of determining the number of I/Os for the computer used in the calculations. If extra 1-D cross sections were specified in the problem (see X1D=, Sect. F11.4.3), the extra 1-D cross-section IDs will be printed under the heading "1-D CROSS-SECTION ARRAY ID NUMBERS." If $\bar{\nu}$ is to be calculated (see NUB=, Sect. F11.4.3), six ID numbers will be printed. The ID number for the total cross section (Σ_T) is 1; the ID number for the sum of the transfer array normalized by Σ_T is 2002; the ID number for the normalized fission-product cross section ($\nu\Sigma_f / \Sigma_t$) is 1452; the ID number for the normalized absorption cross section (Σ_{abs}/Σ_T) is 27; the ID number for the normalized fission cross section (Σ_f/Σ_T) is 18; the ID number for the fission spectrum (χ) is 1018. χ is summed and normalized to 1.0. Other ID numbers that appear in this list have been specified by the user. If the number of blocks on the direct access data set are insufficient to hold the cross-section data, a message is printed stating: THE NUMBER OF DIRECT ACCESS BLOCKS ON UNIT ____ HAS BEEN INCREASED TO ____. If the problem is to write a restart data set (RES=, Sect. F11.4.3), a message is printed stating that restart information was written and the restart I/O unit number is specified. This is followed by a statement of the number of I/Os used in preparing the cross sections. The user should examine the mixing table carefully to verify that the proper nuclides are specified for the proper mixtures and that all the data are correct.

```
                              sample problem 18   1f27 demonstration of options problem

                                             mixing table

                                   number of scattering angles =  2
                                 cross section message threshold =3.0e-05


mixture =      1           density(g/cc) =  1.5550
   nuclide   atom-dens.   wgt. frac.     za       awt                 nuclide title
   1001001  5.78164e-02  6.22143e-02    1001      1.0077      hydrogen       endf/b-iv mat 1269/thrml002                 updated 08/12/94
   1007014  2.13092e-03  3.18658e-02    7014     14.0033      nitrogen-14    endf/b-iv mat 1275                          updated 08/12/94
   1008016  3.74230e-02  6.39034e-01    8016     15.9904      oxygen-16      endf/b-iv mat 1276                          updated 08/12/94
   1092234  1.06784e-05  2.66886e-03   92234    234.0405      uranium-234    endf/b-iv mat 1043                          updated 08/12/94
   1092235  9.84601e-04  2.47136e-01   92235    235.0441      uranium-235    endf/b-iv mat 1261                          updated 08/12/94
   1092236  5.29386e-06  1.33443e-03   92236    236.0458     u-236 1163 sigo=5+4 newxlacs p-3 293k f-1/e-m(1.+5)         updated 08/12/94
   1092238  6.19414e-05  1.57463e-02   92238    238.0510      uranium-238    endf/b-iv mat 1262                          updated 08/12/94

mixture =      2           density(g/cc) =  1.1800
   nuclide   atom-dens.   wgt. frac.     za       awt                 nuclide title
   2001001  5.68242e-02  8.05783e-02    1001      1.0077      hydrogen       endf/b-iv mat 1269/thrml002                 updated 08/12/94
   2006012  3.55151e-02  5.99750e-01    6000     12.0001      carbon-12      endf/b-iv mat 1274/thrml065                 updated 08/12/94
   2008016  1.42060e-02  3.19672e-01    8016     15.9904      oxygen-16      endf/b-iv mat 1276                          updated 08/12/94

mixture =      3           density(g/cc) =  .89999
   nuclide   atom-dens.   wgt. frac.     za       awt                 nuclide title
   3001001  7.99749e-02  1.48689e-01    1001      1.0077      hydrogen       endf/b-iv mat 1269/thrml002                 updated 08/12/94
   3006012  3.84495e-02  8.51311e-01    6000     12.0001      carbon-12      endf/b-iv mat 1274/thrml065                 updated 08/12/94

mixture =      4           density(g/cc) =  .99817e-09
   nuclide   atom-dens.   wgt. frac.     za       awt                 nuclide title
   4001001  6.67692e-11  1.11927e-01    1001      1.0077      hydrogen       endf/b-iv mat 1269/thrml002                 updated 08/12/94
   4008016  3.33846e-11  8.88074e-01    8016     15.9904      oxygen-16      endf/b-iv mat 1276                          updated 08/12/94


                              1001001     hydrogen       endf/b-iv mat 1269/thrml002           updated 08/12/94
                              2001001     hydrogen       endf/b-iv mat 1269/thrml002           updated 08/12/94
                              3001001     hydrogen       endf/b-iv mat 1269/thrml002           updated 08/12/94
                              4001001     hydrogen       endf/b-iv mat 1269/thrml002           updated 08/12/94
                              2006012     carbon-12      endf/b-iv mat 1274/thrml065           updated 08/12/94
                              3006012     carbon-12      endf/b-iv mat 1274/thrml065           updated 08/12/94
                              1007014     nitrogen-14    endf/b-iv mat 1275                    updated 08/12/94
                              1008016     oxygen-16      endf/b-iv mat 1276                    updated 08/12/94
                              2008016     oxygen-16      endf/b-iv mat 1276                    updated 08/12/94
                              4008016     oxygen-16      endf/b-iv mat 1276                    updated 08/12/94
                              1092234     uranium-234    endf/b-iv mat 1043                    updated 08/12/94
                              1092235     uranium-235    endf/b-iv mat 1261                    updated 08/12/94
                              1092236    u-236 1163 sigo=5+4 newxlacs p-3 293k f-1/e-m(1.+5)   updated 08/12/94
                              1092238     uranium-238    endf/b-iv mat 1262                    updated 08/12/94

keno message number k5-222       1 transfers for mixture     1 were corrected for bad moments.

keno message number k5-222       2 transfers for mixture     2 were corrected for bad moments.

keno message number k5-222       3 transfers for mixture     4 were corrected for bad moments.

keno message number k5-222       2 transfers for mixture     3 were corrected for bad moments.

                                 .......    0 io's were used mixing cross-sections       .......

                                  1-d cross section array id numbers
                                     1  2002  1452    27    18  1018

                                 .......    0 io's were used preparing the cross sections     .......
```

Figure F11.6.7 Example of mixing table data

F11.6.7 ALBEDO CROSS-SECTION CORRESPONDENCE

Printing the albedo cross-section correspondence tables is optional. The headings for the tables are printed in subroutine CORRE, then subroutine RATIO prints the data. These tables are printed only if PAX=YES is specified in the parameter data as described in Sect. F11.4.3. Examples of these tables are shown in Figs. F11.6.8 and F11.6.9.

The table shown in Fig. F11.6.8 contains, left to right, the cross-section energy group, the lower and upper lethargy bounds, the corresponding albedo energy groups, and the cumulative probability associated with each albedo energy group for choosing the albedo energy group corresponding to the cross-section energy group. The table shown in Fig. F11.6.9 is the inverse of the table shown in Fig. F11.6.8. It provides the cumulative probabilities for choosing the cross-section energy group corresponding to the albedo energy group. The information in these tables is automatically generated by KENO V.a.

```
                                       cumulative probabilities for choosing the corresponding albedo group for each xsec group
 xsec                                 ---------------------------------------------------------------------------------------
energy     cross section group          albedo                 albedo                 albedo                 albedo
 group     lethargy boundaries          group  probability     group  probability     group  probability     group  probability
          --------------------------   -------------------    -------------------    -------------------    -------------------

   1        .00000e+00    .12040e+01     1      1.00000
   2        .12040e+01    .19661e+01     2      1.00000
   3        .19661e+01    .24079e+01     3      1.00000
   4        .24079e+01    .32189e+01     4      1.00000
   5        .32189e+01    .46052e+01     5      1.00000
   6        .46052e+01    .63771e+01     6      1.00000
   7        .63771e+01    .81117e+01     7      1.00000
   8        .81117e+01    .98082e+01     8      1.00000
   9        .98082e+01    .11513e+02     9      1.00000
  10        .11513e+02    .12717e+02    10      1.00000
  11        .12717e+02    .13816e+02    11      1.00000
  12        .13816e+02    .15019e+02    12      1.00000
  13        .15019e+02    .16118e+02    13      1.00000
  14        .16118e+02    .17034e+02    14      1.00000
  15        .17034e+02    .18421e+02    15      1.00000
  16        .18421e+02    .19807e+02    16      1.00000
```

Figure F11.6.8 Cumulative probabilities for correlating the albedo energy group to the cross-section energy group

```
                                       cumulative probabilities for choosing the corresponding xsec group for each albedo group
albedo                                ---------------------------------------------------------------------------------------
energy       albedo group               xsec                   xsec                   xsec                   xsec
 group     lethargy boundaries          group  probability     group  probability     group  probability     group  probability
          --------------------------   -------------------    -------------------    -------------------    -------------------

   1       -.40547e+00    .12040e+01     1      1.00000
   2        .12040e+01    .19661e+01     2      1.00000
   3        .19661e+01    .24079e+01     3      1.00000
   4        .24079e+01    .32189e+01     4      1.00000
   5        .32189e+01    .46052e+01     5      1.00000
   6        .46052e+01    .63771e+01     6      1.00000
   7        .63771e+01    .81117e+01     7      1.00000
   8        .81117e+01    .98082e+01     8      1.00000
   9        .98082e+01    .11513e+02     9      1.00000
  10        .11513e+02    .12717e+02    10      1.00000
  11        .12717e+02    .13816e+02    11      1.00000
  12        .13816e+02    .15019e+02    12      1.00000
  13        .15019e+02    .16118e+02    13      1.00000
  14        .16118e+02    .17034e+02    14      1.00000
  15        .17034e+02    .18421e+02    15      1.00000
  16        .18421e+02    .23026e+02    16      1.00000
```

Figure F11.6.9 Cumulative probabilities for correlating the cross-section energy group to the albedo energy group

F11.6.8 1-D MACROSCOPIC CROSS SECTIONS

The decision to print the 1-D mixture cross sections is optional. They are printed in subroutine PRT1DS only if XS1=YES is specified in the parameter data as described in Sect. F11.4.3. When the 1-D cross sections are to be printed, they are printed a group at a time for each mixture. The 1-D mixture cross sections for a mixture are shown in Fig. F11.6.10.

```
                     sample problem 18   1f27 demonstration of options problem

         1 =id          material=    1

group     sgt         nap         abp         nfp         chi              mwa1        mwa2        mwa3         sum
   1  1.39944e-01 9.76695e-01 2.33045e-02 2.59767e-02 2.04000e-01           1           6           1      1.00000e+00
   2  1.77770e-01 9.91346e-01 8.65391e-03 1.82268e-02 5.48000e-01           7          12           2      1.00000e+00
   3  3.19702e-01 9.95623e-01 4.37705e-03 9.53099e-03 7.16000e-01          13          18           3      1.00000e+00
   4  3.83997e-01 9.96305e-01 3.69464e-03 7.65954e-03 8.96000e-01          19          24           4      1.00000e+00
   5  5.44520e-01 9.96969e-01 3.03092e-03 6.35962e-03 9.86000e-01          25          30           5      1.00000e+00
   6  8.24261e-01 9.96191e-01 3.80898e-03 7.31460e-03 1.00000e+00          31          36           6      1.00000e+00
   7  9.89720e-01 9.94467e-01 5.53306e-03 1.02342e-02 1.00000e+00          37          42           7      1.00000e+00
   8  1.04405e+00 9.89315e-01 1.06849e-02 1.82530e-02 1.00000e+00          43          48           8      1.00000e+00
   9  1.06120e+00 9.73740e-01 2.62602e-02 4.24972e-02 1.00000e+00          49          54           9      1.00000e+00
  10  1.09463e+00 9.45115e-01 5.48852e-02 8.26186e-02 1.00000e+00          55          60          10      1.00000e+00
  11  1.09554e+00 9.44429e-01 5.55713e-02 7.55643e-02 1.00000e+00          61          66          11      1.00000e+00
  12  1.10314e+00 9.37344e-01 6.26558e-02 6.09451e-02 1.00000e+00          67          71          12      1.00000e+00
  13  1.07433e+00 9.61400e-01 3.85995e-02 6.73441e-02 1.00000e+00          72          75          13      1.00000e+00
  14  1.28920e+00 9.35180e-01 6.48206e-02 1.30946e-01 1.00000e+00          76          78          14      1.00000e+00
  15  1.83807e+00 8.75203e-01 1.24797e-01 2.42731e-01 1.00000e+00          79          80          15      1.00000e+00
  16  3.38272e+00 8.16191e-01 1.83809e-01 3.67874e-01 1.00000e+00          81          81          16      1.00000e+00

 xsec id      18
 grp.
   1     8.85189e-03
   2     6.96651e-03
   3     3.76749e-03
   4     3.07612e-03
   5     2.58521e-03
   6     2.98555e-03
   7     4.17721e-03
   8     7.45021e-03
   9     1.73458e-02
  10     3.37219e-02
  11     3.08426e-02
  12     2.48756e-02
  13     2.74874e-02
  14     5.34473e-02
  15     9.90739e-02
  16     1.50153e-01
```

Figure F11.6.10 Example of macroscopic 1-D cross sections

When the 1-D mixture cross sections are printed, the problem title is printed at the top of the page. The mixture ID and MATERIAL numbers are then printed. ID is the mixture number from the mixing table and MATERIAL is the index used to reference it. This step is followed by a heading to identify the different 1-D cross sections. GROUP is the energy group, SGT is the total cross section for the mixture, NAP is the nonabsorption probability, ABP is the absorption probability, NFP is the production probability, CHI is the fission spectrum, MWA1 is the pointer for the first position of the cross sections for the energy group, MWA2 is the pointer for the last position of the cross sections for the energy group, and MWA3 contains the group for the transfer corresponding to the first position. SUM is the sum of the absorption probability and the nonabsorption probability. The absorption probability is defined as the absorption cross section divided by the total cross section. The nonabsorption probability is the sum of the group-to-group transfers for this group, divided by the total cross section. The production probability is defined as the fission production cross section divided by the total cross section ($\nu\Sigma_f/\Sigma_T$). The nonabsorption probability and the production probability are not true probabilities in that they may be greater than 1. This is because the nonabsorption probability has the (n,2n) transfer array summed into the total transfer array twice, and the (n,3n) is summed three times, etc.

F11.6.9 EXTRA 1-D CROSS SECTIONS

Printing the extra 1-D cross sections is optional. They are printed in subroutine PRT1DS is P1D=YES is specified in the parameter data (Sect. F11.4.3). Extra 1-D cross sections are not used in KENO V.a unless NUB=YES is specified in the parameter data or the user has altered the code to access and utilize other 1-D cross sections. If NUB=YES is specified, the extra 1-D cross section is the fission cross section, and is used to calculate the average number of neutrons per fission. This is printed only for fissile mixtures as shown in Fig. F11.6.10. The fission cross-section heading is XSEC ID 18 and it follows the table of 1-D cross sections.

F11.6.10 2-D MACROSCOPIC CROSS SECTIONS

The decision to print the 2-D mixture cross sections is optional. They are printed in subroutine PRT2DS only if XS2=YES is specified in the parameter data. They are printed after the 1-D cross sections for the mixture. A heading is printed, followed by the transfer data. An example of the 2-D mixture cross sections is given in Fig. F11.6.11.

```
  scattering transfer array for material     1

     from   grp   1      grp   2      grp   3      grp   4      grp   5      grp   6      grp   7      grp   8      grp   9      grp  10
 to grp
   + 0     2.54209e-01 3.10539e-01 2.98182e-01 3.71084e-01 4.55319e-01 4.72411e-01 4.56823e-01 4.47266e-01 4.48962e-01 3.30557e-01
   + 1     4.48688e-01 2.73107e-01 4.37470e-01 4.85849e-01 4.56102e-01 4.37482e-01 4.45877e-01 4.54113e-01 3.88724e-01 4.51241e-01
   + 2     1.04476e-01 2.31550e-01 1.97693e-01 1.18779e-01 7.31741e-02 7.39104e-02 7.95987e-02 6.85965e-02 1.07909e-01 1.52634e-01
   + 3     1.06680e-01 1.39556e-01 5.49774e-02 2.00378e-02 1.25166e-02 1.32261e-02 1.23322e-02 2.02416e-02 3.82178e-02 4.37863e-02
   + 4     6.46550e-02 3.86576e-02 9.48767e-03 3.33964e-03 2.35355e-03 2.12183e-03 3.59935e-03 6.85965e-03 1.07909e-02 1.30236e-02
   + 5     2.12918e-02 6.59086e-03 2.18946e-03 9.10810e-04 5.34899e-04 8.48732e-04 1.77017e-03 2.92379e-03 5.39546e-03 8.75725e-03

     from   grp  11      grp  12      grp  13      grp  14      grp  15      grp  16
 to grp
   + 0     3.02946e-01 3.19203e-01 2.89136e-01 3.46347e-01 8.10619e-01 1.00000e+00
   + 1     4.94030e-01 4.59900e-01 4.33884e-01 4.94691e-01 1.89381e-01
   + 2     1.35724e-01 1.32774e-01 2.07145e-01 1.58962e-01
   + 3     4.03578e-02 6.63872e-02 6.98356e-02
   + 4     2.02070e-02 2.17359e-02
   + 5     6.73566e-03
```

Figure F11.6.11 Example of 2-D macroscopic cross sections

F11.6.11 PROBABILITIES AND ANGLES

Printing the probabilities and angles is optional. They are printed if the number of scattering angles is greater than zero and XAP=YES is specified in the parameter data as described in Sect. F11.4.3. The probabilities and angles are printed for each mixture by subroutine PRT2DS. An example of the probabilities is shown in Fig. F11.6.12 and an example of the angles is shown in Fig. F11.6.13. If the group-to-group transfer for a mixture is isotropic, the first angle for that transfer will be set to -2.0 as a flag to the code.

```
 probability   1  array for material     1

     from   grp   1      grp   2      grp   3      grp   4      grp   5      grp   6      grp   7      grp   8      grp   9      grp  10
to grp
  +  0     1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00
       transfers from grp+  1 to grp+  4 are the same as above
  +  5     1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00

     from   grp  11      grp  12      grp  13      grp  14      grp  15      grp  16
to grp
  +  0     1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00
  +  1     1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00
  +  2     1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00
  +  3     1.00000e+00 1.00000e+00 1.00000e+00
  +  4     1.00000e+00 1.00000e+00
  +  5     1.00000e+00
```

Figure F11.6.12 Example of macroscopic probabilities

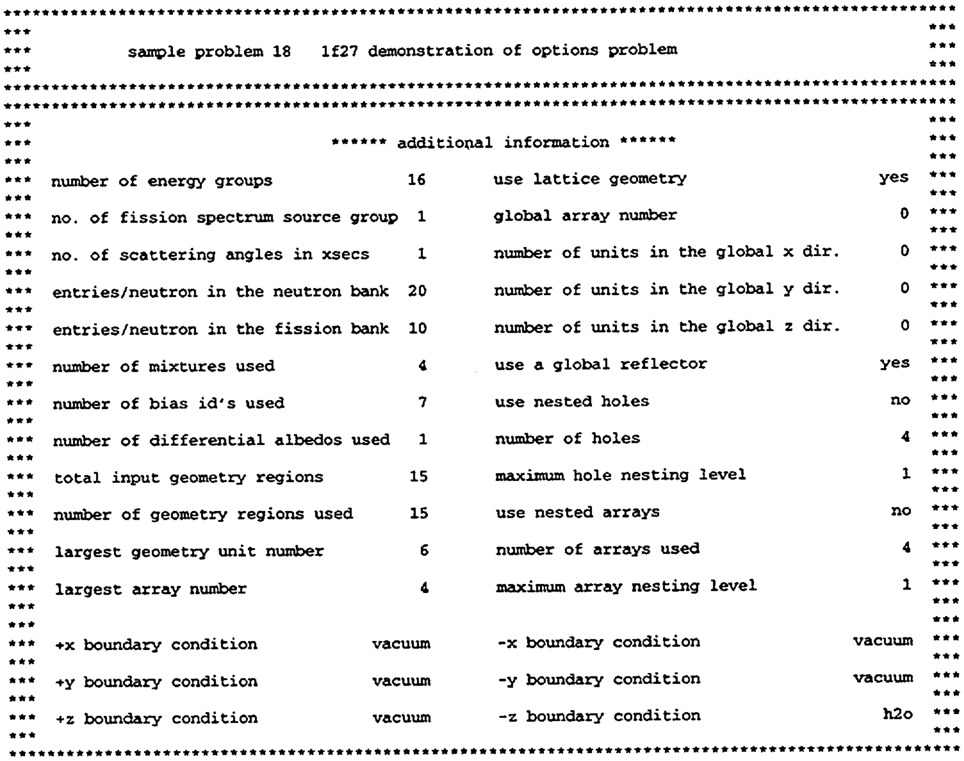
```
 angle   1  array for material     1

     from   grp   1      grp   2      grp   3      grp   4      grp   5      grp   6      grp   7      grp   8      grp   9      grp  10
to grp
  +  0     2.50717e-02 1.37955e-01 1.34616e-02 7.28156e-02 2.88056e-01 4.28189e-01 4.40454e-01 4.42506e-01 4.43150e-01 3.70214e-01
  +  1     5.38829e-01 6.16596e-01 5.69298e-01 5.77713e-01 5.52605e-01 5.45150e-01 5.49196e-01 5.50241e-01 5.78041e-01 6.17541e-01
  +  2     5.20566e-01 5.65012e-01 4.67921e-01 3.16948e-01 2.26608e-01 2.37321e-01 2.40919e-01 2.54098e-01 3.31771e-01 3.63001e-01
  +  3     3.82358e-01 3.49483e-01 2.25674e-01 1.21212e-01 1.02564e-01 1.01604e-01 1.10048e-01 1.41667e-01 1.88235e-01 2.02564e-01
  +  4     2.36583e-01 1.70493e-01 9.61538e-02 4.54545e-02 4.54545e-02 3.33333e-02 6.55738e-02 8.19672e-02 1.04167e-01 1.20690e-01
  +  5     1.19735e-01 5.00000e-02-2.00000e+00-2.00000e+00-2.00000e+00-2.00000e+00 3.33333e-02 3.84615e-02 6.25000e-02 7.05128e-02

     from   grp  11      grp  12      grp  13      grp  14      grp  15      grp  16
to grp
  +  0     3.37214e-01 3.36073e-01 3.09208e-01 5.56326e-01 4.45358e-01 2.52418e-01
  +  1     6.18647e-01 6.19905e-01 6.40449e-01 6.23199e-01 3.81259e-01
  +  2     3.56907e-01 3.79019e-01 4.00109e-01 1.81818e-01
  +  3     2.11405e-01 2.26734e-01 1.28824e-01
  +  4     1.25000e-01 7.23514e-02
  +  5     4.16667e-02
```

Figure F11.6.13 Example of macroscopic angles

F11.6.12 TABLE OF ADDITIONAL INFORMATION

This is the fourth table of data printed by KENO V.a. It contains information determined by the input data and is printed by subroutine NSUPG. An example of this table is shown in Fig. F11.6.14.

This table should be examined by the user to verify the problem data. The NUMBER OF ENERGY GROUPS is read either from the Monte Carlo formatted library, identified by the keyword XSC and the unit function name MIXED CROSS SECTIONS from the Table of Data Sets in Sect. F11.6.5, or from the restart unit, identified by the keyword RST and the unit function name, READ RESTART DATA. The NO. OF FISSION SPECTRUM SOURCE GROUP is the number of different energy groups for which a fission spectrum is defined. In the present version, this number should always be 1. The NO. OF SCATTERING ANGLES IN XSECS is the number of scattering angles to be used in processing the cross sections. The default value is one, and may be overridden by specifying the parameter SCT= in the mixing table input as described in Sect. F11.4.10. One scattering angle yields P_1 cross sections, two scattering angles yield P_3 cross sections, three scattering angles yield P_5 cross sections, etc. ENTRIES/NEUTRON IN THE NEUTRON BANK specifies the number of pieces of data that are banked for each history during tracking. ENTRIES/NEUTRON IN THE FISSION BANK is the amount of data stored for each source neutron for each generation.



```
*****************************************************************************************
***                                                                                  ***
***         sample problem 18   1f27 demonstration of options problem               ***
***                                                                                  ***
*****************************************************************************************
*****************************************************************************************
***                                                                                  ***
***                        ****** additional information ******                      ***
***                                                                                  ***
***  number of energy groups             16       use lattice geometry          yes  ***
***                                                                                  ***
***  no. of fission spectrum source group  1      global array number             0  ***
***                                                                                  ***
***  no. of scattering angles in xsecs     1      number of units in the global x dir.   0  ***
***                                                                                  ***
***  entries/neutron in the neutron bank  20      number of units in the global y dir.   0  ***
***                                                                                  ***
***  entries/neutron in the fission bank  10      number of units in the global z dir.   0  ***
***                                                                                  ***
***  number of mixtures used               4      use a global reflector        yes  ***
***                                                                                  ***
***  number of bias id's used              7      use nested holes               no  ***
***                                                                                  ***
***  number of differential albedos used   1      number of holes                 4  ***
***                                                                                  ***
***  total input geometry regions         15      maximum hole nesting level      1  ***
***                                                                                  ***
***  number of geometry regions used      15      use nested arrays              no  ***
***                                                                                  ***
***  largest geometry unit number          6      number of arrays used           4  ***
***                                                                                  ***
***  largest array number                  4      maximum array nesting level     1  ***
***                                                                                  ***
***                                                                                  ***
***  +x boundary condition          vacuum      -x boundary condition        vacuum  ***
***                                                                                  ***
***  +y boundary condition          vacuum      -y boundary condition        vacuum  ***
***                                                                                  ***
***  +z boundary condition          vacuum      -z boundary condition           h2o  ***
***                                                                                  ***
*****************************************************************************************
```

Figure F11.6.14 Sample table of additional information

The NUMBER OF MIXTURES USED is the number of different mixtures (media) used in the geometry data utilized by the problem. This may be less than the total number of different mixtures specified in the geometry data if portions of the geometry data are not used in the problem.

The NUMBER OF BIAS ID'S USED is the number of different biasing regions used in the problem. This will always be one unless a biasing data block is entered in the problem as described in Sect. F11.4.7.

The NUMBER OF DIFFERENTIAL ALBEDOS USED is the number of different differential albedo reflectors used in the problem. This will always be zero unless the boundary condition data specify the use of differential albedo reflection on one or more faces of the system as described in Sect. F11.4.6. Also check the BOUNDARY CONDITION data printed in this table. The number of different differential albedos specified on the faces should be consistent with the NUMBER OF DIFFERENTIAL ALBEDOS USED. Specular, mirror, vacuum, and periodic are not differential albedos. Be aware that several different keywords may be used to specify the same differential albedo. See Table F11.4.4 for a list of differential albedo keywords.

The TOTAL INPUT GEOMETRY REGIONS is the number of geometry regions specified in the problem input. This excludes UNIT and BOX TYPE, but includes the core boundary description. It excludes the automatic reflector description but includes the geometry regions generated by it. The NUMBER OF GEOMETRY REGIONS USED is the number of geometry regions used in the problem. It may be less than or equal to the TOTAL INPUT GEOMETRY REGIONS. The LARGEST GEOMETRY UNIT NUMBER is the largest unit or box type number defined in the geometry data, including unused units and implicitly defined units. Implicitly defined unit numbers are created when a core boundary specification is not immediately preceded by a unit or box type specification. The unit number is assigned by the code by adding one to the largest unit number encountered in the geometry region data. For example, if two such core boundary specifications are contained in a problem whose largest explicitly defined unit number is 7, a unit number of 8 is assigned to the first one and a unit number of 9 is assigned to the second one. A value of 9 would be printed for the LARGEST GEOMETRY UNIT NUMBER. The LARGEST ARRAY NUMBER is the largest array number specified in the array data. (See Sect. F11.4.5.)

USE LATTICE GEOMETRY is determined by the logical flag that indicates whether or not the problem is a single unit problem. This should be YES for any problem that is not a single unit problem and NO for a single unit problem. By definition, a single unit problem is a problem that does not utilize array data in any form. Section F11.4.5 describes array data. The GLOBAL ARRAY NUMBER is the number of the array designated as the global, overall, or universal array. The global array can be thought of as the array that defines the overall system.

The NUMBER OF UNITS IN THE GLOBAL X/Y/Z DIR. defines the size of the global array in terms of the number of units and/or box types that are located along the edge of the array boundaries in the x/y/z directions. For a single unit, all three of these should be zero. For a simple 1 ×1 × 1 array consisting of one unit type, all three of these numbers should be 1.

USE GLOBAL REFLECTOR indicates if the global array is reflected.

USE NESTED HOLES is set YES if holes are nested deeper than one level.

NUMBER OF HOLES is the number of HOLES that are entered in the geometry region data (Sect. F11.4.4).

The MAXIMUM HOLE NESTING LEVEL is the deepest level of hole nesting.

USE NESTED ARRAYS is set YES if arrays are nested deeper than one level.

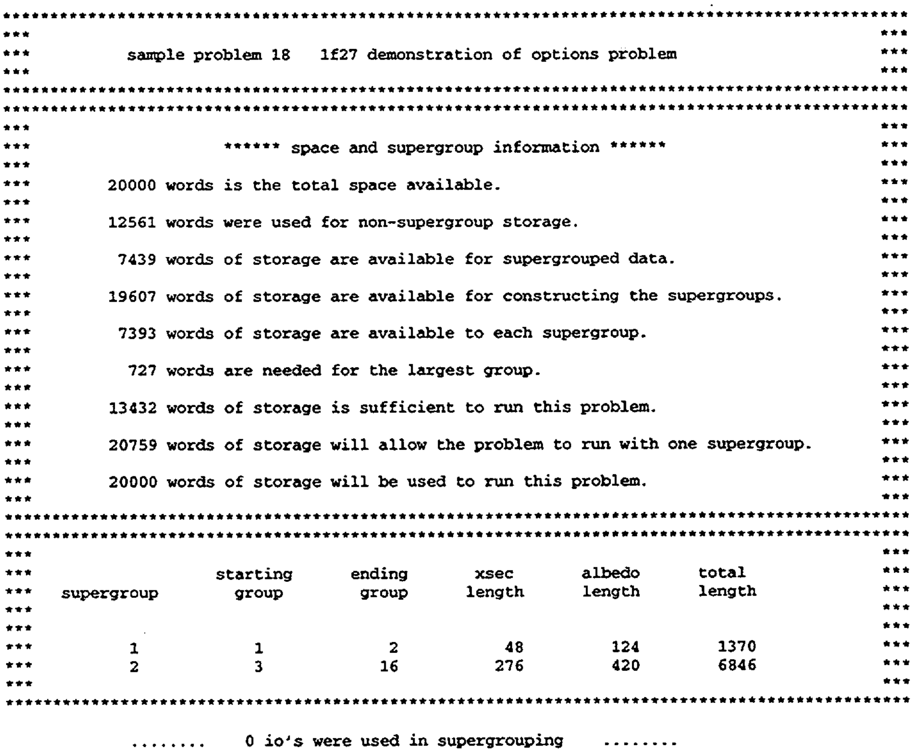
The NUMBER OF ARRAYS USED is the number of array descriptions (Sect. F11.4.5) actually used in the problem description.

MAXIMUM ARRAY NESTING LEVEL is the deepest level of array nesting.

Six BOUNDARY CONDITIONs are printed near the bottom of the table. They show the type of boundary condition that is applied to each face of the system. These should all be VACUUM unless albedo boundary conditions are applied to one or more faces of the system as described in Sect. F11.4.6. Also refer to the NUMBER OF DIFFERENTIAL ALBEDOS USED, discussed previously in the description of this table of information.

F11.6.13 TABLE OF SPACE AND SUPERGROUP INFORMATION

Figure F11.6.15 is the fifth figure printed by KENO V.a. It summarizes the space requirements for the problem and prints information related to the supergroups. This figure is printed by subroutines NSUPG and LIMLN.



```
*********************************************************************************************
***                                                                                       ***
***          sample problem 18   1f27 demonstration of options problem                    ***
***                                                                                       ***
*********************************************************************************************
*********************************************************************************************
***                                                                                       ***
***                       ****** space and supergroup information ******                  ***
***                                                                                       ***
***        20000 words is the total space available.                                      ***
***                                                                                       ***
***        12561 words were used for non-supergroup storage.                              ***
***                                                                                       ***
***         7439 words of storage are available for supergrouped data.                    ***
***                                                                                       ***
***        19607 words of storage are available for constructing the supergroups.         ***
***                                                                                       ***
***         7393 words of storage are available to each supergroup.                       ***
***                                                                                       ***
***          727 words are needed for the largest group.                                  ***
***                                                                                       ***
***        13432 words of storage is sufficient to run this problem.                      ***
***                                                                                       ***
***        20759 words of storage will allow the problem to run with one supergroup.      ***
***                                                                                       ***
***        20000 words of storage will be used to run this problem.                       ***
***                                                                                       ***
*********************************************************************************************
*********************************************************************************************
***                                                                                       ***
***                     starting      ending       xsec       albedo      total           ***
***   supergroup          group        group      length      length      length          ***
***                                                                                       ***
***                                                                                       ***
***        1                1            2          48         124         1370           ***
***        2                3           16         276         420         6846           ***
***                                                                                       ***
*********************************************************************************************

             ........    0 io's were used in supergrouping    ........
```

Figure F11.6.15 Sample table of space and supergroup information

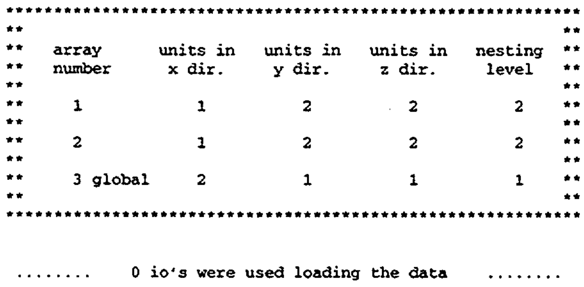
This table contains information about the space requirements of the problem and the number of supergroups used in the problem. The table is basically self-explanatory.

The TOTAL SPACE AVAILABLE is the amount of memory, in words, available to contain the data for the problem. The NONSUPERGROUP STORAGE is the number of words of memory containing nonsupergrouped data. Following the supergroup data are statements concerning the amount of storage available for supergrouping, the amount of storage available for each supergroup, the number of words of storage needed for the largest supergroup, the amount of storage needed to run the problem, and the amount of storage that would be necessary to run the problem with one supergroup.

The bottom portion of this table contains information about the supergroups. Warning and error messages may appear in this portion of the table. The printed supergroup data include the supergroup number, the first energy group in the supergroup, the last energy group in the supergroup, the length (in words) of the cross-section data for the supergroup, the length (in words) of the albedo data for the supergroup, and the total length (in words) of all the data for the supergroup. This information is printed for each supergroup.

F11.6.14 ARRAY SUMMARY

The arrays that are used in the problem are summarized in the table shown in Fig. F11.6.16. This table is printed by subroutine PRTARA whenever more than one array is used in the problem.



```
*******************************************************
**                                                   **
**   array      units in   units in   units in   nesting **
**   number      x dir.     y dir.     z dir.     level  **
**                                                   **
**     1           1          2          2          2    **
**                                                   **
**     2           1          2          2          2    **
**                                                   **
**     3 global    2          1          1          1    **
**                                                   **
*******************************************************

 .......     0 io's were used loading the data     .......
```

Figure F11.6.16 Examples of array summary

The ARRAY NUMBER is the number by which the array is designated in the input data. The number of units in the x, y, and z directions is listed for each array. The NESTING LEVEL indicates the level of nesting for each array. The global, overall, or universe array is flagged by the word GLOBAL. The global array should always appear at the first nesting level. Arrays that have been placed in the global reflector by using holes should also appear at the first nesting level. A nesting level of one is the highest or first nesting level. The larger the number in the nesting level column, the deeper the nesting level.

F11.6.15 GEOMETRY DATA

The geometry region data utilized by the problem are printed by subroutine PRTJOM and cannot be suppressed. They should be carefully examined by the user to verify the mixture number, bias ID, and geometry specifications used in the problem. If geometry region data are input but are not referenced in the unit orientation array data, they will not be printed here. An example would be to input geometry region data describing Units 1, 2, 3, and 4 and to utilize only Units 1, 3, and 4 in the unit orientation array. Then the geometry region data for Unit 2 will not be printed. An example of the geometry region printout for a problem is given in Fig. F11.6.17.

The problem title and a heading are printed at the top of each page. REGION is the region number within a unit. Each unit has its regions numbered sequentially, beginning with one. MEDIA NUM is the mixture number or mixture ID that occupies the volume defined by the region. BIAS ID is the bias ID that corresponds to the desired set of weight average for biasing the region. The unit number is printed at the top of each unit's geometry region description, near the center of the page. The data printed for each geometry region include (1) the region number relative to the unit (numbered sequentially within the unit), (2) the shape of the geometry region, (3) the mixture ID of the material within the volume defined by the region, (4) the bias ID to define the average weight of a neutron in the region, and (5) the dimensions defining the outer boundaries of the geometry region. If additional geometry surrounds an array, a heading is printed stating: UNIT _____ EXTERNAL TO LATTICE _____. The lattice number is the number of the array that is surrounded by the specified geometry. The unit number is the unit or box type that contains the specified geometry. In the case of an external reflector for the global array, the unit number is assigned by the code.

F11.6.16 UNIT ORIENTATION DESCRIPTION

Each unit orientation description defines the location of units in the 3-D lattice that represents the specified array. The array that is described is identified in the heading: UNIT ORIENTATION DESCRIPTION FOR ARRAY _____. The arrays used in the problem are stacked together to represent the physical problem

```
                                       sample problem 18   1f27 demonstration of options problem
                      media bias      geometry description for those units utilized in this problem
region                 num   id
                                                   -----   unit     1   -----

  1 cylinder          1  1  radius =  9.5200      +z =  8.7804      -z = -8.7804      centerline is at  x = 0.00000e+00  y = 0.00000e+00
  2 cylinder          0  1  radius =  9.5200      +z =  8.9896      -z = -8.7804      centerline is at  x = 0.00000e+00  y = 0.00000e+00
  3 cylinder          2  1  radius =  10.160      +z =  9.6296      -z = -9.4204      centerline is at  x = 0.00000e+00  y = 0.00000e+00
  4 cuboid            4  1      +x =  18.450     -x = -18.450     +y =  18.450     -y = -18.450     +z =  17.895     -z = -17.685
                                          -----   unit     2  external to lattice  1   -----
  1 array number      1           +x =  73.800     -x = 0.00000e+00 +y =  73.800     -y = 0.00000e+00 +z =  71.160     -z = 0.00000e+00
                                          -----   unit     3  external to lattice  2   -----
  1 array number      2           +x =  73.800     -x = 0.00000e+00 +y =  73.800     -y = 0.00000e+00 +z =  35.580     -z = 0.00000e+00
                                          -----   unit     4  external to lattice  3   -----
  1 array number      3           +x =  36.900     -x = 0.00000e+00 +y =  73.800     -y = 0.00000e+00 +z =  106.74     -z = 0.00000e+00
                                          -----   unit     5  external to lattice  4   -----
  1 array number      4           +x =  110.70     -x = 0.00000e+00 +y =  36.900     -y = 0.00000e+00 +z =  106.74     -z = 0.00000e+00
                                          *******************   global   *******************
                                                   -----   unit     6   -----

  1 cuboid            4  1      +x =  55.350     -x = -55.350     +y =  55.350     -y = -55.350     +z =  53.370     -z = -53.370
    hole number       1        at x = -55.350      y = -18.450      z = -17.790     is unit number      2
    hole number       2        at x = -55.350      y = -18.450      z = -53.370     is unit number      3
    hole number       3        at x =  18.450      y = -18.450      z = -53.370     is unit number      4
    hole number       4        at x = -55.350      y = -55.350      z = -53.370     is unit number      5
  2 cuboid            3  2      +x =  58.350     -x = -58.350     +y =  58.350     -y = -58.350     +z =  56.370     -z = -56.370
  3 cuboid            3  3      +x =  61.350     -x = -61.350     +y =  61.350     -y = -61.350     +z =  59.370     -z = -59.370
  4 cuboid            3  4      +x =  64.350     -x = -64.350     +y =  64.350     -y = -64.350     +z =  62.370     -z = -62.370
  5 cuboid            3  5      +x =  67.350     -x = -67.350     +y =  67.350     -y = -67.350     +z =  65.370     -z = -65.370
  6 cuboid            3  6      +x =  70.350     -x = -70.350     +y =  70.350     -y = -70.350     +z =  68.370     -z = -68.370
  7 cuboid            3  7      +x =  70.590     -x = -70.590     +y =  70.590     -y = -70.590     +z =  68.610     -z = -68.610
```

Figure F11.6.17 Example of geometry region data

being analyzed. The unit orientation description is not printed if only Unit 1 is described in the problem. The user should carefully examine the unit orientation descriptions to ensure proper placement of the units in each lattice. A sample unit orientation description is shown in Fig. F11.6.18.

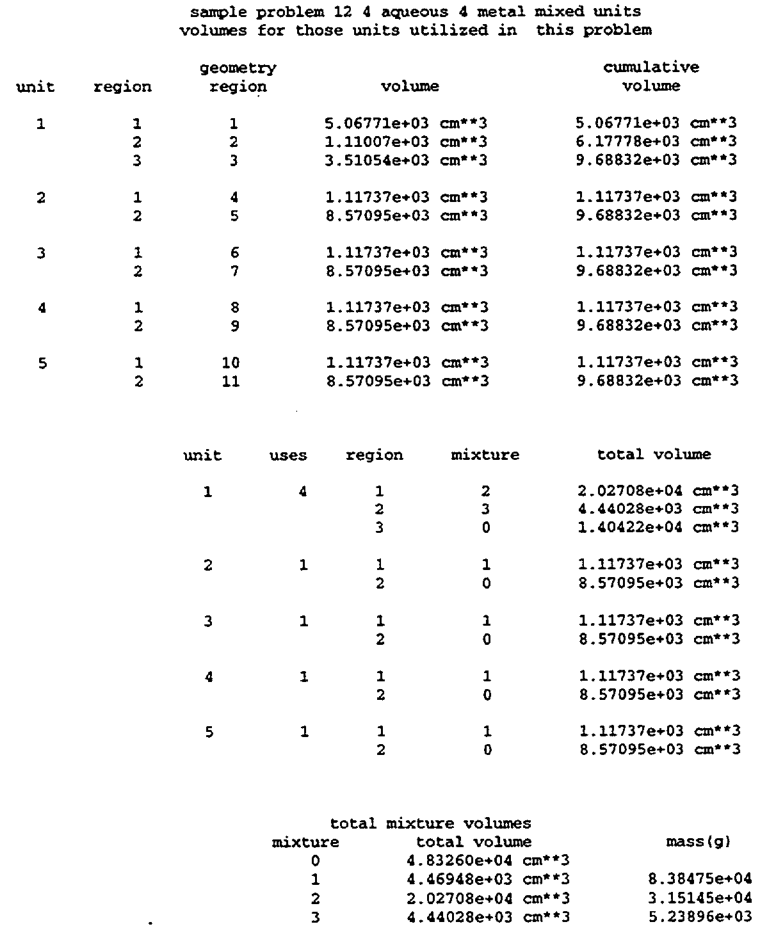
```
                              sample problem 2   2c8 bare with 8 unit types matrix calculation

                      -------   unit orientation description for array   1          -------

 z layer   1, x column   1 to   2 left to right   y row   1 to   2  bottom to top

  3 4

  1 2


 z layer   2, x column   1 to   2 left to right   y row   1 to   2  bottom to top

  7 8

  5 6
```

Figure F11.6.18 Example of unit orientation description

If a very large array is utilized by the problem, its unit orientation description may be spread over several pages. When checking the printout, the user should pay careful attention to the headings that indicate the portion of each lattice being printed. The unit orientation descriptions are printed in subroutine PRTLBA.

F11.6.17 VOLUME INFORMATION

Three tables of volumes are printed by subroutine VOLUME and cannot be suppressed. The problem title is printed at the top of the page, followed by the heading "VOLUMES FOR THOSE UNITS UTILIZED IN THIS PROBLEM." An example of the volume printout is given in Fig. F11.6.19.



```
                         sample problem 12 4 aqueous 4 metal mixed units
                        volumes for those units utilized in  this problem

                      geometry                                  cumulative
 unit     region       region          volume                     volume

  1         1            1        5.06771e+03 cm**3         5.06771e+03 cm**3
            2            2        1.11007e+03 cm**3         6.17778e+03 cm**3
            3            3        3.51054e+03 cm**3         9.68832e+03 cm**3

  2         1            4        1.11737e+03 cm**3         1.11737e+03 cm**3
            2            5        8.57095e+03 cm**3         9.68832e+03 cm**3

  3         1            6        1.11737e+03 cm**3         1.11737e+03 cm**3
            2            7        8.57095e+03 cm**3         9.68832e+03 cm**3

  4         1            8        1.11737e+03 cm**3         1.11737e+03 cm**3
            2            9        8.57095e+03 cm**3         9.68832e+03 cm**3

  5         1           10        1.11737e+03 cm**3         1.11737e+03 cm**3
            2           11        8.57095e+03 cm**3         9.68832e+03 cm**3


                      unit     uses    region     mixture        total volume

                       1        4        1           2        2.02708e+04 cm**3
                                         2           3        4.44028e+03 cm**3
                                         3           0        1.40422e+04 cm**3

                       2        1        1           1        1.11737e+03 cm**3
                                         2           0        8.57095e+03 cm**3

                       3        1        1           1        1.11737e+03 cm**3
                                         2           0        8.57095e+03 cm**3

                       4        1        1           1        1.11737e+03 cm**3
                                         2           0        8.57095e+03 cm**3

                       5        1        1           1        1.11737e+03 cm**3
                                         2           0        8.57095e+03 cm**3


                                  total mixture volumes
                              mixture      total volume                mass(g)
                                 0       4.83260e+04 cm**3
                                 1       4.46948e+03 cm**3           8.38475e+04
                                 2       2.02708e+04 cm**3           3.15145e+04
                                 3       4.44028e+03 cm**3           5.23896e+03
```

Figure F11.6.19 Sample volume information

The first table is arranged by ascending unit number. It includes (1) the unit number, (2) the region number within the unit, (3) the overall geometry region number, (4) the net volume of each individual region, and (5) the cumulative volume through each region in the unit. The cumulative volume of the last region in a unit is the total volume of the unit. The unit number is printed under the heading UNIT. Data listed under the heading REGION refers to the number of the geometry region within the unit. The geometry regions within a unit are numbered sequentially starting with 1. Data entered under the heading GEOMETRY REGION refers to the entry number of the individual geometry region. These are numbered sequentially, starting with 1, through the TOTAL INPUT GEOMETRY REGIONS defined in Sect. F11.6.12. The net volume of each individual region is calculated by subtracting the volume of the interior region from the volume of the region and is listed under the heading VOLUME. The data listed under the heading CUMULATIVE VOLUME is calculated from the dimensions of the region. A simple example demonstrating how volumes are calculated can be given by assuming a unit that is composed of three concentric cubes. Region 1 is a cube 3 cm on a side, region 2 is a cube 4 cm on a side, and region 3 is a cube 5 cm on a side. The cumulative volume of region 1 is 27 cm^3 (3^3); the cumulative volume of region 2 is 64 cm^3 (4^3); the cumulative volume of region 3 is 125 cm^3 (5^3). The volume of region 1 is 27 cm^3 (3^3), the volume of region 2 is 37 cm^3 (64 - 27), and the volume of region 3 is 61 cm^3 (125 - 64).

The second table contains (1) the unit number, (2) the number of times the unit is used in the problem, (3) the region number within the unit, (4) the mixture number present in the region, and (5) the total volume associated with the region in the whole problem. The unit number is printed under the heading UNIT, data printed under the heading USES indicates the number of times the unit is used in the problem. Data printed under the heading MIXTURE indicates the mixture number used in the region. The total volume of each region is printed under the heading TOTAL VOLUME and is determined by multiplying the VOLUME of the region listed in the first table by the number of times the unit containing that region is used in the problem.

The third table is printed following the heading "TOTAL MIXTURE VOLUMES." In this table, the mixtures used in the problem are listed with their associated total volume and total mass. The mixture numbers are printed under the heading MIXTURE, the total volume of each mixture is printed under the heading TOTAL VOLUME, and the mass of each mixture is printed under the heading MASS(G). All masses will be printed as zero if the working-format cross-section library does not contain the data required to calculate the densities of the mixtures used in the problem.

F11.6.18 BIASING INFORMATION

This table specifies the weighting or biasing data to be used in the problem. An example of biasing information is given in Fig. F11.6.20.

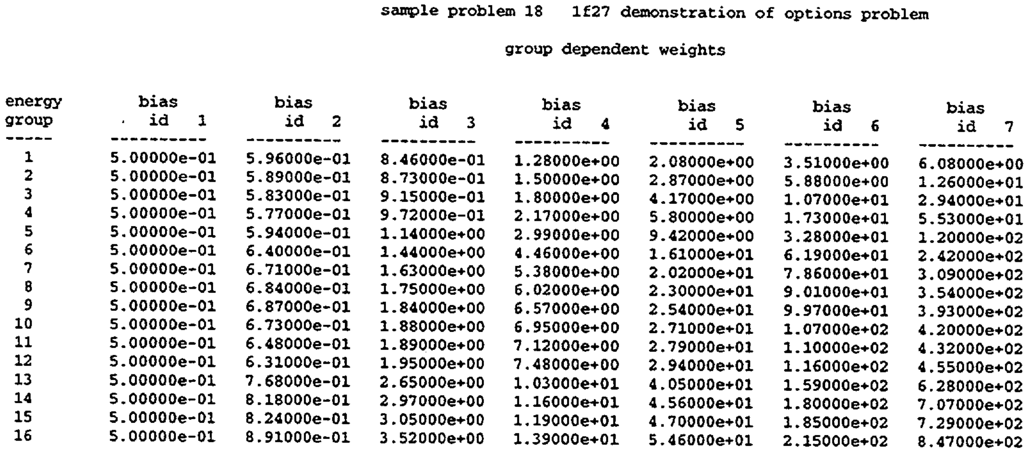
```
********************************************************************************************************************
***                                                                                                              ***
***                                         biasing information                                                  ***
***                                                                                                              ***
***     weighting intervals   1 to   6 for paraffin    ,mat id=  400 will be used for bias id's    2 to    7     ***
***                                                                                                              ***
***     a default weight of     .500 will be used for all other bias id's.                                       ***
***                                                                                                              ***
********************************************************************************************************************
```

Figure F11.6.20 Biasing information

The user is responsible for determining from the input data whether the group-dependent weights (*wtavg*) for the specified material(s) were obtained from the weighting library or were entered by the user. The group-dependent weights can be printed for verification purposes as shown in Sect. F11.6.19.

F11.6.19 GROUP-DEPENDENT WEIGHTS

Printing the group-dependent weights is optional. They are printed by subroutine PRTWTS if PWT=YES (Sect. F11.4.3) is entered in the parameter data. An example of the printed group-dependent weights is shown in Fig. F11.6.21.



```
                                   sample problem 18   1f27 demonstration of options problem

                                                 group dependent weights

energy      bias          bias          bias          bias          bias          bias          bias
group     . id   1        id   2        id   3        id   4        id   5        id   6        id   7
-----    ----------    ----------    ----------    ----------    ----------    ----------    ----------
  1      5.00000e-01   5.96000e-01   8.46000e-01   1.28000e+00   2.08000e+00   3.51000e+00   6.08000e+00
  2      5.00000e-01   5.89000e-01   8.73000e-01   1.50000e+00   2.87000e+00   5.88000e+00   1.26000e+01
  3      5.00000e-01   5.83000e-01   9.15000e-01   1.80000e+00   4.17000e+00   1.07000e+01   2.94000e+01
  4      5.00000e-01   5.77000e-01   9.72000e-01   2.17000e+00   5.80000e+00   1.73000e+01   5.53000e+01
  5      5.00000e-01   5.94000e-01   1.14000e+00   2.99000e+00   9.42000e+00   3.28000e+01   1.20000e+02
  6      5.00000e-01   6.40000e-01   1.44000e+00   4.46000e+00   1.61000e+01   6.19000e+01   2.42000e+02
  7      5.00000e-01   6.71000e-01   1.63000e+00   5.38000e+00   2.02000e+01   7.86000e+01   3.09000e+02
  8      5.00000e-01   6.84000e-01   1.75000e+00   6.02000e+00   2.30000e+01   9.01000e+01   3.54000e+02
  9      5.00000e-01   6.87000e-01   1.84000e+00   6.57000e+00   2.54000e+01   9.97000e+01   3.93000e+02
 10      5.00000e-01   6.73000e-01   1.88000e+00   6.95000e+00   2.71000e+01   1.07000e+02   4.20000e+02
 11      5.00000e-01   6.48000e-01   1.89000e+00   7.12000e+00   2.79000e+01   1.10000e+02   4.32000e+02
 12      5.00000e-01   6.31000e-01   1.95000e+00   7.48000e+00   2.94000e+01   1.16000e+02   4.55000e+02
 13      5.00000e-01   7.68000e-01   2.65000e+00   1.03000e+01   4.05000e+01   1.59000e+02   6.28000e+02
 14      5.00000e-01   8.18000e-01   2.97000e+00   1.16000e+01   4.56000e+01   1.80000e+02   7.07000e+02
 15      5.00000e-01   8.24000e-01   3.05000e+00   1.19000e+01   4.70000e+01   1.85000e+02   7.29000e+02
 16      5.00000e-01   8.91000e-01   3.52000e+00   1.39000e+01   5.46000e+01   2.15000e+02   8.47000e+02
```

Figure F11.6.21 Example of biasing data

The title is printed at the top of the table. The average weight (*wtavg*) is printed for each energy group and each BIAS ID. The BIAS ID number printed at the top of the column corresponds to the BIAS ID used in the geometry region description and printed in the biasing information.

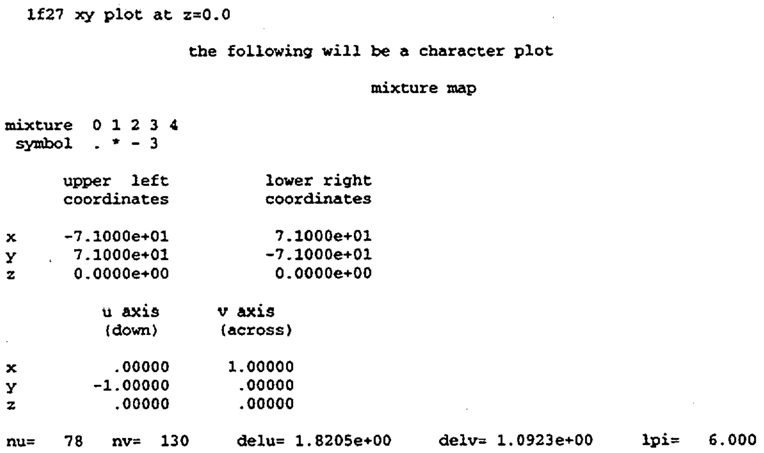
F11.6.20 PLOT REPRESENTATION

Plots representing 2-D slices through the geometrical description of the problem are optional. They are created if plot data are entered as specified in Sect. F11.4.11 unless PLT=NO is specified either in the plot data or the parameter data (Sect. F11.4.3). A plot or plots can be generated and displayed as (1) alphanumeric characters to represent mixture numbers, unit numbers, or bias ID numbers or (2) color plots displayed to the screen or plotting device using colors to represent mixture numbers, unit numbers, or bias ID numbers. Color plots generate a GIF file and require an independent program to be displayed using a plotting device.

An example of the output generated using the character plot method is given in Figs. F11.6.22 and F11.6.23. An example of the output generated using the color plot method is given in Figs. F11.6.24 and F11.6.25.

Figure F11.6.22 summarizes the data used to generate the character plot. Figure F11.6.23 is an example of a character plot of the 2-D slice specified through the geometrical description of the problem.

In Figure F11.6.22, the plot title is printed at the top of the page, followed by a statement that "THE FOLLOWING WILL BE A CHARACTER PLOT." If a plot title was not entered in the plot data, the plot title is defaulted to the problem title. The title is followed by a heading specifying the type of plot (MIXTURE MAP,



```
 1f27 xy plot at z=0.0

                    the following will be a character plot

                                         mixture map

mixture  0 1 2 3 4
 symbol  . * - 3

      upper  left           lower right
      coordinates           coordinates

x     -7.1000e+01            7.1000e+01
y      7.1000e+01           -7.1000e+01
z      0.0000e+00            0.0000e+00

          u axis      v axis
          (down)      (across)

x           .00000     1.00000
y         -1.00000      .00000
z           .00000      .00000

nu=   78   nv=  130    delu= 1.8205e+00     delv= 1.0923e+00     lpi=   6.000
```

Figure F11.6.22 Summary of character plot symbols, coordinates, and data

BIAS ID MAP, or UNIT MAP). This is followed by a table that correlates the symbols to be used in the character plot with the mixture numbers, bias ID numbers, or unit numbers that were used in the problem. If the problem is a bare array, the overall system coordinates are printed. Then the coordinates of the upper-left corner and lower-right corner of the plot are printed. This is followed by the direction cosines down and across the plot. The remaining plot parameters (including both input data and calculated values) are then printed. NU is the number of characters printed in the "U" (down) direction, NV is the number of characters printed in the "V" (across) direction, DELU is the incremental distance, in cm, represented by each character in the "U" (down) direction, DELV is the incremental distance, in cm, represented by each character in the "V" (across) direction, and LPI is the vertical to horizontal scaling factor for plot proportionality.

Figure F11.6.23 shows a character plot of a 2-D slice specified through the geometrical description of the problem. These plots aid the user in verifying that the problem is described correctly. Any number of plots can be made in one problem.

Figure F11.6.24 summarizes the data used to generate the color plot. Figure F11.6.25 is an example of a color plot of the 2-D slice specified through the geometrical description of the problem. This plot does not appear in the KENO V.a printout. It is generated from a GIF file that is created when a color plot has been specified in the KENO V.a input data and requires special processing by the user.

In Figure F11.6.24, the plot title is printed at the top of the page, followed by a statement that "THE FOLLOWING WILL BE A COLOR PLOT." If a plot title was not entered in the plot data, the plot title is defaulted to the problem title. The title is followed by a heading specifying the type of plot (MIXTURE MAP, BIAD ID MAP, or UNIT MAP). If the problem is a bare array, the overall system coordinates are printed. Then the coordinates of the upper-left corner and lower-right corner of the plot are printed. This is followed by the direction cosines down and across the plot. The remaining plot parameters (including both input data and calculated values) are then printed. NU is the number of characters printed in the "U" (down) direction, NV is the number of characters printed in the "V" (across) direction, DELU is the incremental distance, in cm, represented by each character in the "U" (down) direction, DELV is the incremental distance, in cm, represented by each character in the "V" (across) direction, and LPI is the vertical to horizontal scaling factor for plot proportionality.

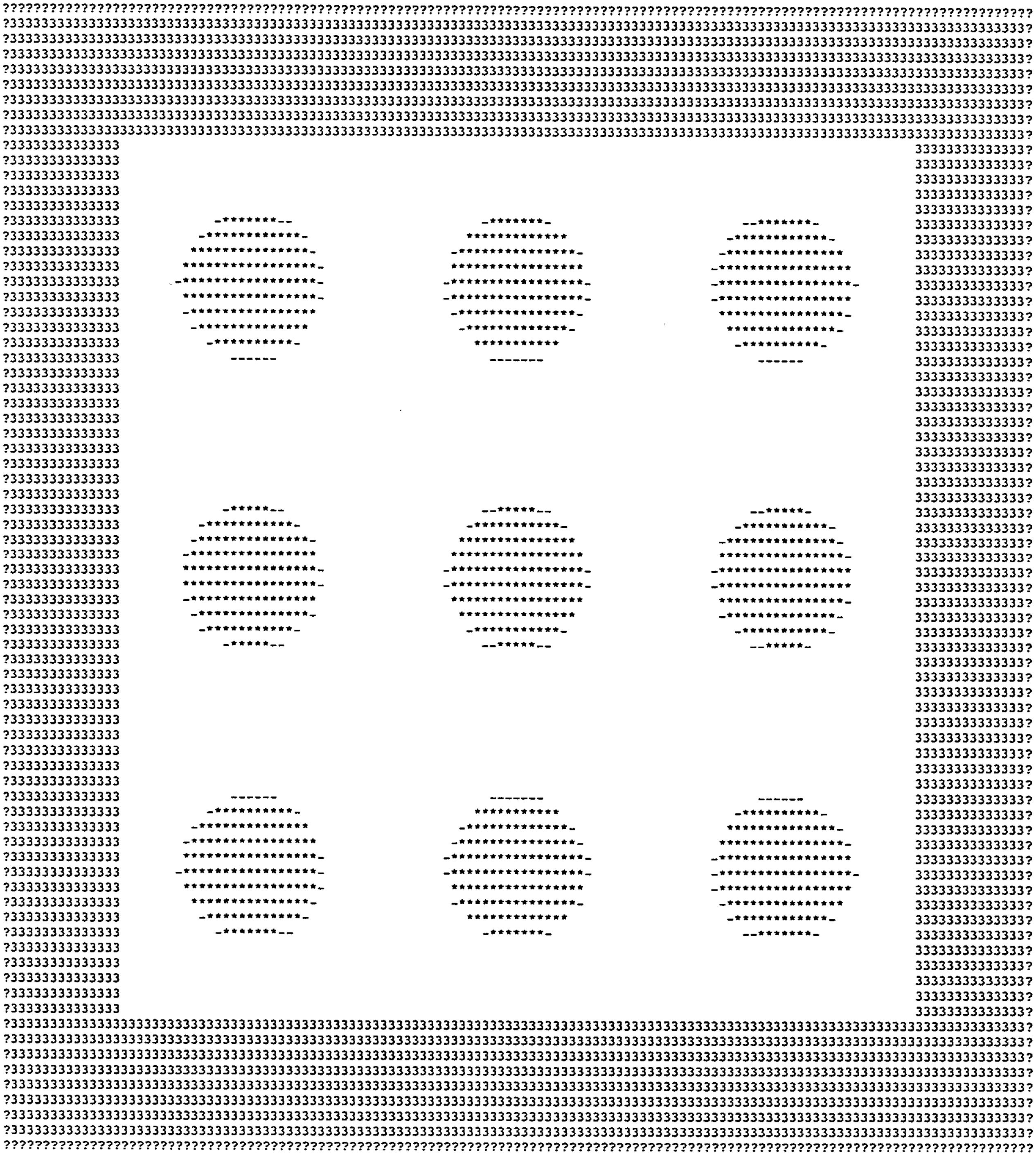


Figure F11.6.23 Sample character plot representation

```
  1f27 xy plot at z=0.0

                   the following will be a color plot

                                  mixture map

mixture  0 1 2 3 4
 symbol    1 2 3 4

      upper  left          lower right
      coordinates          coordinates

x     -7.1000e+01           7.1000e+01
y      7.1000e+01          -7.1000e+01
z      0.0000e+00           0.0000e+00

          u axis       v axis
          (down)      (across)

x          .00000      1.00000
y        -1.00000       .00000
z          .00000       .00000

nu= 640   nv= 640    delu= 2.2187e-01    delv= 2.2187e-01    lpi=  10.000
```

Figure F11.6.24 Summary of CRT plot symbols, coordinates, and data

Figure F11.6.25 shows an example of a color plot of a 2-D slice specified through the geometrical description of the problem. The color plots do not appear in the KENO V.a printout. They are generated from GIF files that are created when a color plot has been specified in the KENO V.a input data. Color plots require special processing by the user. Any number of plots can be made in one problem. The color plots can be a valuable tool to assist the user in verifying that a problem is described correctly.

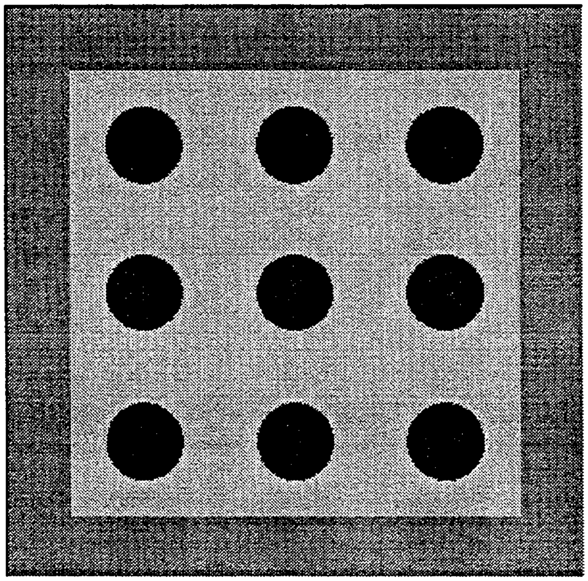


Figure F11.6.25 Sample color plot representation

F11.6.21 INITIAL SOURCE AND FINAL PRETRACKING EDITS

Prior to calculating the k-effectives for each generation, KENO V.a prints the number of I/Os used before tracking and the number of minutes used processing the data. If the problem is not a restart problem, initial source information follows those two lines as shown in Figs. F11.6.26 through F11.6.33.

The volume fraction of fissile material in the core is the first line of data printed. If RFL=YES (Sect. F11.4.8) was specified in the start data, the volume fraction message is changed to: VOLUME FRACTION OF FISSILE MATERIAL IN THE SYSTEM =. This is followed by the start type, other data utilized to generate the initial source distribution, and finally, the amount of time used to generate the initial source distribution and the total elapsed time. Figures F11.6.26 through F11.6.33 illustrate starting information printed for start type 0 through start type 6. The cuboid for choosing starting points is (1) the core boundary for either a bare array or a reflected array with the key to start in the reflector turned off (RFL=NO), (2) the overall system for a reflected array if the key to start in the reflector is turned on (RFL=YES) and the outer region is a cuboid, or (3) a cuboid specified by the user via the keyword XSM=, XSP=, YSM=, YSP=, ZSM=, and ZSP=. For each start type for which the reflector key, RFL= is applicable, a statement is printed stating that "the flag to start neutrons in the reflector is turned 'on' or 'off.'

Figure F11.6.26 illustrates typical starting data for start type 0 (a flat start in fissile material). In this example, the starting cuboid is the core boundary of a bare array. The default value of RFL was not overridden. The parameter used in this example was NST=0.

```
                                 ........    0 io's were used in keno-v before tracking       ........

                                 ........   .02133 minutes were used processing data.        ........

 volume fraction of fissile material in the core= 4.54932e-01

 start type 0 was used.

 the neutrons were started with a flat distribution in a cuboid defined by:
                          +x= 2.74800e+01  -x= 0.00000e+00  +y= 2.74800e+01  -y= 0.00000e+00  +z= 2.60200e+01  -z= 0.00000e+00
 the flag to start neutrons in the reflector was turned off

  .00000 minutes were required for starting.   total elapsed time is  .02133 minutes.
```

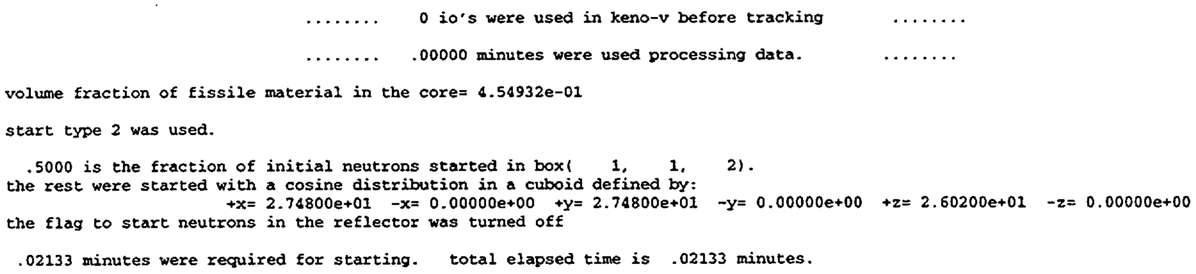
Figure F11.6.26 Example of initial source information for start type 0

Figure F11.6.27 illustrates typical starting data for start type 1 (a cosine distribution throughout a cuboid). In this example the starting cuboid is the core boundary of a bare array. The default value of RFL was not overridden. The parameter used in this example was NST=1.

```
                                 ........    0 io's were used in keno-v before tracking       ........

                                 ........   .02133 minutes were used processing data.        ........

 volume fraction of fissile material in the core= 4.54932e-01

 start type 1 was used.

 the neutrons were started with a cosine distribution in a cuboid defined by:
                          +x= 2.74800e+01  -x= 0.00000e+00  +y= 2.74800e+01  -y= 0.00000e+00  +z= 2.60200e+01  -z= 0.00000e+00
 the flag to start neutrons in the reflector was turned off

  .00000 minutes were required for starting.   total elapsed time is  .02133 minutes.
```

Figure F11.6.27 Example of initial source information for start type 1

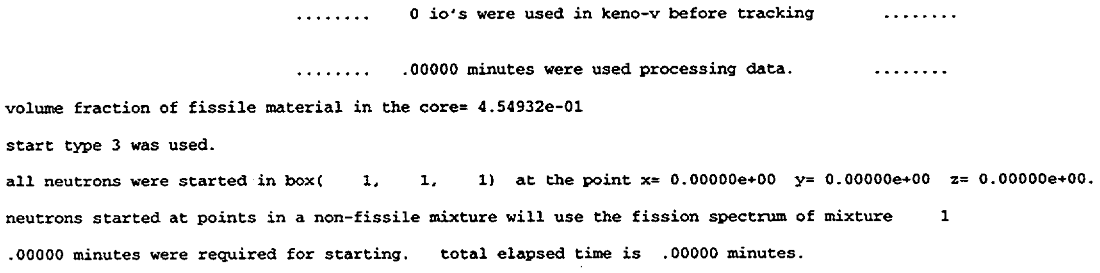
Figure F11.6.28 illustrates typical data for start type 2 (NST=2). In this example, the starting cuboid is the core boundary of the bare array. FCT=0.5 specifies that half of the initial neutrons will be started in a spike in the unit located in position (1,1,2) in the global array. The position was set by specifying NXS=1, NYS=1, and NZS=2. The default value of the reflector key was not overridden.



```
                                ........    0 io's were used in keno-v before tracking        ........

                                ........   .00000 minutes were used processing data.        ........

volume fraction of fissile material in the core= 4.54932e-01

start type 2 was used.

  .5000 is the fraction of initial neutrons started in box(    1,    1,    2).
the rest were started with a cosine distribution in a cuboid defined by:
                         +x= 2.74800e+01  -x= 0.00000e+00  +y= 2.74800e+01  -y= 0.00000e+00  +z= 2.60200e+01  -z= 0.00000e+00
the flag to start neutrons in the reflector was turned off

 .02133 minutes were required for starting.   total elapsed time is  .02133 minutes.
```

Figure F11.6.28 Example of initial source information for start type 2

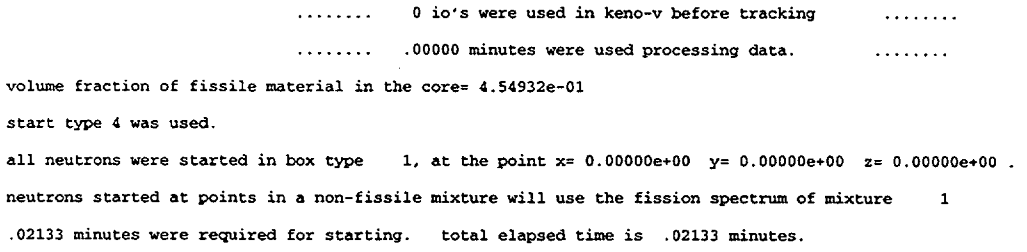
Figure F11.6.29 illustrates data for start type 3 (NST=3). All neutrons are started at the origin of the unit located at position (1,1,1) in the global array. These data were set by specifying the position in the global array using NXS=1 NYS=1 NZS=1. The starting point was set by using TFX=0 TFY=0 TFZ=0. KFS=1 causes neutrons started in a nonfissile mixture to use the fission spectrum associated with mixture 1.



```
                               ........    0 io's were used in keno-v before tracking        ........

                               ........   .00000 minutes were used processing data.        ........

volume fraction of fissile material in the core= 4.54932e-01

start type 3 was used.

all neutrons were started in box(    1,    1,    1)  at the point x= 0.00000e+00  y= 0.00000e+00  z= 0.00000e+00.

neutrons started at points in a non-fissile mixture will use the fission spectrum of mixture      1

.00000 minutes were required for starting.   total elapsed time is  .00000 minutes.
```

Figure F11.6.29 Example of initial source information for start type 3

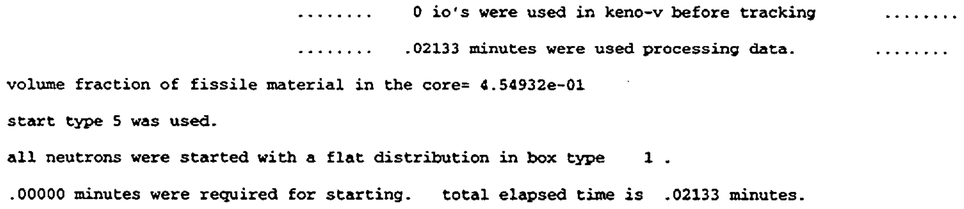
Figure F11.6.30 illustrates data for start type 4 (NST=4). All neutrons are started at the origin of unit 1. The parameters that set these data were NBX=1 TFX=0 TFY=0 TFZ=0 KFS=1.



```
                                   ........    0 io's were used in keno-v before tracking       ........
                                   ........   .00000 minutes were used processing data.        ........
volume fraction of fissile material in the core= 4.54932e-01
start type 4 was used.
all neutrons were started in box type    1, at the point x= 0.00000e+00  y= 0.00000e+00  z= 0.00000e+00 .
neutrons started at points in a non-fissile mixture will use the fission spectrum of mixture     1
.02133 minutes were required for starting.   total elapsed time is  .02133 minutes.
```

Figure F11.6.30 Example of initial source information for start type 4

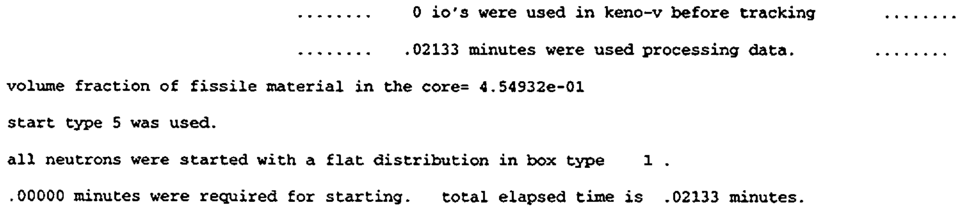
Figure F11.6.31 illustrates data for start type 5 (NST=5). The parameter used in this example was NBX=1.



```
                                   ........    0 io's were used in keno-v before tracking       ........
                                   ........   .02133 minutes were used processing data.        ........
volume fraction of fissile material in the core= 4.54932e-01
start type 5 was used.
all neutrons were started with a flat distribution in box type    1 .
.00000 minutes were required for starting.   total elapsed time is  .02133 minutes.
```

Figure F11.6.31 Example of initial source information for start type 5

Figure F11.6.32 illustrates start type 6 information. The parameters in this example, NST=6. TFX=13.74 TFY=13.74 TFZ=13.01 LNU=150 were used to start the first 150 neutrons at the point (13.74, 13.74, 13.01). TFX=0 TFY=0 TFZ=0 LNU=300 starts neurons 151 through 300 at the point (0,0,0). KFS=1 will cause the fission spectrum for mixture 1 to be used for any starting points that are not in fissile material.



```
                                   ........    0 io's were used in keno-v before tracking       ........
                                   ........   .02133 minutes were used processing data.        ........
volume fraction of fissile material in the core= 4.54932e-01
start type 5 was used.
all neutrons were started with a flat distribution in box type    1 .
.00000 minutes were required for starting.   total elapsed time is  .02133 minutes.
```

Figure F11.6.32 Example of initial source information for start type 6

Figure F11.6.33 illustrates the use of binary start type 6 data. The parameters entered for this example are NST=6, KFS=1, RDU=52.

```
                 ........    0 io's were used in keno-v before tracking       ........

                 ........   .02133 minutes were used processing data.        ........

volume fraction of fissile material in the core= 4.54932e-01

start type 6 was used.

neutrons were started from binary start data on logical unit number  52

neutrons started in non-fissile mixtures will use the fission spectrum for mixture     1

 .00000 minutes were required for starting.   total elapsed time is  .02133 minutes.
```

Figure F11.6.33 Example of initial source information for binary start type 6

Once the initial source distribution has been prepared, the time required for preparing them (starting) is printed, as well as the total cumulative time used by the code.

F11.6.22 PRINT STARTING POINTS

This printout is optional and is used to verify the initial source starting points. This option is activate by specifying the parameter PSP=YES. An example of this information is given in Fig. F11.6.34.

The information pertinent to the initial source distribution is printed two lines at a time and appears under the designated headings. For example, the coordinates X, Y, and Z are printed on one line and the direction cosines U, V, and W are printed directly under them. The data printed for each source neutron include the following:

NEUTRON is the ID number of the neutron.

X, Y, and Z are the coordinates of the starting point relative to the coordinate system of the unit.

K is the region number that contains the point X, Y, Z.

NOW is the array number.

NBX, NBY, NBZ are the coordinates of the unit within the array.

LL is the unit or box type number located at NBX, NBY, NBZ.

KR is the mixture number present at the starting point.

KI is the bias ID number or importance region at the starting point.

WT is the current weight (WT is always 1.0 for a neutron when it is started).

U, V, and W are the direction cosines defining the direction the history is traveling.

```
starting points will be printed using the debug tracking format because start parameter psp= was entered as yes.
 neutron is the id number of the neutron.
 x, y and z are the coordinates of the starting point.
 k is the region number.
 now is the array number.
 nbx, nby and nbz define a location in the now array.
 ll is the unit number at location nbx, nby, nbz.
 kr is the media or mixture number in region k.
 ki is the importance region.
 wt is the starting weight.
 u, v and w are the direction cosines defining the direction.
 ig is the energy group.
 kcor is the region number surrounding the now array.
 kl is the region number of the first region in unit ll.
 k2 is the region number of the last region in unit ll.
 igeo defines the geometry of region k.
 kcol is the region number of the last collision.
 nsig is the position of group ig in its supergroup.
 sg is the supergroup number for group ig.
 random number is the current random number.


 where     neutron             x              y              z         k  now  nbx  nby  nbz   ll   kr   ki        wt
                               u              v              w        ig kcor   kl   k2 igeo kcol nsig   sg   random number

 start          1     -.38147e-05     .00000e+00     .10460e+00    1    1    2    1    1    1    1    1      .10000e+01
                       .68080e+00    -.57359e+00     .45553e+00    1    5    1    4    2    0    1    1     1c2c35e3321d

 start          2     -.38147e-05     .00000e+00     .10460e+00    1    1    2    1    1    1    1    1      .10000e+01
                       .84683e+00    -.53002e+00     .44281e-01    4    5    1    4    2    0    2    2     1c2c35e3321d

 start          3     -.38147e-05     .00000e+00     .10460e+00    1    1    2    1    1    1    1    1      .10000e+01
                      -.62644e+00    -.60236e+00     .49470e+00    1    5    1    4    2    0    1    1     1c2c35e3321d

 start          4     -.38147e-05     .00000e+00     .10460e+00    1    1    2    1    1    1    1    1      .10000e+01
                      -.59019e+00    -.78390e-01     .80345e+00    2    5    1    4    2    0    2    1     1c2c35e3321d

 start          5     -.38147e-05     .00000e+00     .10460e+00    1    1    2    1    1    1    1    1      .10000e+01
                       .58847e+00     .59437e+00     .54811e+00    1    5    1    4    2    0    1    1     1c2c35e3321d

 start          6     -.38147e-05     .00000e+00     .10460e+00    1    1    2    1    1    1    1    1      .10000e+01
                      -.70760e+00     .63163e+00     .31677e+00    2    5    1    4    2    0    2    1     1c2c35e3321d

 start          7     -.38147e-05     .00000e+00     .10460e+00    1    1    2    1    1    1    1    1      .10000e+01
                      -.29209e+00    -.89329e+00    -.34164e+00    2    5    1    4    2    0    2    1     1c2c35e3321d

 start          8     -.38147e-05     .00000e+00     .10460e+00    1    1    2    1    1    1    1    1      .10000e+01
                       .63931e+00    -.16028e+00    -.75206e+00    3    5    1    4    2    0    1    2     1c2c35e3321d

 start          9     -.38147e-05     .00000e+00     .10460e+00    1    1    2    1    1    1    1    1      .10000e+01
                       .88299e+00     .40807e+00    -.23196e+00    5    5    1    4    2    0    3    2     1c2c35e3321d

 start         10     -.38147e-05     .00000e+00     .10460e+00    1    1    2    1    1    1    1    1      .10000e+01
                      -.72451e+00     .66305e+00    -.18827e+00    5    5    1    4    2    0    3    2     1c2c35e3321d
```

Figure F11.6.34 Example of initial source points

IG is the energy group.

KCOR is the core boundary region of the array where the neutron is located.

K1 is the region number of the first region in unit LL.

K2 is the region number of the last region in unit LL.

IGEO is an integer that defines the geometry shape of the region.

KCOL is the region number in which the last collision occurred.

NSIG is the position of the group IG in the supergroup.

SG is the supergroup number.

RANDOM NUMBER is the current random number.

When starting points are printed, many of the above named variables have not been initialized. For starting, the variables of interest are X, Y, Z, U, V, W, NBX, NBY, NBZ, and LL.

F11.6.23 K-EFFECTIVES BY GENERATION

At the completion of each generation, KENO V.a prints the k-effective for that generation and associated information. An example of this printout is given in Fig. F11.6.35.

```
 sample problem 18   1f27 demonstration of options problem

                generation    elapsed time      average        avg k-eff        matrix      matrix k-eff
    generation  k-effective     minutes       k-effective      deviation     k-effective     deviation
         1      1.32662e+00    8.53333e-02    1.00000e+00    0.00000e+00    1.32662e+00    0.00000e+00
         2      1.08427e+00    1.28000e-01    1.00000e+00    0.00000e+00    1.20544e+00    1.21178e-01
         3      1.08000e+00    1.49333e-01    1.08000e+00    0.00000e+00    1.16363e+00    8.15055e-02
         4      1.07653e+00    1.70667e-01    1.07827e+00    1.73360e-03    1.14185e+00    6.16091e-02
         5      1.11658e+00    1.92000e-01    1.09104e+00    1.28093e-02    1.13680e+00    4.79893e-02
         6      1.03783e+00    2.34667e-01    1.07773e+00    1.60936e-02    1.12030e+00    4.25140e-02
         7      1.04516e+00    2.56000e-01    1.07122e+00    1.40659e-02    1.10957e+00    3.75001e-02
         8      9.21061e-01    2.77333e-01    1.04619e+00    2.75357e-02    1.08600e+00    4.01238e-02
         9      1.01138e+00    2.98667e-01    1.04122e+00    2.37973e-02    1.07771e+00    3.63443e-02
        10      1.00784e+00    3.20000e-01    1.03705e+00    2.10273e-02    1.07072e+00    3.32500e-02
        11      9.95260e-01    3.62667e-01    1.03240e+00    1.91168e-02    1.06386e+00    3.08484e-02
        12      1.01399e+00    3.84000e-01    1.03056e+00    1.71975e-02    1.05971e+00    2.84659e-02
        13      1.05725e+00    4.05333e-01    1.03299e+00    1.57438e-02    1.05952e+00    2.61856e-02
        14      9.84478e-01    4.26667e-01    1.02895e+00    1.49298e-02    1.05416e+00    2.48288e-02
        15      1.04069e+00    4.48000e-01    1.02985e+00    1.37631e-02    1.05326e+00    2.31316e-02
        16      9.90864e-01    4.90667e-01    1.02706e+00    1.30429e-02    1.04936e+00    2.19864e-02
        17      9.98263e-01    5.12000e-01    1.02514e+00    1.22931e-02    1.04635e+00    2.08703e-02
        18      1.01268e+00    5.33333e-01    1.02437e+00    1.15255e-02    1.04448e+00    1.97656e-02
        19      1.02362e+00    5.76000e-01    1.02432e+00    1.08264e-02    1.04338e+00    1.87287e-02
        20      9.71344e-01    5.97333e-01    1.02138e+00    1.06231e-02    1.03978e+00    1.81291e-02
        21      1.00861e+00    6.18667e-01    1.02071e+00    1.00709e-02    1.03830e+00    1.73079e-02
        22      9.68697e-01    6.40000e-01    1.01811e+00    9.90171e-03    1.03513e+00    1.68030e-02
        23      9.92491e-01    6.82667e-01    1.01689e+00    9.49706e-03    1.03328e+00    1.61626e-02
        24      1.05154e+00    7.04000e-01    1.01846e+00    9.19110e-03    1.03404e+00    1.54933e-02
        25      1.04795e+00    7.46667e-01    1.01974e+00    8.87551e-03    1.03460e+00    1.48710e-02
        26      1.02083e+00    7.68000e-01    1.01979e+00    8.49777e-03    1.03407e+00    1.42975e-02
        27      9.10628e-01    7.89333e-01    1.01542e+00    9.24667e-03    1.02950e+00    1.44977e-02
        28      1.01445e+00    8.10667e-01    1.01538e+00    8.88399e-03    1.02896e+00    1.39807e-02
        29      9.12893e-01    8.53333e-01    1.01159e+00    9.35353e-03    1.02496e+00    1.40711e-02
        30      9.47510e-01    8.74667e-01    1.00930e+00    9.29928e-03    1.02237e+00    1.38370e-02
        31      9.13629e-01    8.96000e-01    1.00600e+00    9.56014e-03    1.01887e+00    1.38353e-02
        32      1.03127e+00    9.17333e-01    1.00684e+00    9.27430e-03    1.01925e+00    1.34016e-02
        33      1.02911e+00    9.38667e-01    1.00756e+00    8.99886e-03    1.01955e+00    1.29927e-02
        34      9.06475e-01    9.81333e-01    1.00440e+00    9.26809e-03    1.01623e+00    1.30361e-02
        35      9.70098e-01    1.00267e+00    1.00336e+00    9.04280e-03    1.01491e+00    1.27265e-02
        36      9.87617e-01    1.02400e+00    1.00290e+00    8.78502e-03    1.01415e+00    1.23912e-02
        37      9.40109e-01    1.04533e+00    1.00111e+00    8.71694e-03    1.01215e+00    1.22166e-02
        38      1.01317e+00    1.08800e+00    1.00144e+00    8.47797e-03    1.01218e+00    1.18909e-02
        39      1.01746e+00    1.10933e+00    1.00187e+00    8.25701e-03    1.01231e+00    1.15828e-02
        40      9.84114e-01    1.13067e+00    1.00141e+00    8.05037e-03    1.01161e+00    1.13115e-02
        41      1.00827e+00    1.15200e+00    1.00158e+00    7.84320e-03    1.01153e+00    1.10326e-02
        42      1.02063e+00    1.19467e+00    1.00206e+00    7.65942e-03    1.01174e+00    1.07689e-02
        43      9.75931e-01    1.21600e+00    1.00142e+00    7.49740e-03    1.01091e+00    1.05484e-02
        44      9.12735e-01    1.23733e+00    9.99310e-01    7.61533e-03    1.00868e+00    1.05445e-02
        45      9.54701e-01    1.25867e+00    9.98273e-01    7.50813e-03    1.00748e+00    1.03771e-02
        46      1.05862e+00    1.28000e+00    9.99644e-01    7.46263e-03    1.00859e+00    1.02096e-02
        47      1.09711e+00    1.32267e+00    1.00181e+00    7.60965e-03    1.01047e+00    1.01660e-02
        48      1.03034e+00    1.34400e+00    1.00243e+00    7.46818e-03    1.01089e+00    9.96039e-03
        49      1.05597e+00    1.36533e+00    1.00357e+00    7.39582e-03    1.01181e+00    9.79827e-03
        50      1.04596e+00    1.40800e+00    1.00445e+00    7.29377e-03    1.01249e+00    9.62467e-03
        51      1.01392e+00    1.42933e+00    1.00465e+00    7.14598e-03    1.01252e+00    9.43411e-03
        52      9.73808e-01    1.45067e+00    1.00403e+00    7.02872e-03    1.01177e+00    9.28090e-03
        53      1.01788e+00    1.47200e+00    1.00430e+00    6.89487e-03    1.01189e+00    9.10475e-03

  keno message number k5-123          execution terminated due to completion of the specified number of generations.

  the matrix k-effective is the largest eigenvalue of the fission production by unit number matrix.
```

Figure F11.6.35 Example of k-effectives by generation

The k-effectives for each generation are printed in subroutine FISFLX. The headings are printed by subroutine GUIDE. The problem title is printed at the top of the page. A descriptive heading is printed at the top of each column of data. The data that are printed include (1) the generation number, (2) the k-effective calculated for the generation, (3) the elapsed CPU time in minutes, (4) the average value of k-effective through the current generation (excluding the first two generations), (5) the deviation associated with the average k-effective, and (6) the matrix k-effective for the generation and the deviation associated with the matrix k-effective. The last two columns are filled with zeros if the user did not specify matrix k-effective calculations. The matrix k-effective is the largest eigenvalue of the fission production matrix. Matrix information can be calculated based on (1) position index, (2) unit number, (3) hole number, and (4) array number. The matrix k-effective printed in the sixth column is based on this order. If the matrix k-effective is calculated by position index, it is the one printed in the sixth column. The matrix k-effective by unit number is given second preference, followed by hole number and then array number. After the last generation, a message is printed to indicate why execution was terminated. If matrix k-effectives were calculated, this is followed by a message stating the method used to determine the matrix k-effective.

The user should examine this portion of the printed results to ensure that the two methods of calculating k-effective are in acceptable agreement and to verify that the average value of k-effective has become relatively stable. If the k-effectives appear to be oscillating or drifting significantly, the user should consider rerunning the problem with a larger number of histories per generation.

If a problem is restarted, the generation numbers and k-effectives are printed and the words FROM RESTART UNIT are printed in the elapsed time column. All other columns are blank. When the generation at which the problem is to be restarted is reached, the print reverts to the normal format as shown in Fig. F11.6.35.

F11.6.24 FINAL K-EFFECTIVE EDIT

The final edit of the k-effectives is printed by subroutine KEDIT following the k-effectives by generation. The title is printed at the top of the page, followed by the lifetime and the generation time and their associated deviations. The lifetime is the average lifespan of a neutron (in seconds) from the time it is born until it is absorbed or leaks from the system. The generation time is the average time (in seconds) between successive neutron generations. If NUB=YES is specified in the parameter data, (Sect. F11.4.3) the average number of neutrons per fission, NU BAR and its associated deviation are printed and the AVERAGE FISSION GROUP (the average energy group at which fission occurs) and its associated deviation are printed. Then the ENERGY(EV) OF THE AVERAGE LETHARGY CAUSING FISSION and its associated deviation are printed. If SMU=YES is specified in the parameter data, the average selfmultiplication of a unit and its associated deviation is printed. This self-multiplication results from fissions caused by neutrons born in the unit. Fissions caused by neutrons that exit the unit and return are not included. Then the final k-effective edit is printed as shown in Fig. F11.6.36.

The final k-effective edit prints the average k-effective and its associated deviation and the limits of k-effective for the 67, 95, and 99% confidence intervals. The number of histories used in calculating the average k-effective is also printed. This is done skipping various numbers of generations. The user should carefully examine the final k-effective edit to determine if the average k-effective is relatively stable. If a noticeable drift is apparent as the number of initial generations skipped increases, it may indicate a problem in converging the source. If this appears to be the case, the problem should be rerun with a better initial source distribution and should be run for a sufficient number of generations that the average k-effective becomes stable.

```
                                            sample problem 18   1f27 demonstration of options problem

lifetime =  2.59459e-04 + or -  1.03486e-05            generation time =  2.09643e-04 + or -  8.38685e-06
nu bar   =  2.42305e+00 + or -  6.89977e-05       average fission group =  2.20786e+01 + or -  1.99050e-02
                   energy(ev) of the average lethargy causing fission =  1.81656e-01 + or -  2.58105e-03

no. of initial
 generations     average                         67 per cent           95 per cent           99 per cent        number of
   skipped     k-effective       deviation    confidence interval   confidence interval   confidence interval   histories

      3          1.01402    + or - 0.00752    1.00650 to 1.02153    0.99898 to 1.02905    0.99146 to 1.03657      17500
      4          1.01487    + or - 0.00792    1.00696 to 1.02279    0.99904 to 1.03071    0.99112 to 1.03862      17150
      5          1.01755    + or - 0.00692    1.01063 to 1.02447    1.00371 to 1.03139    0.99680 to 1.03831      16800
      6          1.01891    + or - 0.00710    1.01180 to 1.02601    1.00470 to 1.03311    0.99760 to 1.04021      16450
      7          1.01965    + or - 0.00688    1.01277 to 1.02654    1.00588 to 1.03342    0.99900 to 1.04031      16100
      8          1.02085    + or - 0.00713    1.01372 to 1.02799    1.00659 to 1.03512    0.99945 to 1.04226      15750
      9          1.02181    + or - 0.00684    1.01497 to 1.02865    1.00813 to 1.03549    1.00128 to 1.04234      15400
     10          1.02158    + or - 0.00743    1.01416 to 1.02901    1.00673 to 1.03644    0.99930 to 1.04386      15050
     11          1.02087    + or - 0.00711    1.01377 to 1.02798    1.00666 to 1.03509    0.99955 to 1.04220      14700
     12          1.01993    + or - 0.00759    1.01233 to 1.02752    1.00474 to 1.03511    0.99715 to 1.04271      14350
     17          1.02100    + or - 0.00792    1.01308 to 1.02892    1.00517 to 1.03683    0.99725 to 1.04475      12600
     22          1.02352    + or - 0.00957    1.01395 to 1.03309    1.00438 to 1.04266    0.99481 to 1.05223      10850
     27          1.02720    + or - 0.00878    1.01842 to 1.03598    1.00964 to 1.04476    1.00086 to 1.05354       9100
     32          1.02174    + or - 0.00979    1.01195 to 1.03153    1.00217 to 1.04132    0.99238 to 1.05110       7350
     37          1.01840    + or - 0.01140    1.00700 to 1.02980    0.99560 to 1.04120    0.98420 to 1.05260       5600
     42          1.01089    + or - 0.01625    0.99464 to 1.02714    0.97839 to 1.04339    0.96213 to 1.05964       3850
     47          1.03967    + or - 0.01791    1.02177 to 1.05758    1.00386 to 1.07548    0.98595 to 1.09339       2100
```

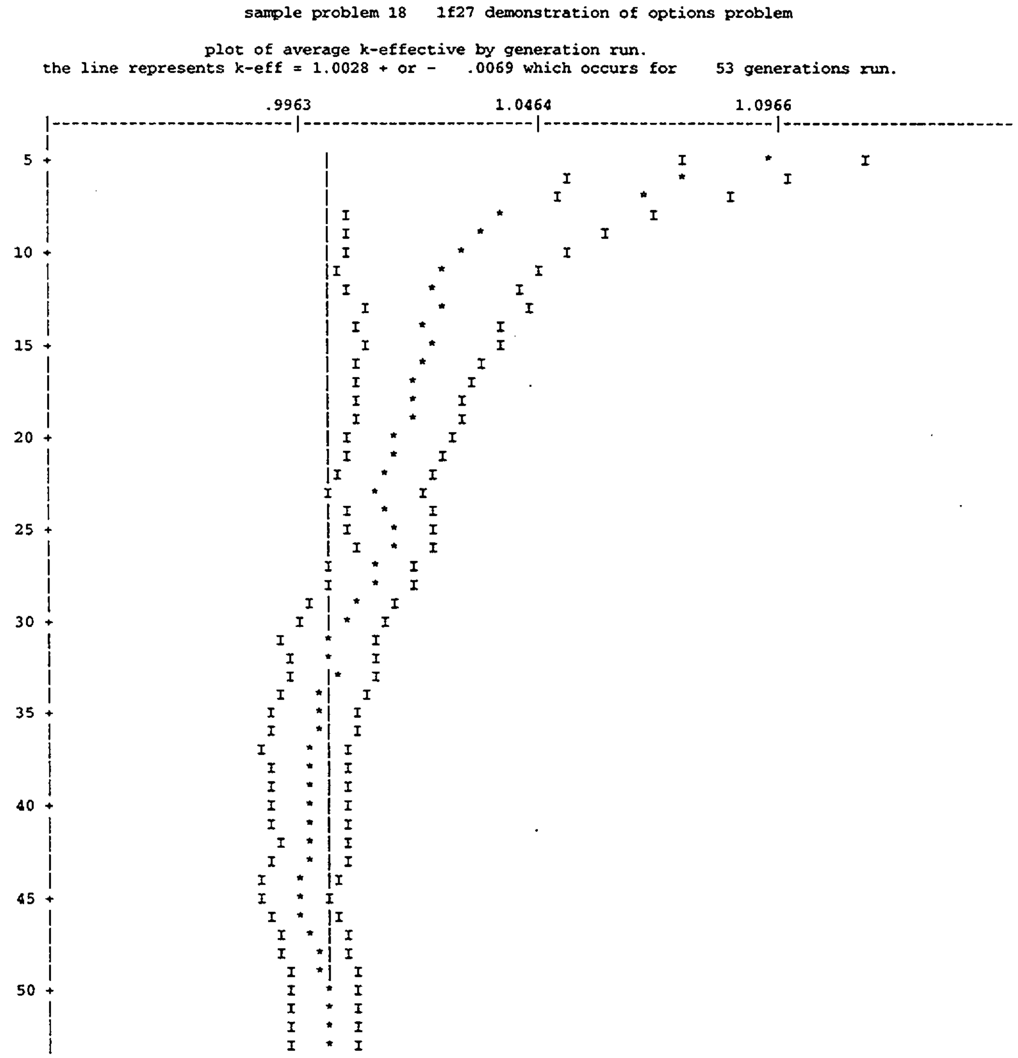
Figure F11.6.36 Example of the final k-effective edit

F11.6.25 PLOT OF AVERAGE K-EFFECTIVE BY GENERATION RUN

This plot consists of average k-effectives plotted versus the number of generations run. The limits of one standard deviation are plotted on either side of each average k-effective. These average k-effectives are not necessarily the same as the average k-effectives described in Sect. F11.6.23. The code omits the k-effectives of the first *nskip* generations when the average k-effectives for this plot are calculated. Although the k-effective of the *nskip* generation is summed into the average k-effective, it is not plotted because standard deviations cannot be calculated for a single point. Thus, if *nskip* is 3 (i.e., the first three generations are skipped), the first value plotted is the average k-effective corresponding to the fifth generation. The dotted line represents the value of the average k-effective corresponding to the smallest deviation when the average k-effective and its deviation are computed for each generation over the range of nskip through the total number of generations. Figure F11.6.37 is an example of this type of plot. The primary use for this plot is to determine if the problem has source convergence difficulties.

F11.6.26 PLOT OF AVERAGE K-EFFECTIVE BY GENERATION SKIPPED

This plot illustrates the average k-effective versus the number of generations skipped as shown in Fig. F11.6.38. The limits of one standard deviation are plotted on either side of the average k-effective. The dotted line represents the value of the average k-effective corresponding to the smallest deviation when the average k-effective and its deviation are computed for the number of generations skipped over the range of nskip+1 through 2/3 the total number of generations calculated. The plot is essentially a plot of the data described in Sect. F11.6.24. It is useful for determining if source convergence has been achieved.



```
                    sample problem 18   1f27 demonstration of options problem

                plot of average k-effective by generation run.
the line represents k-eff = 1.0028 + or -   .0069 which occurs for    53 generations run.
```

Figure F11.6.37 Sample plot of average k-effective by generation run

```
               sample problem 18   1f27 demonstration of options problem

           plot of average k-effective by generation skipped.
 the line represents k-eff =  .9989 + or -   .0065 which occurs for     5 generations skipped.
```

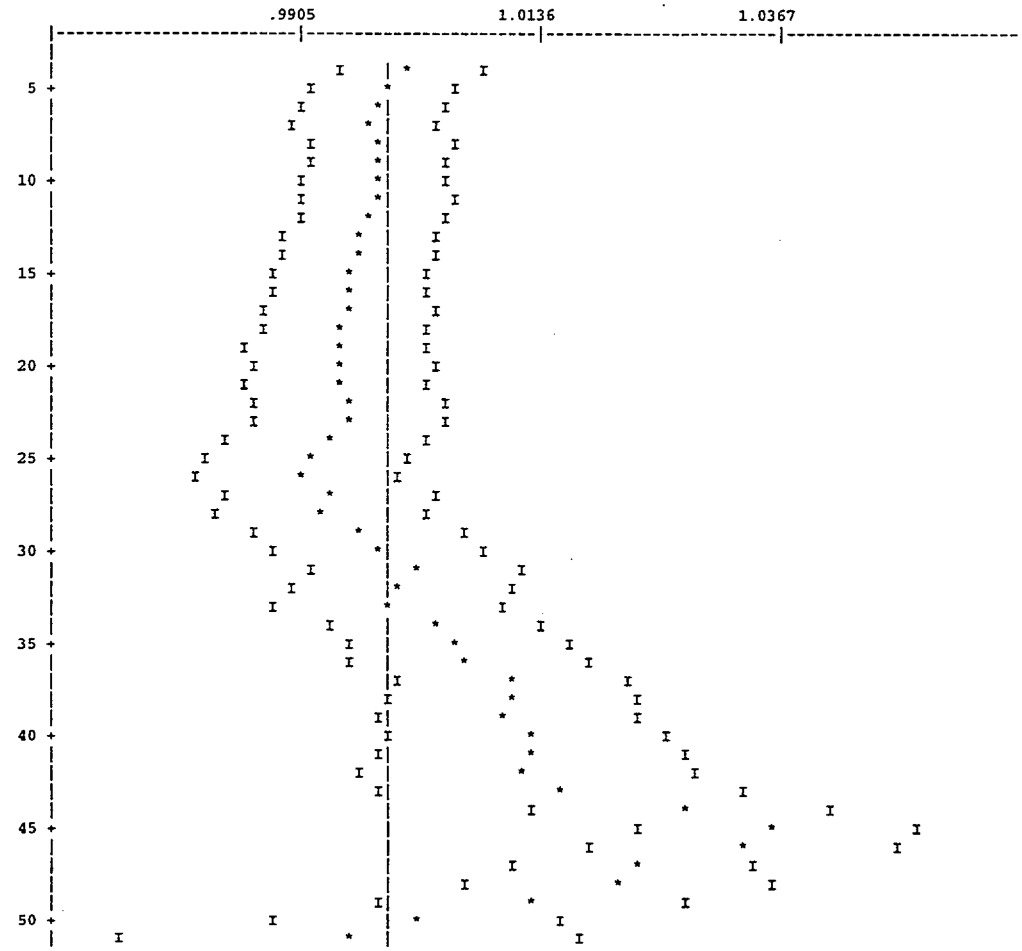


Figure F11.6.38 Sample plot of average k-effective by generation skipped

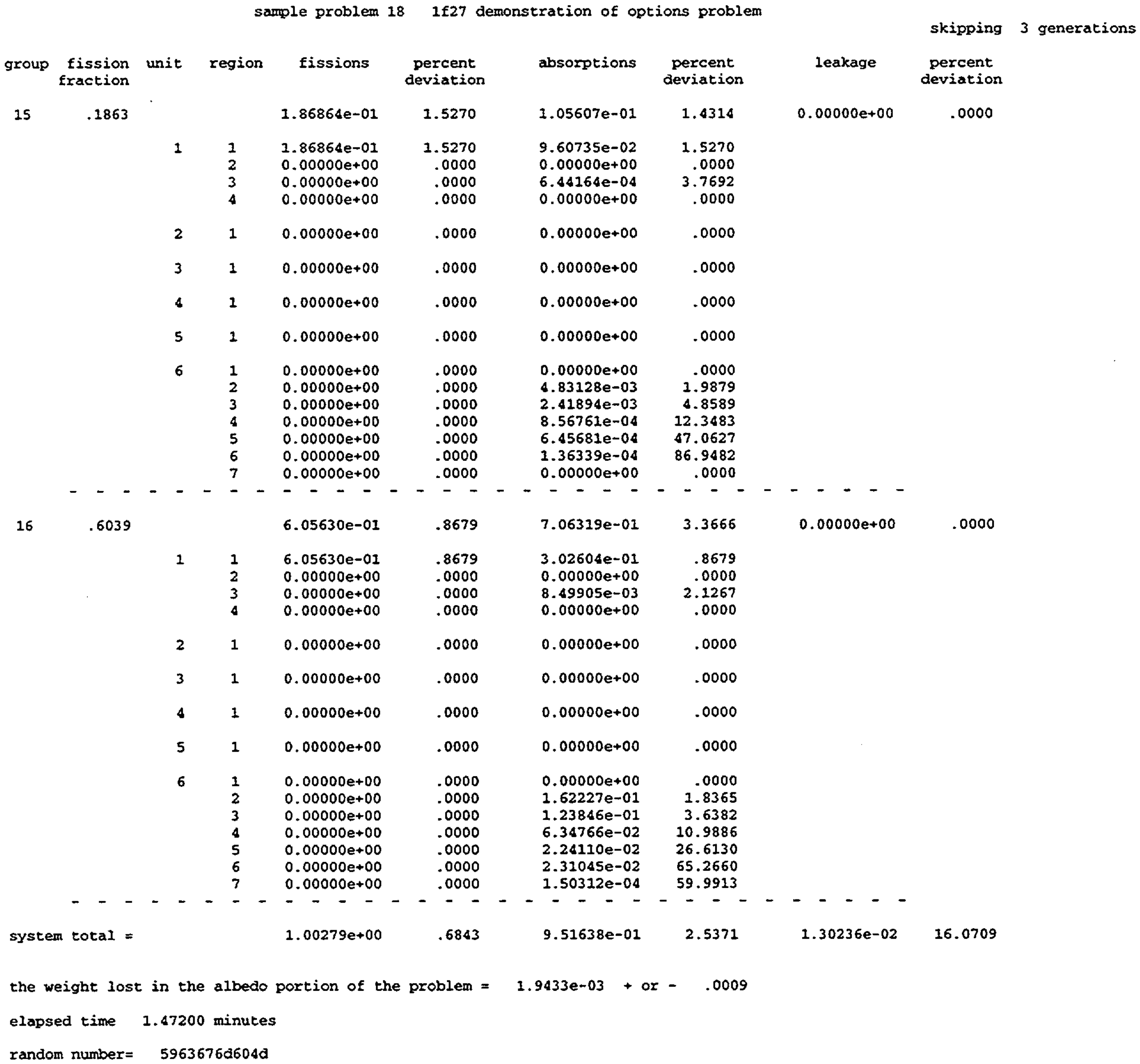
F11.6.27 FINAL EDIT OF FISSIONS, ABSORPTIONS, AND LEAKAGE

The final edit of fissions, absorptions, and leakage is printed by subroutine KEDIT after the final k-effective edit. It prints the fission fraction for each group and the fission production, absorptions, and leakage with their associated percent deviation. Examples of the final edits of fissions, absorptions, and leakage are shown in Figs. F11.6.39 through F11.6.42.

If FAR=NO is specified, region-dependent fissions and absorptions are not printed. Figures F11.6.39 and F11.6.42 demonstrate the printout if FAR=NO.

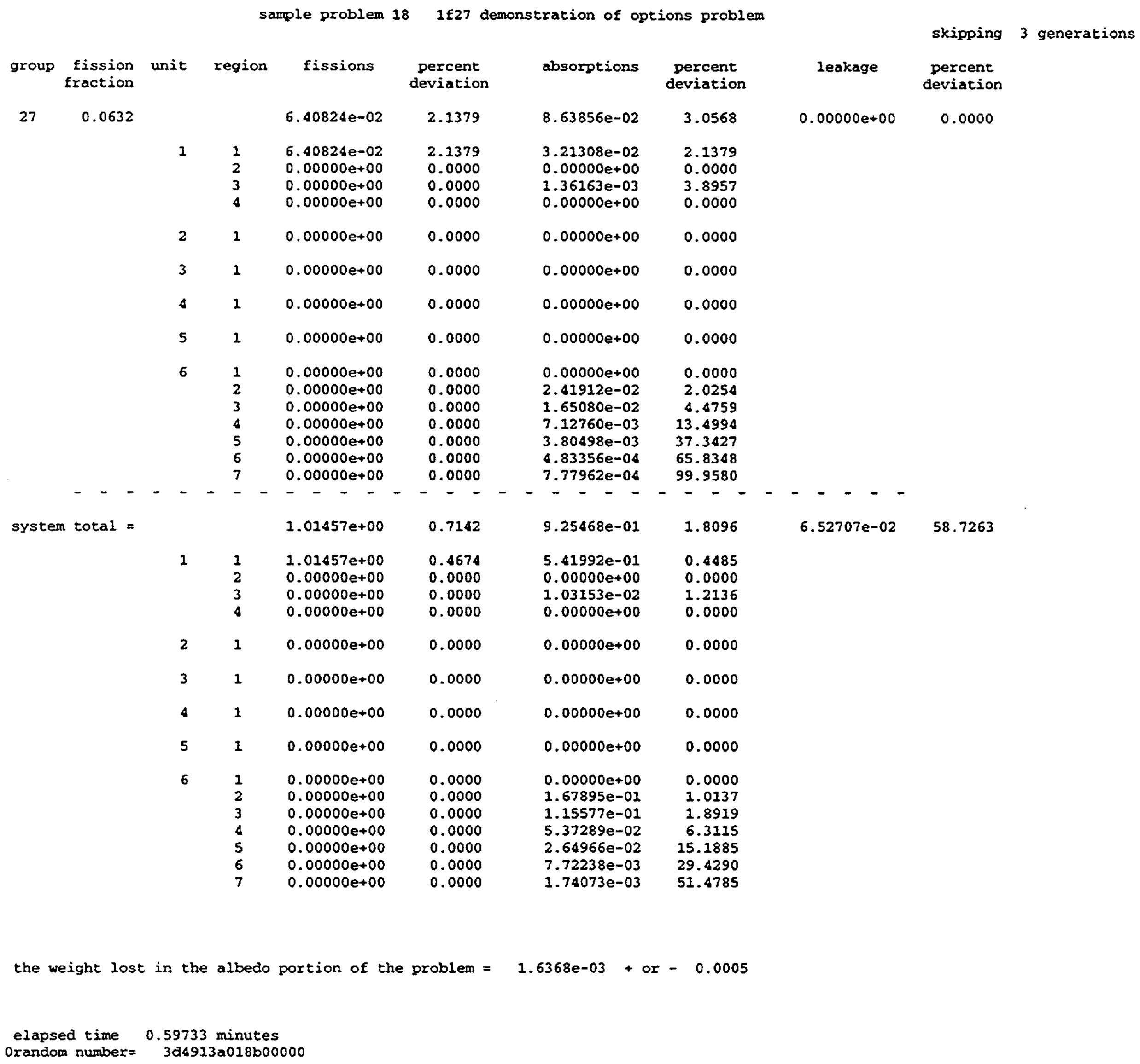
```
                           sample problem 2  2c8 bare with 8 unit types matrix calculation
                                                                                         skipping  3 generations

group  fission  unit   region    fissions      percent      absorptions    percent       leakage       percent
       fraction                               deviation                   deviation                   deviation

  1      .1439                 1.43327e-01    1.6542      5.06130e-02    1.6542      7.82282e-02    2.0448

  2      .2391                 2.38112e-01     .8605      9.69963e-02     .8605      1.44757e-01    1.3068

  3      .1429                 1.42343e-01    1.6270      6.15892e-02    1.6270      8.82221e-02    1.8458

  4      .2303                 2.29375e-01    1.2187      1.04317e-01    1.2187      1.50874e-01    1.0063

  5      .1940                 1.93177e-01    1.3393      9.17772e-02    1.3393      9.41416e-02    1.5240

  6      .0490                 4.87624e-02    2.5599      2.53366e-02    2.5599      1.25316e-02    3.3854

  7      .0007                 7.31275e-04   21.1490      3.94460e-04   21.1490      1.12024e-04   32.0667

  8      .0000                 0.00000e+00     .0000      0.00000e+00     .0000      0.00000e+00     .0000

  9      .0000                 0.00000e+00     .0000      0.00000e+00     .0000      0.00000e+00     .0000

 10      .0000                 0.00000e+00     .0000      0.00000e+00     .0000      0.00000e+00     .0000

 11      .0000                 0.00000e+00     .0000      0.00000e+00     .0000      0.00000e+00     .0000

 12      .0000                 0.00000e+00     .0000      0.00000e+00     .0000      0.00000e+00     .0000

 13      .0000                 0.00000e+00     .0000      0.00000e+00     .0000      0.00000e+00     .0000

 14      .0000                 0.00000e+00     .0000      0.00000e+00     .0000      0.00000e+00     .0000

 15      .0000                 0.00000e+00     .0000      0.00000e+00     .0000      0.00000e+00     .0000

 16      .0000                 0.00000e+00     .0000      0.00000e+00     .0000      0.00000e+00     .0000

system total =                 9.95826e-01     .4520      4.31024e-01     .4528      5.68867e-01     .3422

elapsed time    .21333 minutes

random number=   6567269e6d36
```

Figure F11.6.39 Sample of the final edit of fissions, absorptions, and leakage with all region-dependent information suppressed



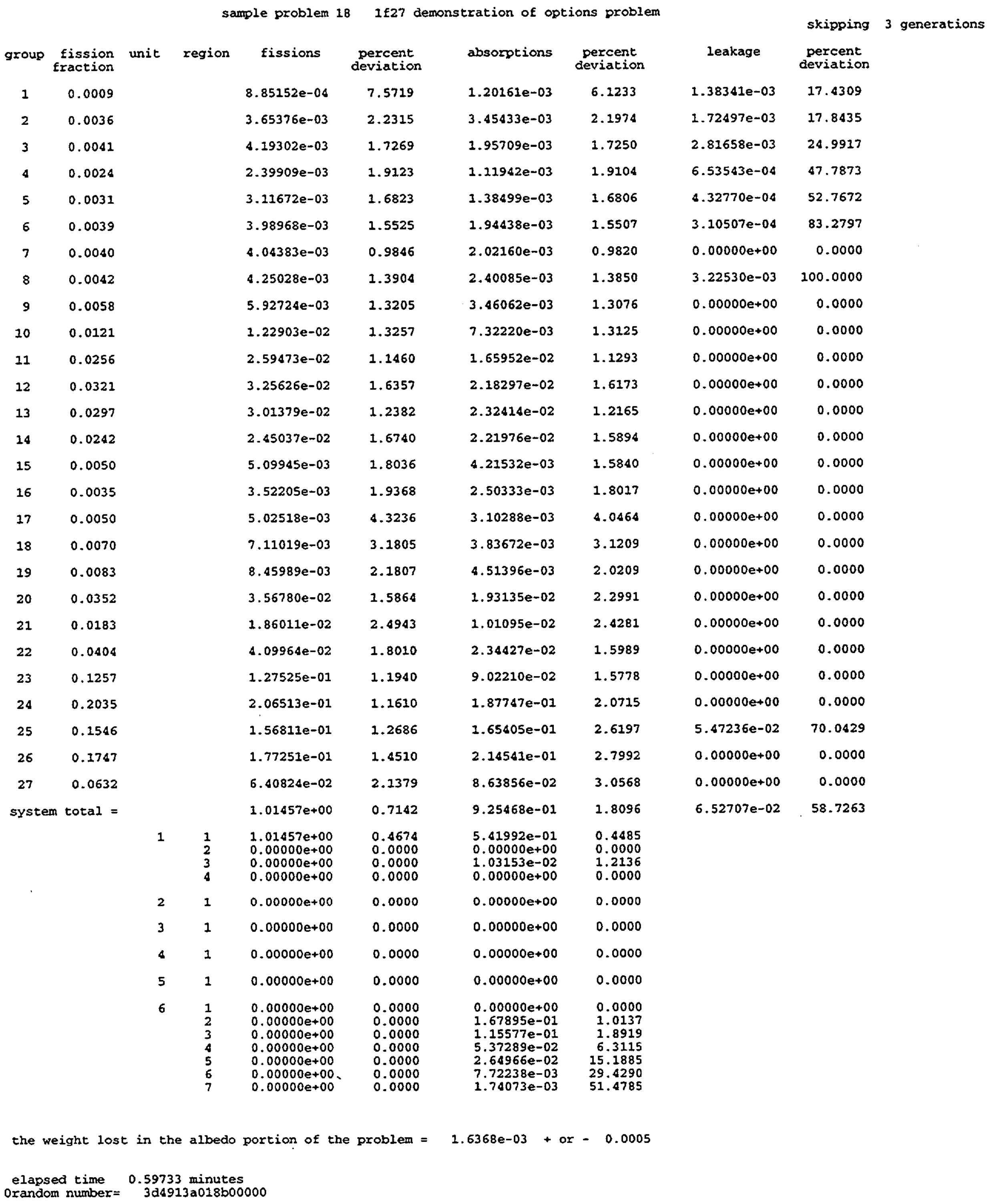
```
                              sample problem 18   1f27 demonstration of options problem
                                                                                                  skipping  3 generations

group  fission  unit   region    fissions      percent       absorptions     percent          leakage       percent
       fraction                               deviation                     deviation                      deviation

 15     .1863                   1.86864e-01    1.5270       1.05607e-01     1.4314        0.00000e+00       .0000

                  1      1      1.86864e-01    1.5270       9.60735e-02     1.5270
                         2      0.00000e+00     .0000       0.00000e+00      .0000
                         3      0.00000e+00     .0000       6.44164e-04     3.7692
                         4      0.00000e+00     .0000       0.00000e+00      .0000

                  2      1      0.00000e+00     .0000       0.00000e+00      .0000

                  3      1      0.00000e+00     .0000       0.00000e+00      .0000

                  4      1      0.00000e+00     .0000       0.00000e+00      .0000

                  5      1      0.00000e+00     .0000       0.00000e+00      .0000

                  6      1      0.00000e+00     .0000       0.00000e+00      .0000
                         2      0.00000e+00     .0000       4.83128e-03     1.9879
                         3      0.00000e+00     .0000       2.41894e-03     4.8589
                         4      0.00000e+00     .0000       8.56761e-04    12.3483
                         5      0.00000e+00     .0000       6.45681e-04    47.0627
                         6      0.00000e+00     .0000       1.36339e-04    86.9482
                         7      0.00000e+00     .0000       0.00000e+00      .0000
      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

 16     .6039                   6.05630e-01     .8679       7.06319e-01     3.3666        0.00000e+00       .0000

                  1      1      6.05630e-01     .8679       3.02604e-01      .8679
                         2      0.00000e+00     .0000       0.00000e+00      .0000
                         3      0.00000e+00     .0000       8.49905e-03     2.1267
                         4      0.00000e+00     .0000       0.00000e+00      .0000

                  2      1      0.00000e+00     .0000       0.00000e+00      .0000

                  3      1      0.00000e+00     .0000       0.00000e+00      .0000

                  4      1      0.00000e+00     .0000       0.00000e+00      .0000

                  5      1      0.00000e+00     .0000       0.00000e+00      .0000

                  6      1      0.00000e+00     .0000       0.00000e+00      .0000
                         2      0.00000e+00     .0000       1.62227e-01     1.8365
                         3      0.00000e+00     .0000       1.23846e-01     3.6382
                         4      0.00000e+00     .0000       6.34766e-02    10.9886
                         5      0.00000e+00     .0000       2.24110e-02    26.6130
                         6      0.00000e+00     .0000       2.31045e-02    65.2660
                         7      0.00000e+00     .0000       1.50312e-04    59.9913
      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

system total =                  1.00279e+00     .6843       9.51638e-01     2.5371        1.30236e-02     16.0709

the weight lost in the albedo portion of the problem =   1.9433e-03  + or -   .0009

elapsed time   1.47200 minutes

random number=   5963676d604d
```

Figure F11.6.40 Sample of the final edit of region-dependent fissions, absorptions, and leakage with the region-dependent totals suppressed



```
                                 sample problem 18   1f27 demonstration of options problem
                                                                                                    skipping  3 generations
group  fission  unit   region    fissions     percent      absorptions    percent        leakage       percent
       fraction                               deviation                   deviation                    deviation

 27     0.0632                 6.40824e-02    2.1379      8.63856e-02    3.0568      0.00000e+00     0.0000

                  1      1     6.40824e-02    2.1379      3.21308e-02    2.1379
                         2     0.00000e+00    0.0000      0.00000e+00    0.0000
                         3     0.00000e+00    0.0000      1.36163e-03    3.8957
                         4     0.00000e+00    0.0000      0.00000e+00    0.0000

                  2      1     0.00000e+00    0.0000      0.00000e+00    0.0000

                  3      1     0.00000e+00    0.0000      0.00000e+00    0.0000

                  4      1     0.00000e+00    0.0000      0.00000e+00    0.0000

                  5      1     0.00000e+00    0.0000      0.00000e+00    0.0000

                  6      1     0.00000e+00    0.0000      0.00000e+00    0.0000
                         2     0.00000e+00    0.0000      2.41912e-02    2.0254
                         3     0.00000e+00    0.0000      1.65080e-02    4.4759
                         4     0.00000e+00    0.0000      7.12760e-03   13.4994
                         5     0.00000e+00    0.0000      3.80498e-03   37.3427
                         6     0.00000e+00    0.0000      4.83356e-04   65.8348
                         7     0.00000e+00    0.0000      7.77962e-04   99.9580
      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

system total =                 1.01457e+00    0.7142      9.25468e-01    1.8096      6.52707e-02    58.7263

                  1      1     1.01457e+00    0.4674      5.41992e-01    0.4485
                         2     0.00000e+00    0.0000      0.00000e+00    0.0000
                         3     0.00000e+00    0.0000      1.03153e-02    1.2136
                         4     0.00000e+00    0.0000      0.00000e+00    0.0000

                  2      1     0.00000e+00    0.0000      0.00000e+00    0.0000

                  3      1     0.00000e+00    0.0000      0.00000e+00    0.0000

                  4      1     0.00000e+00    0.0000      0.00000e+00    0.0000

                  5      1     0.00000e+00    0.0000      0.00000e+00    0.0000

                  6      1     0.00000e+00    0.0000      0.00000e+00    0.0000
                         2     0.00000e+00    0.0000      1.67895e-01    1.0137
                         3     0.00000e+00    0.0000      1.15577e-01    1.8919
                         4     0.00000e+00    0.0000      5.37289e-02    6.3115
                         5     0.00000e+00    0.0000      2.64966e-02   15.1885
                         6     0.00000e+00    0.0000      7.72238e-03   29.4290
                         7     0.00000e+00    0.0000      1.74073e-03   51.4785


 the weight lost in the albedo portion of the problem =    1.6368e-03  + or -  0.0005


 elapsed time   0.59733 minutes
0random number=   3d4913a018b00000
```

Figure F11.6.41 Sample of the final edit of region-dependent fissions, absorptions, and leakage and the region-dependent totals



```
                              sample problem 18    1f27 demonstration of options problem
                                                                                           skipping  3 generations

group  fission   unit    region     fissions      percent      absorptions     percent        leakage       percent
       fraction                                   deviation                    deviation                    deviation

  1     0.0009                     8.85152e-04    7.5719      1.20161e-03     6.1233       1.38341e-03    17.4309
  2     0.0036                     3.65376e-03    2.2315      3.45433e-03     2.1974       1.72497e-03    17.8435
  3     0.0041                     4.19302e-03    1.7269      1.95709e-03     1.7250       2.81658e-03    24.9917
  4     0.0024                     2.39909e-03    1.9123      1.11942e-03     1.9104       6.53543e-04    47.7873
  5     0.0031                     3.11672e-03    1.6823      1.38499e-03     1.6806       4.32770e-04    52.7672
  6     0.0039                     3.98968e-03    1.5525      1.94438e-03     1.5507       3.10507e-04    83.2797
  7     0.0040                     4.04383e-03    0.9846      2.02160e-03     0.9820       0.00000e+00     0.0000
  8     0.0042                     4.25028e-03    1.3904      2.40085e-03     1.3850       3.22530e-03   100.0000
  9     0.0058                     5.92724e-03    1.3205      3.46062e-03     1.3076       0.00000e+00     0.0000
 10     0.0121                     1.22903e-02    1.3257      7.32220e-03     1.3125       0.00000e+00     0.0000
 11     0.0256                     2.59473e-02    1.1460      1.65952e-02     1.1293       0.00000e+00     0.0000
 12     0.0321                     3.25626e-02    1.6357      2.18297e-02     1.6173       0.00000e+00     0.0000
 13     0.0297                     3.01379e-02    1.2382      2.32414e-02     1.2165       0.00000e+00     0.0000
 14     0.0242                     2.45037e-02    1.6740      2.21976e-02     1.5894       0.00000e+00     0.0000
 15     0.0050                     5.09945e-03    1.8036      4.21532e-03     1.5840       0.00000e+00     0.0000
 16     0.0035                     3.52205e-03    1.9368      2.50333e-03     1.8017       0.00000e+00     0.0000
 17     0.0050                     5.02518e-03    4.3236      3.10288e-03     4.0464       0.00000e+00     0.0000
 18     0.0070                     7.11019e-03    3.1805      3.83672e-03     3.1209       0.00000e+00     0.0000
 19     0.0083                     8.45989e-03    2.1807      4.51396e-03     2.0209       0.00000e+00     0.0000
 20     0.0352                     3.56780e-02    1.5864      1.93135e-02     2.2991       0.00000e+00     0.0000
 21     0.0183                     1.86011e-02    2.4943      1.01095e-02     2.4281       0.00000e+00     0.0000
 22     0.0404                     4.09964e-02    1.8010      2.34427e-02     1.5989       0.00000e+00     0.0000
 23     0.1257                     1.27525e-01    1.1940      9.02210e-02     1.5778       0.00000e+00     0.0000
 24     0.2035                     2.06513e-01    1.1610      1.87747e-01     2.0715       0.00000e+00     0.0000
 25     0.1546                     1.56811e-01    1.2686      1.65405e-01     2.6197       5.47236e-02    70.0429
 26     0.1747                     1.77251e-01    1.4510      2.14541e-01     2.7992       0.00000e+00     0.0000
 27     0.0632                     6.40824e-02    2.1379      8.63856e-02     3.0568       0.00000e+00     0.0000
 system total =                    1.01457e+00    0.7142      9.25468e-01     1.8096       6.52707e-02    58.7263
                  1        1       1.01457e+00    0.4674      5.41992e-01     0.4485
                           2       0.00000e+00    0.0000      0.00000e+00     0.0000
                           3       0.00000e+00    0.0000      1.03153e-02     1.2136
                           4       0.00000e+00    0.0000      0.00000e+00     0.0000

                  2        1       0.00000e+00    0.0000      0.00000e+00     0.0000

                  3        1       0.00000e+00    0.0000      0.00000e+00     0.0000

                  4        1       0.00000e+00    0.0000      0.00000e+00     0.0000

                  5        1       0.00000e+00    0.0000      0.00000e+00     0.0000

                  6        1       0.00000e+00    0.0000      0.00000e+00     0.0000
                           2       0.00000e+00    0.0000      1.67895e-01     1.0137
                           3       0.00000e+00    0.0000      1.15577e-01     1.8919
                           4       0.00000e+00    0.0000      5.37289e-02     6.3115
                           5       0.00000e+00    0.0000      2.64966e-02    15.1885
                           6       0.00000e+00    0.0000      7.72238e-03    29.4290
                           7       0.00000e+00    0.0000      1.74073e-03    51.4785


 the weight lost in the albedo portion of the problem =    1.6368e-03   + or -   0.0005

 elapsed time    0.59733 minutes
0random number=   3d4913a018b00000
```

Figure F11.6.42 Sample of the final edit of fissions, absorptions, and leakage and the region-dependent totals

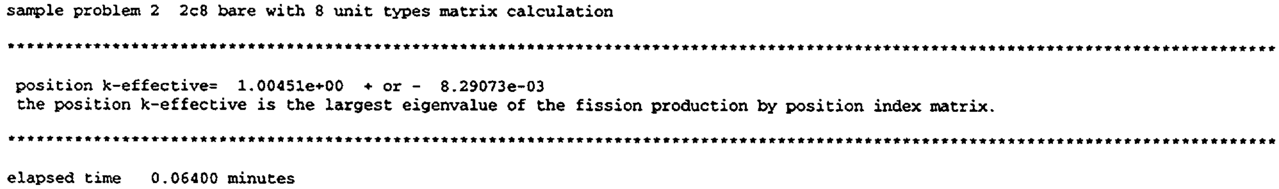
If FAR=YES is specified in the parameter data, the fissions, and absorptions for each geometry region used in the problem are printed for each energy group as shown in Figs. F11.6.40 and F11.6.41. Leakage is not collected by geometry region but rather represents the leakage from the system. GROUP is the energy group number, FISSION FRACTION is the fraction of the fissions that occur in that energy group. The percent deviation for the fission fraction is the same as that of the fissions in the same group. The heading, UNIT, refers to the unit or box type, and REGION is the region number within the specified unit or box type. The geometry regions are numbered sequentially within each unit, starting with 1. The sum of the fissions for every region for a given energy group is the total printed for that energy group. The same is true of absorptions. The fissions, absorptions, and leakages are given in units of "per source neutron." The SYSTEM TOTAL is the sum, over all the energy groups, of the fissions, absorptions and leakage. The associated percent deviation is printed for each.

The parameter GAS is used to control printing the total fission productions and absorptions for each geometry region, summed over all energy groups. GAS=YES causes these totals to be printed after the SYSTEM TOTAL as shown in Figs. F11.6.41 and F11.6.42. GAS=NO suppresses printing these data as shown in Figs. F11.6.39 and F11.6.40.

The sum of the leakage and absorptions printed for the system total should be close to 1. The fissions printed for the system total should be the same as the first k-effective printed in the final k-effective edit described in Sect. F11.6.24. If differential albedos are used, the leakage does not include the weight lost in the albedo reflection. A message stating the weight lost in the albedo is printed. This is the weight lost due to absorptions in the albedo reflector and leakage from the albedo reflector. No leakage is associated with faces having specular, mirror image, or periodic reflection. Thus there is no leakage associated with an infinite problem. The total elapsed time and final random number are printed at the end of this edit.

F11.6.28 MATRIX K-EFFECTIVE BY POSITION INDEX

The matrix k-effective by unit location (also referred to as array position or position index) is calculated if MKP=YES is specified in the parameter data (Sect. F11.4.3). It is the largest eigenvalue of the fission production matrix, collected by position index. The position index is a number referencing a position in a 3-D lattice. An example of the matrix k-effective by unit location is given in Fig. F11.6.43. It is contained within two rows of asterisks to draw attention to it.



```
sample problem 2  2c8 bare with 8 unit types matrix calculation

***********************************************************************************************************************************

 position k-effective=  1.00451e+00  + or -  8.29073e-03
 the position k-effective is the largest eigenvalue of the fission production by position index matrix.

***********************************************************************************************************************************

elapsed time   0.06400 minutes
```

Figure F11.6.43 Example of matrix k-effective by position index

F11.6.29 FISSION PRODUCTION BY POSITION INDEX MATRIX

To obtain this information, the user must specify MKP=YES and FMP=YES in the parameter data. It is then printed by subroutine MATRIX. The number of entries in the fission production matrix by position index is the square of the array size. Thus for a 2 × 2 × 2 array there are 64 entries, and for a 4 × 4 × 4 array there are 4096 entries in the fission production matrix by position index. An example of the fission production matrix by position index for a 2 × 2 × 2 array is shown in Fig. F11.6.44.

```
fission production by position index matrix
  ( i,  j) p is the number of next generation neutrons produced at position index j by a neutron born at position index i.

  ( 0,  0) 0.00e+00  ( 0,  1) 0.00e+00  ( 0,  2) 0.00e+00  ( 0,  3) 0.00e+00  ( 0,  4) 0.00e+00  ( 0,  5) 0.00e+00
  ( 0,  6) 0.00e+00  ( 0,  7) 0.00e+00  ( 0,  8) 0.00e+00

  ( 1,  0) 0.00e+00  ( 1,  1) 7.23e-01  ( 1,  2) 5.27e-02  ( 1,  3) 5.35e-02  ( 1,  4) 2.42e-02  ( 1,  5) 7.88e-02
  ( 1,  6) 2.51e-02  ( 1,  7) 2.66e-02  ( 1,  8) 1.87e-02

  ( 2,  0) 0.00e+00  ( 2,  1) 5.00e-02  ( 2,  2) 7.41e-01  ( 2,  3) 2.68e-02  ( 2,  4) 5.28e-02  ( 2,  5) 2.61e-02
  ( 2,  6) 6.59e-02  ( 2,  7) 1.98e-02  ( 2,  8) 2.39e-02

  ( 3,  0) 0.00e+00  ( 3,  1) 4.75e-02  ( 3,  2) 2.84e-02  ( 3,  3) 7.28e-01  ( 3,  4) 6.10e-02  ( 3,  5) 2.58e-02
  ( 3,  6) 2.34e-02  ( 3,  7) 6.44e-02  ( 3,  8) 2.23e-02

  ( 4,  0) 0.00e+00  ( 4,  1) 2.91e-02  ( 4,  2) 5.38e-02  ( 4,  3) 5.56e-02  ( 4,  4) 7.18e-01  ( 4,  5) 1.89e-02
  ( 4,  6) 2.71e-02  ( 4,  7) 2.58e-02  ( 4,  8) 6.84e-02

  ( 5,  0) 0.00e+00  ( 5,  1) 7.33e-02  ( 5,  2) 2.54e-02  ( 5,  3) 2.42e-02  ( 5,  4) 1.92e-02  ( 5,  5) 7.39e-01
  ( 5,  6) 5.54e-02  ( 5,  7) 5.06e-02  ( 5,  8) 2.89e-02

  ( 6,  0) 0.00e+00  ( 6,  1) 2.81e-02  ( 6,  2) 7.24e-02  ( 6,  3) 2.02e-02  ( 6,  4) 2.76e-02  ( 6,  5) 5.19e-02
  ( 6,  6) 7.16e-01  ( 6,  7) 2.98e-02  ( 6,  8) 6.58e-02

  ( 7,  0) 0.00e+00  ( 7,  1) 2.50e-02  ( 7,  2) 1.81e-02  ( 7,  3) 6.89e-02  ( 7,  4) 2.51e-02  ( 7,  5) 5.13e-02
  ( 7,  6) 2.98e-02  ( 7,  7) 7.25e-01  ( 7,  8) 5.36e-02

  ( 8,  0) 0.00e+00  ( 8,  1) 2.09e-02  ( 8,  2) 2.50e-02  ( 8,  3) 2.30e-02  ( 8,  4) 6.61e-02  ( 8,  5) 2.88e-02
  ( 8,  6) 5.68e-02  ( 8,  7) 5.85e-02  ( 8,  8) 7.24e-01
```

Figure F11.6.44 Sample fission production matrix by position index

The position index definition is given in Sect. F11.6.31. For each position index in the array, the number of next-generation neutrons produced at position index J per neutron born at position index I is determined. The fission production matrix by position index is used to determine the matrix k-effective, cofactor k-effective and source vector by position index.

F11.6.30 SOURCE VECTOR BY POSITION INDEX

This information is printed by subroutine MATRIX only if MKP=YES is specified in the parameter data. The source vector by position index is the eigenvector of the fission production matrix by position index and should sum to 1.0. It represents the fission source for the specified locations in the 3-D lattice representing the physical problem being analyzed. The position index is defined in Sect. F11.6.31. An example of the source vector by position index is shown in Fig. F11.6.45. The average self-multiplication by array position is the overall average of the self-multiplication of all units used in the problem.

```
sample problem 2  2c8 bare with 8 unit types matrix calculation

  source vector by position index


   index      vector
     0     0.00000e+00
     1     1.22999e-01
     2     1.29915e-01
     3     1.22339e-01
     4     1.20775e-01
     5     1.31622e-01
     6     1.24200e-01
     7     1.23447e-01
     8     1.24703e-01

                      average self multiplication by array position

the number of next generation neutrons produced in a unit located at a given position in the array by
a neutron born in that same unit is   7.27017e-01 + or -   2.90715e-03
```

Figure F11.6.45 Example of source vector by position index

F11.6.31 COFACTOR K-EFFECTIVE BY POSITION INDEX

These data are printed by subroutine MATRIX only if MKP=YES is specified in the parameter data. This means that the fission production matrix is collected by position index. Calculating and printing cofactor k-effectives by position index can be avoided by specifying CKP=NO in the parameter data. An example of the cofactor k-effective by position index is shown in Fig. F11.6.46. See Sect. F11.D for a description of the problem used for the example.

```
sample problem 2  2c8 bare with 8 unit types matrix calculation

      position    position                      cofactor
       index       x  y  z          unit       k-effective      deviation
         0         0  0  0            0        1.00452e+00     8.29076e-03
         1         1  1  1            1        9.66215e-01     8.83895e-03
         2         2  1  1            2        9.65953e-01     8.89594e-03
         3         1  2  1            3        9.67871e-01     8.81301e-03
         4         2  2  1            4        9.67524e-01     8.81828e-03
         5         1  1  2            5        9.63823e-01     8.91912e-03
         6         2  1  2            6        9.63424e-01     8.72140e-03
         7         1  2  2            7        9.67260e-01     8.74468e-03
         8         2  2  2            8        9.66186e-01     8.63773e-03

elapsed time   0.08533 minutes
```

Figure F11.6.46 Example of cofactor k-effective by position index

The cofactor k-effective for a given position index is the largest eigenvalue of the fission production matrix collected by position index, reduced by the row and column associated with that position index. Thus the cofactor k-effective is the value of k-effective for the system calculated without the fission source of the unit located at the specified position index.

The POSITION INDEX is a number referencing a position in a three-dimensional lattice. POSITION is the x, y, and z location within the lattice. UNIT is the unit or box type located at the specified location in the lattice. Thus, in Fig. F11.6.46, Unit 1 is located at the lower left-hand front corner of the array or 3-D lattice representing the problem (x=1, y=1, z=1) and the corresponding POSITION INDEX is 1. POSITION INDEX 8 is the top right-hand back corner of the lattice, POSITION x=2, y=2, z=2 and the unit located at that position is UNIT 8.

F11.6.32 MATRIX K-EFFECTIVE BY UNIT NUMBER

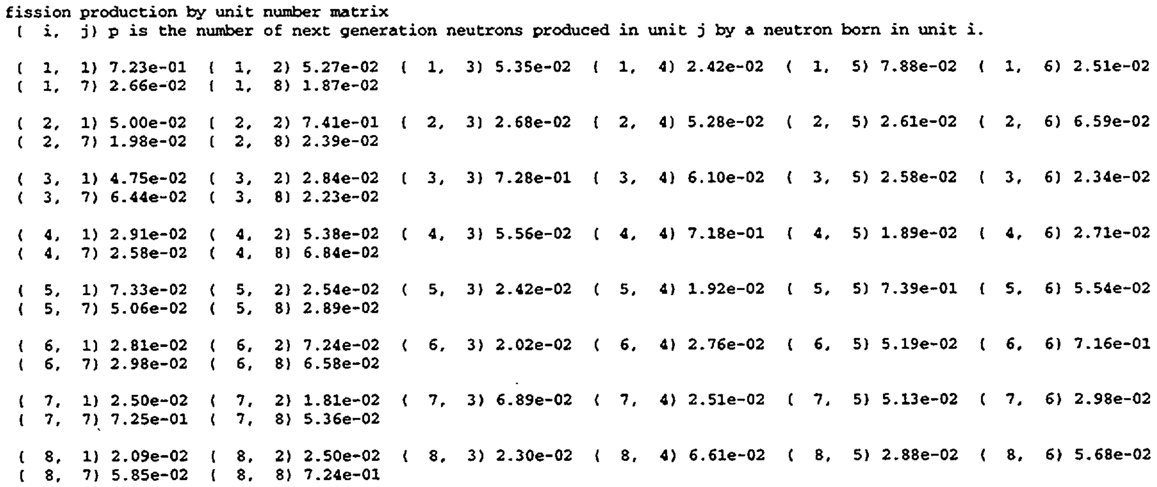
The matrix k-effective by unit number (unit k-effective) is the largest eigenvalue of the fission production by unit matrix. It is calculated only if MKU=YES is specified in the parameter data (Sect. F11.4.3). An example of the matrix k-effective by unit is given in Fig. F11.6.47.

```
sample problem 2  2c8 bare with 8 unit types matrix calculation
************************************************************************************************************************

 unit k-effective=  1.00451e+00  + or -  8.29073e-03
 the unit k-effective is the largest eigenvalue of the fission production by unit number matrix.

************************************************************************************************************************
elapsed time   0.08533 minutes
```

Figure F11.6.47 Example of matrix k-effective by unit number

F11.6.33 FISSION PRODUCTION BY UNIT NUMBER MATRIX

These data are printed by subroutine MATRIX only if MKU=YES is specified in the parameter data, which results in the code calculating the fission production matrix by unit. Thus, for each unit or box type in the array, the number of next-generation neutrons produced in Unit J per neutron born in Unit I is determined. This is the fission production matrix by unit, and it is used to determine the matrix k-effective by unit, the cofactor k-effective by unit, and the source vector by unit. An example of the fission production matrix by unit is shown in Fig. F11.6.48.



```
fission production by unit number matrix
 ( i,  j) p is the number of next generation neutrons produced in unit j by a neutron born in unit i.

 (  1,   1) 7.23e-01  (  1,   2) 5.27e-02  (  1,   3) 5.35e-02  (  1,   4) 2.42e-02  (  1,   5) 7.88e-02  (  1,   6) 2.51e-02
 (  1,   7) 2.66e-02  (  1,   8) 1.87e-02

 (  2,   1) 5.00e-02  (  2,   2) 7.41e-01  (  2,   3) 2.68e-02  (  2,   4) 5.28e-02  (  2,   5) 2.61e-02  (  2,   6) 6.59e-02
 (  2,   7) 1.98e-02  (  2,   8) 2.39e-02

 (  3,   1) 4.75e-02  (  3,   2) 2.84e-02  (  3,   3) 7.28e-01  (  3,   4) 6.10e-02  (  3,   5) 2.58e-02  (  3,   6) 2.34e-02
 (  3,   7) 6.44e-02  (  3,   8) 2.23e-02

 (  4,   1) 2.91e-02  (  4,   2) 5.38e-02  (  4,   3) 5.56e-02  (  4,   4) 7.18e-01  (  4,   5) 1.89e-02  (  4,   6) 2.71e-02
 (  4,   7) 2.58e-02  (  4,   8) 6.84e-02

 (  5,   1) 7.33e-02  (  5,   2) 2.54e-02  (  5,   3) 2.42e-02  (  5,   4) 1.92e-02  (  5,   5) 7.39e-01  (  5,   6) 5.54e-02
 (  5,   7) 5.06e-02  (  5,   8) 2.89e-02

 (  6,   1) 2.81e-02  (  6,   2) 7.24e-02  (  6,   3) 2.02e-02  (  6,   4) 2.76e-02  (  6,   5) 5.19e-02  (  6,   6) 7.16e-01
 (  6,   7) 2.98e-02  (  6,   8) 6.58e-02

 (  7,   1) 2.50e-02  (  7,   2) 1.81e-02  (  7,   3) 6.89e-02  (  7,   4) 2.51e-02  (  7,   5) 5.13e-02  (  7,   6) 2.98e-02
 (  7,   7) 7.25e-01  (  7,   8) 5.36e-02

 (  8,   1) 2.09e-02  (  8,   2) 2.50e-02  (  8,   3) 2.30e-02  (  8,   4) 6.61e-02  (  8,   5) 2.88e-02  (  8,   6) 5.68e-02
 (  8,   7) 5.85e-02  (  8,   8) 7.24e-01
```

Figure F11.6.48 An example of the fission probability matrix by unit

F11.6.34 SOURCE VECTOR BY UNIT NUMBER

These data are printed by subroutine MATRIX only if MKU=YES is specified in the parameter data. The source vector by unit is the eigenvector of the fission production matrix by unit. It represents the fission source for the units used in the problem. The components of the source vector should sum to 1.0. An example of the source vector by unit is given in Fig. F11.6.49. The average self-multiplication by unit is printed following the source vector. This value of self-multiplication includes those histories born in the unit which cause fissions in the same unit regardless of whether or not it exited and then returned. Therefore, this value will not agree with the value printed for the self-multiplication of the unit in Sect. F11.6.24 if the problem utilizes multiple units, the system is reflected, or a differential albedo is used in the problem.

```
sample problem 2  2c8 bare with 8 unit types matrix calculation

  source vector by unit


    unit     vector
     1    1.22999e-01
     2    1.29915e-01
     3    1.22339e-01
     4    1.20775e-01
     5    1.31622e-01
     6    1.24200e-01
     7    1.23447e-01
     8    1.24703e-01

                  average self multiplication by unit

the number of next generation neutrons produced in a unit by
a neutron born in that same unit is   7.27017e-01 + or -  2.90715e-03
```

Figure F11.6.49 Example of the source vector by unit

F11.6.35 COFACTOR K-EFFECTIVE BY UNIT NUMBER

Cofactor k-effectives are printed by subroutine MATRIX only if MKU=YES is specified in the parameter data. Calculating and printing cofactor k-effectives by unit can be avoided by specifying CKU=NO in the parameter data. The cofactor k-effective for a given unit is the k-effective of the system calculated without the fission source of that unit. This step is accomplished by determining the eigenvalue of the fission production matrix by unit after it has been reduced by the row and column associated with that unit. An example of the cofactor k-effective by unit is given in Fig. F11.6.50.

```
sample problem 2  2c8 bare with 8 unit types matrix calculation

                             cofactor
                     unit   k-effective     deviation
                      1    9.66215e-01    8.83895e-03
                      2    9.65953e-01    8.89594e-03
                      3    9.67871e-01    8.81301e-03
                      4    9.67524e-01    8.81828e-03
                      5    9.63823e-01    8.91912e-03
                      6    9.63424e-01    8.72140e-03
                      7    9.67260e-01    8.74468e-03
                      8    9.66186e-01    8.63773e-03

elapsed time   0.08533 minutes
```

Figure F11.6.50 Example of cofactor k-effective by unit number

F11.6.36 MATRIX K-EFFECTIVE BY HOLE NUMBER

The matrix k-effective by hole number is calculated if MKH=YES was specified in the parameter data, Sect. F11.4.3. It is the largest eigenvalue of the fission production matrix collected by hole number. An example of the matrix k-effective by hole number is given in Fig. F11.6.51.

```
sample problem 18   1f27 demonstration of options problem
************************************************************************************************************************************

 hole k-effective=  1.02170e+00  + or -  2.72020e-02
 the hole k-effective is the largest eigenvalue of the fission production by hole number matrix.

************************************************************************************************************************************

elapsed time   0.51200 minutes
```

Figure F11.6.51 Example of matrix k-effective by hole number

F11.6.37 FISSION PRODUCTION BY HOLE NUMBER MATRIX

This is the fission production matrix collected by hole number. It is printed only if MKH=YES and FMH=YES were specified in the parameter data, Sect. F11.4.3. An example of this fission production matrix is given in Fig. F11.6.52. This matrix indicates the number of next generation neutrons produced in hole number J by a neutron born in hole number I.

```
fission production by hole number matrix
  (  i,  j) p is the number of next generation neutrons produced in hole j by a neutron born in hole i.

  (  0,  0) 0.00e+00  (  0,  1) 0.00e+00  (  0,  2) 0.00e+00  (  0,  3) 0.00e+00  (  0,  4) 0.00e+00

  (  1,  0) 0.00e+00  (  1,  1) 7.75e-01  (  1,  2) 6.90e-02  (  1,  3) 9.84e-02  (  1,  4) 1.26e-01

  (  2,  0) 0.00e+00  (  2,  1) 1.29e-01  (  2,  2) 6.69e-01  (  2,  3) 9.21e-02  (  2,  4) 1.20e-01

  (  3,  0) 0.00e+00  (  3,  1) 1.09e-01  (  3,  2) 6.00e-02  (  3,  3) 6.97e-01  (  3,  4) 1.23e-01

  (  4,  0) 0.00e+00  (  4,  1) 1.04e-01  (  4,  2) 4.90e-02  (  4,  3) 7.71e-02  (  4,  4) 7.74e-01
```

Figure F11.6.52 Example of fission production matrix by hole

F11.6.38 SOURCE VECTOR BY HOLE NUMBER

This information is printed by subroutine MATRIX only if MKH=YES is specified in the parameter data, Sect. F11.4.3. The source vector by hole is the eigenvalue of the fission production matrix by hole number. The source vector should sum to 1.0. An example of the source vector by hole is shown in Fig. F11.6.53. The average self-multiplication by hole is the overall average of the self-multiplication of all the holes in the problem.

```
sample problem 18   1f27 demonstration of options problem

  source vector by hole


    hole     vector
     0    0.00000e+00
     1    3.09948e-01
     2    1.43218e-01
     3    2.13815e-01
     4    3.33020e-01

                    average self multiplication by hole

the number of next generation neutrons produced in a hole by
a neutron born in that same hole is   7.42620e-01  + or -  1.52665e-02
```

Figure F11.6.53 Example of source vector by hole number

F11.6.39 COFACTOR K-EFFECTIVE BY HOLE NUMBER

The cofactor k-effective by hole number is calculated if CKH=YES is entered in the parameter data, Sect. F11.4.3. The cofactor k-effective for a given hole is the k-effective of the system calculated without the fission source of that hole. This is done by determining the eigenvalue of the fission production matrix by hole after it has been reduced by the row and column associated with that hole. An example of the cofactor k-effective by hole number is given in Fig. F11.6.54.

```
sample problem 18    1f27 demonstration of options problem

                                                cofactor
                 hole          unit          k-effective       deviation
                  0              0           1.02174e+00      2.72224e-02
                  1              2           8.90207e-01      2.05430e-02
                  2              3           9.64636e-01      2.89249e-02
                  3              4           9.42547e-01      3.18154e-02
                  4              5           9.07257e-01      4.01007e-02

elapsed time    0.51200 minutes
```

Figure F11.6.54 Example of cofactor k-effective by hole number

F11.6.40 MATRIX K-EFFECTIVE BY ARRAY NUMBER

The matrix k-effective by array number is calculated if MKA=YES is entered in the parameter data, Sect. F11.4.3. It is the largest eigenvalue of the fission production matrix collected by array number. An example is given in Fig. F11.6.55. The number of next generation neutrons produced in array number J by a neutron born in array number I is given in this fission production matrix.

```
sample problem 18    1f27 demonstration of options problem
***************************************************************************************************************************

 array k-effective=  1.02170e+00  + or -  2.72020e-02
 the array k-effective is the largest eigenvalue of the fission production by array number matrix.

***************************************************************************************************************************
elapsed time    0.51200 minutes
```

Figure F11.6.55 Example of matrix k-effective by array number

F11.6.41 FISSION PRODUCTION BY ARRAY NUMBER MATRIX

The fission production matrix collected by array number is shown in Fig. F11.6.56. It is printed only if MKA=YES and FMA=YES are specified in the parameter data, Sect. F11.4.3.

```
fission production by array number matrix
 (  i,  j) p is the number of next generation neutrons produced in array j by a neutron born in array i.

 (  0,  0) 0.00e+00  (  0,  1) 0.00e+00  (  0,  2) 0.00e+00  (  0,  3) 0.00e+00  (  0,  4) 0.00e+00

 (  1,  0) 0.00e+00  (  1,  1) 7.75e-01  (  1,  2) 6.90e-02  (  1,  3) 9.84e-02  (  1,  4) 1.26e-01

 (  2,  0) 0.00e+00  (  2,  1) 1.29e-01  (  2,  2) 6.69e-01  (  2,  3) 9.21e-02  (  2,  4) 1.20e-01

 (  3,  0) 0.00e+00  (  3,  1) 1.09e-01  (  3,  2) 6.00e-02  (  3,  3) 6.97e-01  (  3,  4) 1.23e-01

 (  4,  0) 0.00e+00  (  4,  1) 1.04e-01  (  4,  2) 4.90e-02  (  4,  3) 7.71e-02  (  4,  4) 7.74e-01
```

Figure F11.6.56 An example of the fission production matrix by array number

F11.6.42 SOURCE VECTOR BY ARRAY NUMBER

This information is printed by subroutine MATRIX only if MKA=YES is specified in the parameter data, Sect. F11.4.3. The source vector by array number is the eigenvector of the fission production matrix by array number. The source vector should sum to 1.0. An example of the source vector by array number is shown in Fig. F11.6.57. The average self-multiplication by array number is the overall self-multiplication of all the arrays in the problem.

```
sample problem 18   1f27 demonstration of options problem

  source vector by array


   array     vector
     0     0.00000e+00
     1     3.09948e-01
     2     1.43218e-01
     3     2.13815e-01
     4     3.33020e-01

                    average self multiplication by array

the number of next generation neutrons produced in  an array by
a neutron born in that same array is   7.42620e-01  + or -  1.52665e-02
```

Figure F11.6.57 Example of source vector by array number

F11.6.43 COFACTOR K-EFFECTIVE BY ARRAY NUMBER

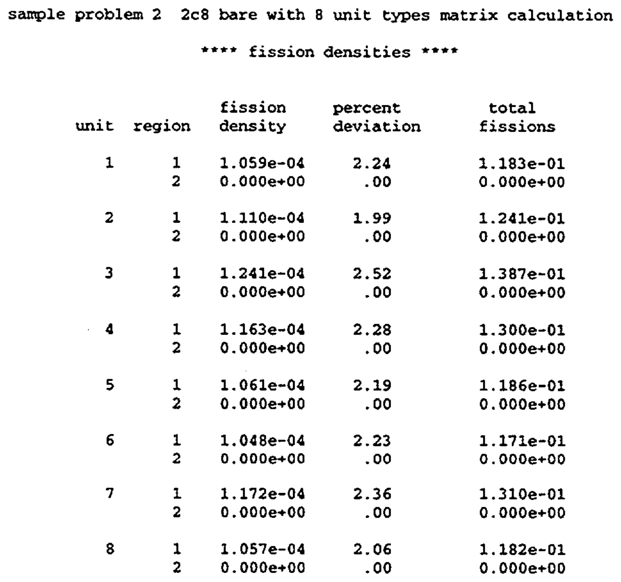
The cofactor k-effective by array number is calculated if CKA=YES is entered in the parameter data, Sect. F11.4.3. The cofactor k-effective for a given array is the k-effective of the system calculated without the fission source of that array. This is achieved by determining the eigenvector of the fission production matrix by array after reducing it by the row and column associated with the specified array. Figure F11.6.58 is an example of the cofactor k-effective by array number.

```
sample problem 18   1f27 demonstration of options problem

               array        array       cofactor
               index        number     k-effective     deviation
                 0            0        1.02174e+00     2.72224e-02
                 1            1        8.90207e-01     2.05430e-02
                 2            2        9.64636e-01     2.89249e-02
                 3            3        9.42547e-01     3.18154e-02
                 4            4        9.07257e-01     4.01007e-02

elapsed time    0.51200 minutes
```

Figure F11.6.58 Example of cofactor k-effective by array number

F11.6.44 FISSION DENSITY EDIT

The fission density edit is optional. Subroutine KEDIT prints the fission density for each geometry region if FDN=YES is specified in the parameter data. An example of the fission density edit is shown in Fig. F11.6.59.



```
              sample problem 2  2c8 bare with 8 unit types matrix calculation

                                **** fission densities ****

                                fission      percent          total
                  unit  region  density      deviation        fissions

                    1      1    1.059e-04     2.24           1.183e-01
                           2    0.000e+00      .00           0.000e+00

                    2      1    1.110e-04     1.99           1.241e-01
                           2    0.000e+00      .00           0.000e+00

                    3      1    1.241e-04     2.52           1.387e-01
                           2    0.000e+00      .00           0.000e+00

                    4      1    1.163e-04     2.28           1.300e-01
                           2    0.000e+00      .00           0.000e+00

                    5      1    1.061e-04     2.19           1.186e-01
                           2    0.000e+00      .00           0.000e+00

                    6      1    1.048e-04     2.23           1.171e-01
                           2    0.000e+00      .00           0.000e+00

                    7      1    1.172e-04     2.36           1.310e-01
                           2    0.000e+00      .00           0.000e+00

                    8      1    1.057e-04     2.06           1.182e-01
                           2    0.000e+00      .00           0.000e+00
```

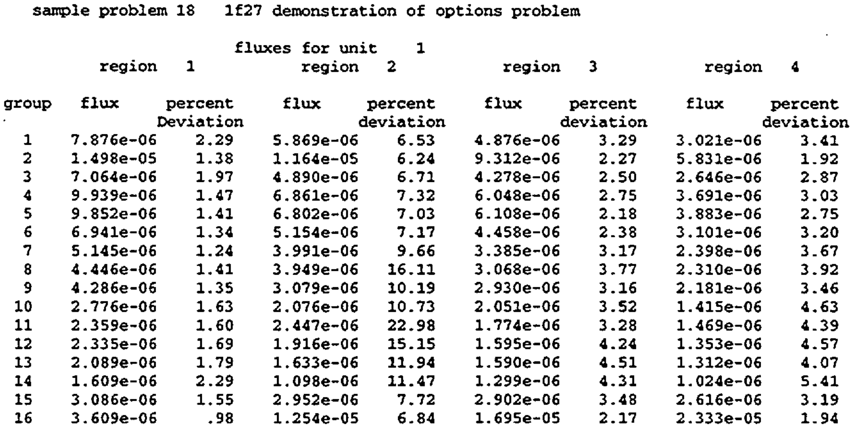
Figure F11.6.59 Example of the fission density edit

The UNIT is the unit or box-type number from the geometry data, the REGION is the region number relative to the unit, the FISSION DENSITY is the fissions/cc per source-neutron for that geometry region, the PERCENT DEVIATION is 100 times the fractional standard deviation associated with the fission density, and the TOTAL FISSIONS is the total number of fissions per source neutron in the geometry region.

F11.6.45 FLUX EDIT

Printing the fluxes is optional. They are printed by subroutine PRTFLX only if FLX=YES is specified in the parameter data. The fluxes are printed for each unit and each geometry region in the unit for every energy group. A sample of a flux edit is given in Fig. F11.6.60.

The title of the problem is printed at the top of the page. The heading FLUXES FOR UNIT ____ indicates the geometry unit for which fluxes are being printed. The region numbers relative to the unit are identified by the heading REGION _____. The geometry regions within each unit are numbered sequentially, beginning with 1. GROUP is the heading for the energy groups. The headings FLUX and PERCENT DEVIATION are printed for each geometry region in the unit. The flux and its associated percent deviation are printed for every energy group and every geometry region. The flux is in units of neutrons/cm^2/source neutron.

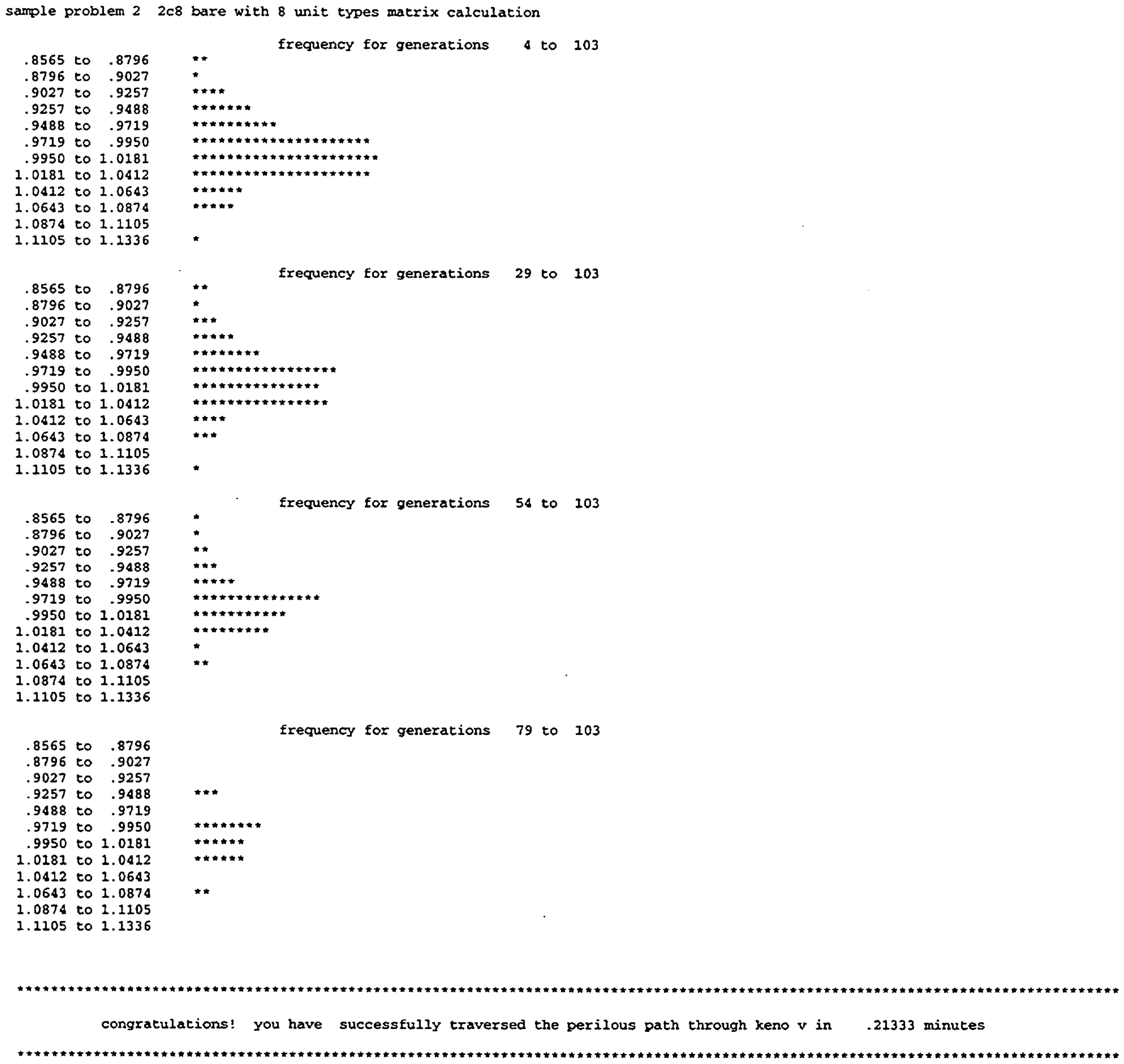


```
 sample problem 18   1f27 demonstration of options problem

                              fluxes for unit    1
         region   1               region   2               region   3               region   4

group    flux      percent      flux      percent      flux      percent      flux      percent
.                 Deviation              deviation              deviation              deviation
  1    7.876e-06    2.29     5.869e-06    6.53     4.876e-06    3.29     3.021e-06    3.41
  2    1.498e-05    1.38     1.164e-05    6.24     9.312e-06    2.27     5.831e-06    1.92
  3    7.064e-06    1.97     4.890e-06    6.71     4.278e-06    2.50     2.646e-06    2.87
  4    9.939e-06    1.47     6.861e-06    7.32     6.048e-06    2.75     3.691e-06    3.03
  5    9.852e-06    1.41     6.802e-06    7.03     6.108e-06    2.18     3.883e-06    2.75
  6    6.941e-06    1.34     5.154e-06    7.17     4.458e-06    2.38     3.101e-06    3.20
  7    5.145e-06    1.24     3.991e-06    9.66     3.385e-06    3.17     2.398e-06    3.67
  8    4.446e-06    1.41     3.949e-06   16.11     3.068e-06    3.77     2.310e-06    3.92
  9    4.286e-06    1.35     3.079e-06   10.19     2.930e-06    3.16     2.181e-06    3.46
 10    2.776e-06    1.63     2.076e-06   10.73     2.051e-06    3.52     1.415e-06    4.63
 11    2.359e-06    1.60     2.447e-06   22.98     1.774e-06    3.28     1.469e-06    4.39
 12    2.335e-06    1.69     1.916e-06   15.15     1.595e-06    4.24     1.353e-06    4.57
 13    2.089e-06    1.79     1.633e-06   11.94     1.590e-06    4.51     1.312e-06    4.07
 14    1.609e-06    2.29     1.098e-06   11.47     1.299e-06    4.31     1.024e-06    5.41
 15    3.086e-06    1.55     2.952e-06    7.72     2.902e-06    3.48     2.616e-06    3.19
 16    3.609e-06     .98     1.254e-05    6.84     1.695e-05    2.17     2.333e-05    1.94
```

Figure F11.6.60 An example of a flux edit

F11.6.46 FREQUENCY DISTRIBUTIONS

Subroutine FREAK is responsible for printing the frequency distributions, which are the last data printed for a problem. A frequency distribution consists of a bar graph indicating the number of generations having k-effective in a specified interval. The intervals are determined by the code, based on the upper and lower limits of the k-effectives calculated for the generations. One asterisk is printed for each generation k-effective. Four frequency distributions are printed as shown in Fig. F11.6.61.



```
sample problem 2  2c8 bare with 8 unit types matrix calculation
                                 frequency for generations    4 to  103
  .8565 to  .8796     **
  .8796 to  .9027     *
  .9027 to  .9257     ****
  .9257 to  .9488     *******
  .9488 to  .9719     **********
  .9719 to  .9950     *********************
  .9950 to 1.0181     **********************
 1.0181 to 1.0412     *********************
 1.0412 to 1.0643     ******
 1.0643 to 1.0874     *****
 1.0874 to 1.1105
 1.1105 to 1.1336     *

                                 frequency for generations   29 to  103
  .8565 to  .8796     **
  .8796 to  .9027     *
  .9027 to  .9257     ***
  .9257 to  .9488     *****
  .9488 to  .9719     ********
  .9719 to  .9950     *****************
  .9950 to 1.0181     ***************
 1.0181 to 1.0412     ****************
 1.0412 to 1.0643     ****
 1.0643 to 1.0874     ***
 1.0874 to 1.1105
 1.1105 to 1.1336     *

                                 frequency for generations   54 to  103
  .8565 to  .8796     *
  .8796 to  .9027     *
  .9027 to  .9257     **
  .9257 to  .9488     ***
  .9488 to  .9719     *****
  .9719 to  .9950     ***************
  .9950 to 1.0181     ***********
 1.0181 to 1.0412     *********
 1.0412 to 1.0643     *
 1.0643 to 1.0874     **
 1.0874 to 1.1105
 1.1105 to 1.1336

                                 frequency for generations   79 to  103
  .8565 to  .8796
  .8796 to  .9027
  .9027 to  .9257
  .9257 to  .9488     ***
  .9488 to  .9719
  .9719 to  .9950     ********
  .9950 to 1.0181     ******
 1.0181 to 1.0412     ******
 1.0412 to 1.0643
 1.0643 to 1.0874     **
 1.0874 to 1.1105
 1.1105 to 1.1336


*************************************************************************************************************************************************

           congratulations!  you have  successfully traversed the perilous path through keno v in    .21333 minutes

*************************************************************************************************************************************************
```

Figure F11.6.61 An example of a frequency distribution

F11.7 WARNING MESSAGES AND ERROR MESSAGES

KENO V.a prints warning and error messages that are identified by K5- followed by a unique number (i.e., K5-1 is the identifier of the first message). For additional information concerning the message, simply look up the identifier number in this section.

Warning messages appear when a possible error is encountered. If the code alters data, that fact is stated in the message. It is the responsibility of the user to verify correct usage whenever a warning message is printed.

When an error is encountered, the error flag MFLAG is set true and an error message is printed. The code stops if the error is too severe to continue. The warning and error messages in this section may show an underscore ____ or a numbered underscore (1) where data will be printed by the code. The explanation of the message will show an underscore or a numbered underscore to indicate the corresponding data.

F11.7.1 MESSAGES

K5-1 ***** WARNING ***** READ FLAG NOT FOUND. ASSUME PARAMETER DATA FOLLOWS.

This message occurs in subroutine INITAL. It indicates that the word READ was not the first word of data encountered after the title card. If a parameter data block is to be entered, the code expects the words READ PARAMETERS to precede the parameter input data. If the word READ is not the first word, the code expects parameter input data immediately.

K5-2 *** ERROR *** THE NUMBER OF ENERGY GROUPS IS OUT OF RANGE FOR THE CROSS SECTION LIBRARY ON UNIT ____. THE JOB CONTROL LANGUAGE MAY NOT SPECIFY A VALID DATA SET ON THIS UNIT OR THE MODULE THAT WAS TO CREATE THE CROSS SECTION LIBRARY ON THIS UNIT MAY HAVE FAILED.

This message occurs in subroutine INITAL after subroutine PARAM has been executed. Check unit number ____ to see that it was properly specified in the job control language. Verify that the data set name associated with this unit number is the correct one. This information is given in the printout in the third table. Make sure the module that generated the cross sections executed properly and that the data were saved or passed correctly. When this message is printed for an AMPX working format library, a STOP 108 is executed. When this message is printed for a mixed cross-section format library, a STOP 109 is executed.

K5-3 *** WARNING *** THE CROSS SECTION LIBRARY ON UNIT ____ HAS ____ ENERGY GROUPS.

This message is printed in subroutine INITAL after the call to PARAM. It is activated if the cross-section library has more than 300 energy groups. The largest standard cross-section library in the SCALE system contains 227 energy groups.

K5-4 INVALID INPUT PARAMETER NAME ____

This message comes from subroutine PARAM. The keyword for entering parameter data was misspelled. A list of allowed keywords is given in Table F11.4.1 in the KENO V.a input outline.

K5-5 ***** AN ERROR WAS ENCOUNTERED IN THE ALPHANUMERIC PARAMETER DATA. THE DATA WERE ____ ____ *****

This message comes from subroutine PARAM. The keyword for the alphanumeric parameter data was entered correctly, but the data associated with it were not YES or NO as is required. The ____ ____in the error message could be something like FLX=YEX instead of FLX=YES.

K5-6 ***** WARNING ***** READ FLAG FOUND WHEN LOOKING FOR END FLAG. PARAMETER INPUT ASSUMED COMPLETE

This message occurs in subroutine PARAM. It indicates that the keywords END PARAMETERS were not found. The keywords READ ____ were found instead. The code assumes the parameter data are complete and proceed normally.

K5-7 ATTEMPT TO FIND END PARAMETER FLAG WAS UNSUCCESSFUL

This message from subroutine PARAM occurs during the reading of the parameter data if the word END is found and it is not followed by the word PARAMETERS. A STOP 118 may be executed when this message is printed.

K5-8 ***** AN END OF FILE WAS ENCOUNTERED WHILE ATTEMPTING TO READ PARAMETER DATA *****

This self-explanatory message is from subroutine PARAM. A STOP 118 may be executed when this message is printed.

K5-9 *** DUE TO INCONSISTENCIES BETWEEN INPUT AND RESTART DATA, FISSIONS AND ABSORPTIONS BY REGION WILL BE CALCULATED BUT NOT PRINTED. INPUT DATA SET FAR=NO, BUT DATA FROM THE RESTART UNIT SPECIFIED YES.

This message occurs in subroutine PARAM. It is mostly self-explanatory. The original problem (parent case) that wrote the restart data specified data inconsistent with the parameter data input to the restarted problem. The title of the parent case is given at the end of the parameter tables. The specification of the restart unit RST is given in the third table of the KENO V.a output.

K5-10 *** DUE TO INCONSISTENCIES BETWEEN INPUT AND RESTART DATA, FLUXES WILL BE CALCULATED BUT NOT PRINTED. INPUT DATA SET FLX=NO, BUT DATA FROM THE RESTART UNIT SPECIFIED YES.

This message occurs in subroutine PARAM. It is mostly self-explanatory. The original problem (parent case) that wrote the restart data specified data that did not agree with the parameter data input to the restarted problem. The title of the parent case is given at the end of the parameter tables. The specification of the restart unit RST is given in the third table of the KENO V.a output.

K5-12 INPUT PARAMETER NBK WAS ENTERED AS ____. IT WAS CHANGED TO ____. AT LEAST ____ POSITIONS ARE NECESSARY TO ACCOMMODATE THE NEUTRON BANK DATA.

This self-explanatory message is from subroutine PARAM. NBK should not be entered as input data unless it is known that the default value is inadequate.

K5-14 ***** ERROR - KEYWORD ____ ____ IS NOT A VALID MIXING TABLE KEYWORD. *****

This message is from subroutine MIXIT. It can only be encountered if a mixing table is expected (i.e., READ MIX or READ MIXT has been entered as data). At this point the only valid keywords are MIX=, EPS= or SCT=. The keyword that was entered is printed in the message. See Sect. F11.4.10 for assistance in setting up the mixing table data. A common error made by KENO IV users is to enter a negative nuclide ID number in the mixing table. The code will interpret this to be a keyword. A whole list of K5-14 error messages will be generated as the code reads through the mixing table in search of a valid keyword. The code considers any character string that does not begin with a number to be a keyword.

K5-15 MIXING TABLE TOO BIG

This message is from subroutine MIXIT. It indicates that additional core space is necessary to allow entry of the existing mixing table. A STOP 114 is executed in conjunction with this message and a traceback may be printed from subroutine STOP.

K5-16 *** ERROR *** ERROR *** A VALUE MUST BE ENTERED FOR LIB IN THE PARAMETER INPUT SO CROSS SECTIONS CAN BE MIXED.

This message is from subroutine DATAIN. It occurs when a mixing table has been read but the unit number for the AMPXS working library is undefined. This is corrected by entering LIB=____ in the parameter input data and making sure the desired AMPXS working library is properly defined as being on that unit in the job control language.

K5-17 UNRECOGNIZABLE GEOMETRY WORD ____

This message is from subroutine KENOG. In the process of reading the geometry data, the word __ ___ was encountered when a geometry word was expected. Several of these messages may be generated. A message is generated for each word of data that is read, until a valid geometry word is found. The data are out of phase or the geometry word is misspelled. Check the previous geometry card for a mixture ID, a bias ID, and the proper number of dimensions. See Sect. F11.4.4 for a list of accepted geometry words.

K5-18 ********** ERROR.....NHCYL=_____ **********

This message from subroutine KENOG indicates that the hemicylinder geometry word was incorrectly specified. See Sect. F11.4.4 for the correct hemicylinder specification.

K5-19 AN ERROR WAS FOUND IN THE HEMISPHERE DESIGNATION

This message from subroutine KENOG indicates that the direction in which the hemisphere exists was incorrectly specified. See Sect. F11.4.4 to determine the correct specification.

K5-20 ********** IGEO=_____ INVALID IN READGM **********

This message from subroutine READGM means that IGEO is negative or larger than 23 when the geometry data are read from the scratch unit, SKRT. This usually means that a code error was introduced when changes were made to the program. True geometry errors should be detected when the scratch unit, SKRT, is written. A STOP 125 is executed when this message is printed.

K5-21 _____ IS AN INVALID PARAMETER NAME FOR BIASING DATA. ID= OR WT= OR WTS= SHOULD HAVE BEEN ENTERED.

This self-explanatory message is from subroutine RDBIAS. See Sect. F11.4.7 for assistance in determining the proper procedure for entering biasing data.

K5-22 FIRST TWO NUMBERS ARE WORDS OF STORAGE NEEDED AND ALLOCATED. THIRD IS REQUIRED ADDITIONAL REGION SIZE.

PERTINENT CONSTANTS

_____ _____ _____

This message from subroutine WATES indicates that the allocated computer storage will not hold the weighting or biasing array. The first number printed is the amount of storage, in words, needed to hold the data. The second number is the allocated computer storage in words. The third number is the minimum additional region size, in units of K bytes necessary to hold the biasing or weighting data to this point. Increase the region size for the "go step" in the job control language by the additional required region size (the third number) and resubmit the problem. A STOP 150 is executed when this message is printed.

K5-23 *** ERROR *** ERROR *** NO _____ ENERGY GROUP WEIGHTS WERE FOUND FOR ID _____ *** ERROR *** ERROR ***

This message from subroutine WATES occurs if the weights requested in the biasing information were not on the standard weights data set and were not entered from cards. See Table F11.4.5 for the weights that are available on the standard data set. The procedure for entering weights from cards is explained in Sect. F11.4.7.

K5-24 INCORRECT FLAG RETURNED FROM AREAD. IRET=____

This message from subroutine RDBIAS indicates that an error was encountered while reading the biasing data. The biasing data were not entered properly. See Sect. F11.4.7 for assistance.

K5-25 THE FIRST NUMBER IS THE AMOUNT OF STORAGE NEEDED; THE SECOND IS THE AMOUNT ALLOCATED. PERTINENT CONSTANTS ____ ____

This message is from subroutine ARAYIN or subroutine GEOMIN. It indicates that additional core space is required to allow the use of the array definition data or the geometry region data. At least _____ words of storage are needed to run the problem but only _____words of storage are available. Increase the amount of computer storage requested in the job control language to correspond to the amount needed. A STOP 100 or 155 is executed in conjunction with this message and a traceback may be printed from subroutine STOP.

K5-26 SET NUMBER _____ OF THE UNIT ORIENTATION DATA CONTAINS _____ERROR(S)

This message from subroutine RDBOX is triggered when input errors are recognized in the unit orientation data. A set of unit orientation data consists of 10 numbers as shown in the companion message K5-27. The number of errors printed in this message is a lower bound. More errors may actually exist. This message often means that a number was omitted or a blank was omitted when entering the unit orientation data.

K5-27 LTYPE=_____ IX1=_____ IX2=_____ INCX=_____ IY1=_____ IY2=_____
INCY=_____ IZ1=_____ IZ2=_____ INCZ=_____

This message is a companion message for K5-26. It indicates how the unit orientation data description for the set named in K5-26 was entered. See Sect. F11.4.5 for information pertaining to unit orientation data.

K5-28 THE ABOVE UNIT ORIENTATION CARD(S) CONTAIN(S) AT LEAST ONE OF THE FOLLOWING ERRORS

1. IX1,IY1,IZ1,INCX,INCY, OR INCZ IS LESS THAN OR EQUAL TO ZERO

2. IX2 IS LESS THAN IX1, IY2 IS LESS THAN IY1, OR IZ2 IS LESS THAN IZ1

3. IX2 IS GREATER THAN NBXMAX, IY2 IS LARGER THAN NBYMAX OR IZ2 IS LARGER THAN NBZMAX

4. LTYPE IS LESS THAN 1 OR GREATER THAN NBOX

This self-explanatory message is from subroutine RDBOX. It pertains to the input orientation data for LOOP. See Sect. F11.4.5 for input instructions.

K5-29 *** ERROR *** THE ARRAY SIZE HAS BEEN SPECIFIED INCORRECTLY FOR ARRAY _____.
NBXMAX=_____ NBYMAX=_____ NBZMAX=_____
UNIT ORIENTATION DATA CANNOT BE READ UNLESS NBXMAX, NBYMAX, AND NBZMAX ARE GREATER THAN ZERO.

This message from subroutine ARAYIN indicates that the array definition data were incorrectly specified. It occurs only if one or more of NBXMAX, NBYMAX, or NBZMAX is less than 1. In the array information data these are entered in the form NUX=_____ NUY=_____ NUZ=_____. See Sect. F11.4.5. If a unit orientation data description is to be entered, NBXMAX, NBYMAX, and NBZMAX must all be greater than zero.

K5-30 END _____ FLAG WAS NOT FOUND. _____ _____ WAS READ INSTEAD.

This message from subroutine ARAYIN occurs if the unit orientation data description is terminated with the incorrect END flag.

K5-31 _____ IS AN INVALID PARAMETER NAME IN THE ARRAY DATA.

This message is written from subroutine ARAYIN if the array data block contains an incorrect keyword. The allowed keywords include NUX= NUY= NUZ= FILL and LOOP. See Sect. F11.4.5 for additional assistance. A STOP 101 is executed when this message is printed.

K5-32 *** AN ERROR EXISTS IN UNIT ORIENTATION ARRAY NUMBER _____ ***

This message from subroutine SORTA is printed when an error is recognized in the array description. The type of error that will trigger the message is for a position in the unit orientation array to be undefined, zero, negative or greater than NBOX, the number of input units. K5-33 is a companion message.

K5-33 UNIT _____ IS INVALID AT X INDEX= _____ Y INDEX= _____ Z INDEX= _____

This message comes from subroutine SORTA. It is printed for each position in the unit orientation array that is in error. The message is printed a maximum of 10 times. Refer to Sect. F11.4.5 for assistance in correcting the error(s).

K5-34 ***** ERROR ***** THE NUMBER OF MIXTURES REQUESTED IN THE GEOMETRY IS ___ THE NUMBER OF MIXTURE CROSS SECTIONS IS ___.

This message from subroutine FLDATA occurs if the number of mixture cross sections from the restart unit, RSTRT, does not equal the number of mixtures requested in the geometry for a restarted problem.

K5-35 ***** ERROR ***** IN THE ALBEDO INPUT DATA _____ IS AN INVALID FACE CODE NAME.

This message is from subroutine RDREF. It occurs if an invalid face code name was entered in the albedo data. See Table F11.4.3 in Sect. F11.4.6 for a list of acceptable face code names.

K5-36 A PERIODIC BOUNDARY CONDITION WAS SPECIFIED WITH A NON-COMPATIBLE BOUNDARY CONDITION ON THE OPPOSING FACE. THE PROBLEM WILL NOT BE RUN.

This self-explanatory message is from subroutine RDREF. If a periodic boundary condition is specified on one x face, it must also be specified on the other x face, etc.

K5-37 ***** ERROR AVERAGE NU-BAR AND AVG. FISSION GROUP WAS SPECIFIED, BUT THE FISSION XSEC ID (18) WAS NOT FOUND IN THE EXTRA 1-D ARRAY (MT).

This message is from subroutine IDX1D. It indicates that the parameter data contained NUB=YES but the corresponding necessary type of data was absent from the extra 1-D array. This can be due to a code error or an error concerning the extra 1-D data (X1D= in the parameter data).

K5-38 INPUT DATA INDICATED NO EXTRA 1-D XSEC IDS TO BE READ, BUT A READ FLAG WAS ENCOUNTERED.

This message from subroutine DATAIN is printed when the parameter data did not specify X1D= and the words READ X1DS were encountered later in the data. If extra 1-D data are to be used, X1D= must be entered in the parameter data and appropriate code modifications must be made to properly utilize the extra 1-D data.

K5-39 INVALID START PARAMETER NAME _____

This message is from subroutine RDSTRT. It indicates that an invalid start parameter name was encountered when the start data block was being read. A list of allowed start parameter names is contained in Sect. F11.4.8.

K5-40 LNU FOR START TYPE 6 WAS ENTERED AS _____. IT WAS CHANGED TO _____ WHICH IS THE VALUE ENTERED FOR NFB.
THE LARGEST VALUE NEEDED FOR LNU IS NPG. THE LARGEST VALUE ALLOWED FOR LNU IS NFB. BOTH NPG AND NFB ARE PARAMETER DATA.

This self-explanatory message is from subroutine RDSTRT. See Sect. F11.4.8 for assistance in determining a valid value for LNU. NFB, the fission bank size, is the largest value allowed for LNU. NPG, the number of histories per generation, is the smallest value allowed for LNU.

K5-41 *** ERROR *** ALPHANUMERIC START DATA MUST BE ENTERED AS YES OR NO. THE DATA READ WERE _____ _____.

This self-explanatory message is from subroutine RDSTRT. See Sect. F11.4.8 for assistance concerning start data.

K5-42 END _____ FLAG WAS NOT FOUND. _____ _____ WAS READ INSTEAD.

This message is from subroutine DATAIN. It occurs when the READ _____ and END _____do not match. When entering data blocks, each block must start with READ _____ and end with END _____.

K5-43 ***** AN END OF FILE WAS ENCOUNTERED BEFORE AN END DATA WAS FOUND. THE PROBLEM WILL NOT RUN. *****

This message is from subroutine DATAIN. It occurs when an end of file is encountered while reading data.

K5-45 ***** ILLEGAL DATA BLOCK IDENTIFIER ____ ____ *****

This message from subroutine DATAIN is printed whenever an invalid data block identifier is encountered. This can be caused by having the data out of order, by omitting data or by misspelling data. A block identifier consists of the words READ XXXX where XXXX is a keyword identifying the type of data to be read. Acceptable keywords are listed in Table F11.4.1, in Sect. F11.4.1.

Consider the following examples:

ERROR MESSAGE EXAMPLE 1

```
READ PARAM TME=2.9 FLX=YES XSC=38 END PARAM
READ GOEM  CYLINDER  1 1 5.0 5.0 -5.0 END GEOM
END DATA
END
```

The following message would occur:

***** ILLEGAL DATA BLOCK IDENTIFIER READ GOEM *****

The keyword GEOM was misspelled as GOEM. The correct data block identifier is READ GEOM.

ERROR MESSAGE EXAMPLE 2

```
READ PARAM  TME=2.9 FLX=YES XSC=38 END PARAM
END GEOM
CYLINDER  1 1 5.0 5.0 -5.0  END GEOM
END DATA
END
```

The following message would occur:

***** ILLEGAL DATA BLOCK CYLINDER *****

The words END GEOM are read and recognized as the end of a data block. The words CYLI and NDER are then read as the data block identifier. Since CYLI is not END, the code assumes it is at the beginning of a data block that is identified by the keyword NDER, which is not one of the acceptable keywords.

ERROR MESSAGE EXAMPLE 3

```
READ PARAM  TME=2.9 FLX=YES XSC=38 END PARAM
END GEOM
READ GEOM
CYLINDER  1 1 5.0 5.0 -5.0
END GEOM
END DATA
END
```

In this example, no errors will be found and the problem will run correctly. The END PARAM signals the end of the parameter data block. The first END GEOM signals the end of a geometry block, the READ GEOM signals the beginning of a geometry block, the second END GEOM signals the end of a geometry block, and the END DATA signals the end of the problem.

K5-46 ***** IPT= _____ IS OUTSIDE THE ALLOWABLE LIMIT OF _____

This self-explanatory message is from subroutine DATAIN. It is indicative of a code error. IPT is the index into the LPOINT array, which contains the direct access pointers for the various types of data.

LPOINT(1) is the pointer for the geometry region data.
LPOINT(2) is the pointer for the array description (unit orientation) data.
LPOINT(3) is the pointer for the mixing table data.
LPOINT(4) is the pointer for extra data.
LPOINT(5) is the pointer for the biasing or weighting data.
LPOINT(6) is the pointer for the start data.
LPOINT(7) is the pointer for the albedo data.
LPOINT(8) is the pointer for the mixed cross-section data.
LPOINT(9) is the pointer for the energy and inverse velocity data.
LPOINT(10) is the pointer for the plot data.
LPOINT(11) is the pointer for the biasing input data.
LPOINT(12) is the pointer for albedo-xsec energy correspondence.

K5-48 *** ERROR *** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO READ RESTART DATA FROM UNIT _____ NDX=_____ NREC=_____

This message is from subroutine RDRST. It indicates that the restart data associated with the index NDX had the wrong number of records. This message can also be caused by a code error introduced as the result of making changes in the code.

NDX=1 is the geometry data. NREC should be 3.
NDX=2 is the array description data. NREC should be 3 or more.
NDX=3 is the mixing table data. NREC should be 3.
NDX=4 is the extra data. NREC must be set by the user.
NDX=5 is the biasing or weight data. NREC should be 3.
NDX=6 is the start data. NREC should be 3.

NDX=7 is the albedo data. NREC should be at least 2.
NDX=8 is the mixed cross-section data.
NDX=9 is the energy and inverse velocity data.
NDX=10 is the plot data.
NDX=11 is the biasing input data.
NDX=12 is the albedo-xsec energy correspondence.
A STOP 123 is executed when this message is printed.

K5-49 ____ WORDS ARE NEEDED TO HOLD THE ALBEDO DATA. BUT ONLY ____ WORDS ARE AVAILABLE.

This message from subroutine RDALB is self-explanatory. More space is needed to contain the albedo data. A STOP 120 is executed when this message is printed.

K5-50 THE SPACE AVAILABLE IN SUBROUTINE RDICE IS ____. MORE IS NEEDED.

This self-explanatory message from subroutine RDICE indicates that more space is needed to store the macroscopic cross sections in ICE format. A STOP 122 is executed when this message is printed.

K5-51 ***** ERROR ***** THE ____ FACE REQUESTED A DIFFERENTIAL ALBEDO THAT IS NOT ON THE ALBEDO DATA SET. THE REQUESTED ALBEDO NAME IS ____ ____.

This self-explanatory message is from subroutine ALBRD. A list of the albedos that are on the standard albedo data set is given in Table F11.4.4 in Sect. F11.4.6.

K5-52 MIXTURE NUMBER TOO LARGE

This self-explanatory message is from subroutine MIXCRS. A STOP 111 is executed in conjunction with this message, and a traceback may be printed from subroutine STOP.

K5-53 NOT ENOUGH STORAGE TO MIX

This message from subroutine MIXER indicates that more storage is necessary in order to do the cross-section mixing operations. A STOP 112 is executed in conjunction with this message, and a traceback may be printed from subroutine STOP.

K5-54 NOT ENOUGH STORAGE TO MAKE ANGLES

This message from subroutine MIXER indicates that more storage is necessary to perform the calculations required to make the angles during the cross-section mixing operations. A STOP 113 is executed in conjunction with this message, and a traceback may be printed from subroutine STOP.

K5-55 **** ERROR FOUND IN MIXING CROSS SECTIONS ****
THE FOLLOWING NUCLIDE(S) SPECIFIED IN THE MIXING TABLE WERE NOT FOUND ON THE CROSS SECTION LIBRARY.

ENTRY	NUCLIDE ID
.	.
.	.
.	.

This self-explanatory message is from subroutine MIXMIX. Either a nuclide ID was misspelled or was not in the cross-section library.

K5-56 ERROR - MIXTURE _____ LACKS EITHER NU*FISSION OR CHI DATA.

This message from subroutine NORM1D indicates that a mixture that contains fissile material is missing the nu-fission cross section or the fission spectrum. A STOP 115 is executed when this message is printed.

K5-57 NOT ENOUGH STORAGE TO MAKE ANGLES AND PROBS

This self-explanatory message from subroutine MAKANG indicates that more storage is needed to complete the cross-section mixing operations. Resubmit the problem and request more storage space in the job control language. A STOP 110 is executed in conjunction with this message and a traceback may be printed from subroutine STOP.

K5-58 This message appears in two forms, listed below:

K5-58 ** ERROR **** ERROR ** INVALID BIAS ID _____ IN REGION _____
OF UNIT _____.

This form of message K5-58 is from subroutine READGM. It is printed if a negative or zero bias ID is encountered anywhere in the geometry data. The error flag (MFLAG) is not activated, so the problem will execute if the unit containing the error is not utilized in the unit orientation array. If that unit is used in the unit orientation array, the following form of message K5-58 will also be printed.

K5-58 ** ERROR **** ERROR ** INVALID BIAS ID IN REGION ABOVE.

This form of message K5-58 is from subroutine PRTJOM. It is printed if a negative or zero bias ID is entered for the specified geometry region. The problem will not execute if this form of message K5-58 is printed. Review Sect. F11.4.4 for correct geometry data specification information.

K5-59 THE CALCULATION WAS TERMINATED BECAUSE OF EXCESSIVE SPLITTING

This message from subroutine GUIDE is printed only if message K5-128 is printed 50 or more times for a given generation. This indicates that the problem and/or the code is incapable of achieving a reasonable solution. If changes have been made in the code, they should be carefully scrutinized. If a biasing data block has been entered (Sect. F11.4.7), it should be checked carefully.

K5-60 THE ANGULAR SCATTERING DISTRIBUTION FOR MIXTURE ____ HAS BAD MOMENTS
FOR THE TRANSFER FROM GROUP ____ TO GROUP ____.
____ MOMENTS WERE ACCEPTED.

THE LEGENDRE EXPANSION OF THE CROSS SECTION (P0-PN) IS
(P_0) (P_1) (P_2) ... (P_n)
__ __ __ __
THE MOMENTS CORRESPONDING TO THIS DISTRIBUTION ARE
(M_1) (M_2) ... (M_n)
__ __ __
THE MOMENTS CORRESPONDING TO THE GENERATED DISTRIBUTION ARE
(M_1) (M_2) ... (M_n)
__ __ __
THE LEGENDRE EXPANSION CORRESPONDING TO THESE MOMENTS IS
(P_0) (P_1) (P_2) ... (P_n)
__ __ __ __
THE WEIGHTS/ANGLES FOR THIS DISTRIBUTION ARE
(W_1) (W_2) ... (Wm)
__ __ __
(A_1) (A_2) ... (Am)
__ __ __
THE MOMENTS CORRESPONDING TO THIS DISTRIBUTION ARE
(M_1) (M_2) ... (M_n)
__ __ __

This message from subroutine BADMOM indicates that the moments from the cross-section data are incorrect for the group transfer shown. The code replaces the incorrect moments with acceptable moments and proceeds normally. The moments printed in the last line of the message should match those printed in the eighth line. The user can suppress these messages by entering an appropriate value for the "cross-section message cutoff parameter," EPS= in the mixing table data, Sect. F11.4.10. See Sect. F11.5.4.4 for assistance in determining an appropriate value.

K5-62 POSIT ERROR IN TRACK — ILLEGAL GEOMETRY TYPE X=____ Y=____
Z=____ K1=____ K2=____ K=____ IGEO=____

This message from subroutine TRACK is usually the result of a code error that was introduced when changes were made to the code. A STOP 134 is printed in conjunction with this message.

K5-63 NOT ENOUGH I/Os ARE LEFT TO PRINT THE FLUXES.

This message from subroutine FITFLX is self-explanatory. The number of I/Os specified in the job control language must be increased and the problem must be rerun if printed fluxes are required. All other results are unaffected by the failure to print the fluxes.

K5-64 THE ANGULAR SCATTERING DISTRIBUTION FOR MIXTURE ____ HAS BAD MOMENTS FOR THE TRANSFER FROM GROUP ____ TO GROUP ____. THE P0 COEFFICIENT IS ____.

This message from subroutine BADMOM is printed to inform the user that the cross sections were altered by the code because the moments from the cross-section data were incorrect for the group transfer shown. The P_0 coefficient was larger than the "cross-section message cutoff parameter," EPS, but the relative change in the moments was smaller than EPS.

K5-65 AN AMPX WORKING LIBRARY WAS SPECIFIED ON UNIT ____ BUT NO MIXING DATA WAS READ.

This message from subroutine ICEMIX occurs if the parameter data specified LIB=____ but no cross-section mixing data block was entered. The cross-section mixing data block begins with READ MIXT. See Sect. F11.4.3 for parameter data and Sect. F11.4.10 for mixing table information.

K5-66 ____ MIXTURES WERE REQUESTED IN THE GEOMETRY, BUT ONLY ____ MIXTURES ARE ON THE MIXED CROSS SECTION LIBRARY.

This message from subroutine ICEMIX indicates that more mixtures were requested in the geometry region data than were available on the mixed cross-section library. See Sect. F11.4.3 for the specification of the unit number of the mixed cross-section library (XSC=), Sect. F11.4.10 for the specification of the mixing table, and Sect. F11.4.4 to determine the mixtures used in the geometry region data.

K5-67 THE ADJOINT INPUT PARAMETER WAS ____ BUT THE ADJOINT INDICATOR FROM THE MIXTURE CROSS SECTION LIBRARY WAS ____. KENO V.a WILL NOT EXECUTE.

This message from subroutine ICEMIX occurs if the adjoint input parameter ADJ= specified a forward calculation and the cross sections were adjointed or the adjoint input parameter specified an adjoint calculation and the cross sections were not adjointed. T indicates true, F indicates false.

K5-68 THE MIXED CROSS SECTION LIBRARY ON UNIT ____ IS COMPLETELY COUPLED. KENO V.a WILL NOT BE EXECUTED.

This message is from subroutine ICEMIX. It indicates that the mixed cross-section library is a completely coupled neutron-gamma library and therefore cannot be used for a KENO V.a calculation. A STOP 106 is executed when this message is printed.

K5-69 THE AMOUNT OF REMAINING SPACE IS INSUFFICIENT TO CONTAIN THE NECESSARY DATA. LIMIT=____.

This message from subroutine RDTAPE indicates that more storage is needed to allow processing of the premixed ice format cross-section data. A STOP 124 is executed when this message is printed.

K5-70 ERROR IN SUBROUTINE WRTRST. NDX=____

This message from subroutine WRTRST occurs only if the type of data to be written on the restart (WSTRT) unit is undefined (i.e., NDX is less than 1 or greater than 12). NDX is the index in the LPOINT array as described in messages K5-46 and K5-48. This error is usually caused by code errors that were introduced when changes were made to the code. A STOP 133 is executed when this message is printed.

K5-71 INSUFFICIENT SPACE ALLOWED IN SUBROUTINE ALBWRT. ____ WORDS ARE ALLOWED, BUT ____ WORDS WERE REQUESTED.

This message from subroutine WRTALB indicates that more core storage is needed to allow loading the albedo data from the direct access device into core. A STOP 105 is executed when this message is printed. Resubmit the problem, requesting more storage space in the job control language.

K5-72 ARRAY IS TOO LARGE TO FIT IN SPACE ALLOTTED IN SUBROUTINE WRTICE.

This message is from subroutine WRTICE. It indicates that more storage is needed to allow loading the mixed cross-section data from the direct access device into core. A STOP 132 is executed when this message is printed. Resubmit the problem, requesting more storage space in the job control language.

K5-73 ____ MIXTURES WERE REQUESTED IN THE GEOMETRY DATA, BUT ONLY ____ OF THESE WERE FOUND IN THE MIXED CROSS SECTIONS.

This self-explanatory message is from subroutine MASTER. Either the wrong mixed cross-section data set (XSC= from the parameter data, Sect. F11.4.3) is being used, or one or more mixture numbers are in error in the geometry region data (see Sect. F11.4.4).

K5-74 ***** ____ WORDS OF STORAGE WERE ALLOCATED TO RUN THIS PROBLEM, BUT_ ___ WORDS ARE REQUESTED FOR THE INPUT DATA. *****

This self-explanatory message is from subroutine MASTER. Additional storage is required to run this problem. Alter the job control language to request sufficient storage.

K5-75 *** NOT ENOUGH STORAGE TO PRINT ONE DIMENSIONAL CROSS SECTIONS.____ LOCATIONS ARE NEEDED, BUT ONLY ____ ARE AVAILABLE

This self-explanatory message is from subroutine PRTXS. Additional storage must be requested in the job control language in order to print the 1-D mixture cross sections.

K5-76 *** NOT ENOUGH STORAGE TO PRINT TWO DIMENSIONAL CROSS SECTIONS. ARE NEEDED, BUT ONLY ____ ARE AVAILABLE.

This self-explanatory message is from subroutine PRTXS. Additional storage is needed to print the 2-D mixture cross sections. More storage must be requested in the job control language.

K5-77 *** NOT ENOUGH STORAGE TO PRINT ANGLES AND PROBABILITIES. _____ LOCATIONS ARE NEEDED, BUT ONLY _____ ARE AVAILABLE.

This self-explanatory message is from subroutine PRTXS. Additional storage must be requested in the job control language to print the angles and probabilities for the mixture cross sections.

K5-78 ***** WARNING ***** MORE SPACE MAY BE NEEDED THAN IS INDICATED.

This message is from subroutine NSUPG. In the process of supergrouping the energy-dependent information, the code ran out of available storage space. An attempt has been made to estimate the amount of storage needed, but the estimate may be low. Resubmit the job and request sufficient storage in the job control language.

K5-79 THIS PROBLEM IS TOO LARGE FOR THE AMOUNT OF CORE THAT WAS ALLOCATED.

This message is from subroutine NSUPG. It occurs during the process of determining the number of supergroups, if the amount of storage is found to be insufficient to contain the energy-dependent information associated with one of the energy groups. Change the job control language to allow more storage. A STOP 116 is executed in conjunction with this message and a traceback may be printed from subroutine STOP.

K5-80 _____ WORDS OF STORAGE WERE ALLOCATED, BUT AT LEAST _____ ADDITIONAL WORDS ARE NEEDED TO HOLD THE INPUT DATA. EVEN MORE SPACE WILL BE NECESSARY TO RUN THE PROBLEM.

This message from subroutine NSUPG is printed if the available storage is too small to hold the input data. Change the job control language to allow more storage.

K5-81 THE AVAILABLE SPACE IS TOO SMALL TO CONTAIN THE DATA OF THE LARGEST ENERGY GROUP. MAXL=_____ LFTLNG=_____ LTOT=_____ LTOTAL=_____ NSG=_____

This message is from subroutine NSUPG.

MAXL is the size of the energy-dependent data associated with the largest energy group.
LFTLNG is the amount of space available to the super group.
LTOT is the total length of the cross-section data.
LTOTAL is the total length of the albedo data.
NSG is the supergroup being processed.

A STOP 117 is executed when this message is printed. Increase the allocated storage in the "go step" of the job control language and resubmit the problem.

K5-82 TOO MANY STORAGE LOCATIONS REQUIRED. _______ WORDS OF STORAGE ARE NEEDED, BUT ONLY _____ ARE AVAILABLE.

This self-explanatory message is from subroutine POINT. To run the problem, the job control language must be changed to increase the amount of storage to be consistent with that specified in the message.

K5-83 ***** FOR ARRAY ____ THE ____ DIMENSIONS OF UNIT ____ AT (____, ____, ____)
DO NOT MATCH THOSE OF UNIT ____ AT (____, ____, ____)*****
FOR UNIT _____+_____=_____ AND -_____=_____ WHILE FOR UNIT _____+ _____=
_____ AND -_____=_____.

This message is from subroutine ARASIZ. The common faces of adjacent units or box types must be the same size and shape. This message occurs whenever this requirement is not met. One or more of the dimensions of the units or box types specified in the message may be incorrect, or the array definition data may be incorrect. Carefully check the input data relating to the geometry region data and the array definition data as described in Sects. F11.4.4 and F11.4.5.

K5-84 ********** UNIT _____ IN UNIT ORIENTATION ARRAY NUMBER _____ IS UNDEFINED IN THE INPUT DATA **********

This message from subroutine ARASIZ occurs when the array description data block specifies a unit or box type that was not defined in the geometry region data. Verify the array definition data and the geometry region data as described in Sects. F11.4.4 and F11.4.5.

K5-85 UNIT _____ IS INVALID AT POSITION X=_____ Y=_____ Z=_____ IN UNIT ORIENTATION ARRAY NUMBER _____.

This message from subroutine ARASIZ occurs if the unit number named in the message is less than or equal to zero or greater than NBOX (the number of different box types). The position of the offending unit is also given. This error usually results from leaving some positions undefined in the unit orientation array or from erroneous data in the unit orientation data. (This includes extra data, mispunched data and omitted data.) See Sect. F11.4.5 for additional information.

K5-86 UNRECOGNIZED GEOMETRY TYPE. IGEOM=_____ IN SUBROUTINE PRTJOM.

This message from subroutine PRTJOM is self-explanatory. The printed value of IGEOM must be greater than zero and less than 24 to be valid. If the geometry words (see *fgeom* in Sect. F11.4.4) are correct, this message is due to a code error that has been introduced when changes were made to the code.

K5-87 ERROR IN HEMISPHERE DESIGNATION. ISET=_____ IN SUBROUTINE PRTJOM.

This message from subroutine PRTJOM is self-explanatory. ISET should be greater than zero and less than 7. It defines the orientation of a hemisphere. A code error is the likely cause of this message.

K5-88 ERROR IN HEMICYLINDER DESIGNATION. NHCYL=____ IN SUBROUTINE PRTJOM

This message from subroutine PRTJOM can occur if the hemicylinder designation is incorrect or if a code error has been introduced. See Sect. F11.4.4 for correct hemicylinder specification information.

K5-90 THE UNIT ORIENTATION DESCRIPTION MAY APPEAR TO BE INCORRECT BECAUSE THE ARRAY SIZE WAS INCORRECTLY SPECIFIED.

This warning message from subroutine PRTLBA occurs if NBXMAX, NBYMAX or NBZMAX (NUX=, NUY=, NUZ=) from the array data, Sect. F11.4.5 was incorrectly specified.

K5-91 ********** UNIT ____ CONTAINS THE FOLLOWING GEOMETRY INCONSISTENCIES. ***********

This message from subroutines JOMCHK, HOLEXT or HOLHOL indicates that one or more intersecting geometry regions were encountered in unit or box type ____. K5-92 is the companion message from subroutine JOMCHK and specifies the regions that intersect. K5-166 is the companion message from subroutine HOLEXT. K5-169 and K5-179 are the companion messages from subroutine HOLHOL. Correct the geometry region data and resubmit the problem. Section F11.4.4 may provide assistance in correctly specifying the data.

K5-92 REGION NUMBER ____ INTERSECTS REGION NUMBER ____

This message from subroutine JOMCHK is a companion message to K5-91 and K5-94. It specifies the intersecting regions. The user must determine which region is incorrectly specified or if the data are out of order. KENO V.a REQUIRES THAT EACH SUCCESSIVE GEOMETRY REGION MUST COMPLETELY ENCLOSE THE PREVIOUS REGION. THIS DOES ALLOW COMMON FACES AND TANGENCY.

K5-93 INVALID GEOMETRY TYPE. IGEO=____

This message is from subroutine JOMCHK. IGEO must be greater than zero and less than 20. If it does not fall in this range, a code error is the probable cause. KENOG or READGM are the subroutines likely to have the code error. Verify that all the geometry words (*fgeom*, Sect. F11.4.4)
are correct.

K5-94 ***** THE REFLECTOR DIMENSIONS ARE INCONSISTENT *****

This message is from subroutine JOMCHK. It is a companion to message K5-95 and is printed whenever one or more intersecting geometry regions are encountered in the external reflector.

K5-95 REGION NUMBER ____ IN UNIT NUMBER ____ CONTAINS AN ERROR IN THE DIMENSIONS.

This message from subroutine VOLUME indicates an error in the geometry input data such that the negative dimension specification for a cube or cuboid is larger than the positive dimension specification (i.e., the -x dimension is greater than the +x dimension, or the -y dimension is greater than the +y dimension or the -z dimension is greater than the +z dimension). This message is also printed if the magnitude of the chord for a hemicylinder or hemisphere is larger than the radius. See Sect. F11.4.4 for assistance in specifying the geometry correctly.

K5-96 THE VOLUME DEFINED BY GEOMETRY CARD ____ IS NEGATIVE.

This message from subroutine VOLUME is printed whenever a negative volume is calculated. This can be caused by the positive dimension being smaller than the negative dimension on a face of a geometry region. It can also be caused by having intersecting regions, or be the result of roundoff when the volumes are calculated. Either the geometry regions are incorrectly specified, or the data are out of order, or the dimensions are so tight fitting that roundoff causes the net volume of the region to be negative. If the error is caused by roundoff, adjust the appropriate dimensions slightly. See Sect. F11.4.4.

K5-97 ERROR ERROR. THE VOLUME FOR UNIT ____ IS NEGATIVE.

This message is from subroutine VOLUME. A negative volume for a unit or box type can be caused by having intersecting regions within the unit, or by having a unit or box type consisting of one region and having a positive dimension smaller than the negative dimension on one or more faces. Message K5-95 or K5-96 may accompany this message. See Sect. F11.4.4 for assistance in specifying the geometry data correctly.

K5-98 INVALID GEOMETRY ENCOUNTERED FOR THE LAST GEOMETRY REGION. IGEO= ___

This message is from subroutine CORSIZ. It indicates that an unrecognized geometry word was entered for the last geometry region. This fact should have triggered several error messages prior to this time. This message is usually indicative of a code error.

K5-99 *** ERROR IN UNIT ____ *** THE LAST GEOMETRY REGION OF A UNIT UTILIZED IN THE UNIT ORIENTATION DESCRIPTION OF THE ARRAY DATA MUST BE A CUBE OR CUBOID.

This message is from subroutine ARASIZ. It can occur when a single unit problem whose outer region is not a cube or cuboid is specified as a 1 × 1 × 1 array. To eliminate this message, add a cuboidal outer region containing void, or remove the array data. If the problem is an array problem, be sure each unit used in the unit orientation description ends with a cube or cuboid. See Sects. F11.4.4 and F11.4.5.

K5-100 THIS PROBLEM WILL NOT BE RUN BECAUSE ERRORS WERE ENCOUNTERED IN THE INPUT DATA.

This message is from subroutine MASTER and indicates that other error messages were printed in the problem output. Find these messages and correct the data accordingly. A STOP 129 is executed when this message is printed.

K5-101 *** ERROR *** NO FISSILE MATERIAL WAS FOUND IN SUBROUTINE START.

This message is from subroutine START. It indicates that none of the mixtures utilized in this problem have a fission spectrum associated with them. Either the geometry data did not specify a fissionable mixture number, the mixing table is incorrect, the wrong mixed cross section data set was mounted, or the mixed cross-section data set was incorrectly or incompletely made. A STOP 128 is executed in conjunction with this message, and a traceback may be printed from subroutine STOP.

K5-102 THE START DATA SPECIFIES THAT NEUTRONS CAN BE STARTED IN THE REFLECTOR. HOWEVER NEUTRONS WILL NOT BE STARTED BECAUSE THE OUTER REGION OF THE REFLECTOR IS NOT A CUBE OR CUBOID. NEUTRONS CAN BE STARTED FOR THE EXISTING GEOMETRY IF XSM, XSP, YSM, YSP, ZSM AND ZSP ARE ENTERED AS START DATA.

This message is from subroutine START. Start type 0 allows starting points throughout noncuboidal regions. If a start type other than 0 or 6 is desired and the outermost region of the reflector is not a cube or cuboid, data must be input to specify an imaginary cube or cuboid within this outer region. See Sect. F11.4.8 for assistance in specifying this data.

K5-103 START TYPE ___ IS OUT OF RANGE.

This message from subroutine START indicates that the start type was less than zero or greater than 6. The start type is defined by entering the keyword NST= followed by the desired start type in the start data. The available starting options are given in Table F11.4.6, Sect. F11.4.8.

K5-104 A POSIT ERROR INDICATES THAT THE POINT X=____ Y=____ Z=____ DOES NOT OCCUR WITHIN UNIT ____.

If XSM, XSP, YSM, YSP, ZSM, ZSP were entered in the start data. Verify that they fit within the overall coordinates of the system. The overall coordinates may not be printed for a bare array. This message from subroutine START may result from precision difficulties. It is allowed to occur a maximum of five times before being considered fatal. A code error may be the cause of this message if it becomes fatal.

K5-105 ***** WARNING, ONLY ____ INDEPENDENT STARTING POSITIONS WERE GENERATED. *****

This message is from subroutine START. KENO V.a must have *npb* (NPG=, see parameter data, Sect. F11.4.3) starting positions: This message is to inform the user that fewer than *npb* starting positions were generated. The remaining starting positions are randomly selected from those that were generated, thus giving duplicate starting positions. If the number of independent starting positions is nearly *npb*, the starting distribution is probably acceptable. If it is much smaller than *npb*, a different start type should be used to give a better starting distribution (see Sect. F11.4.8). The amount of time allowed to generate the starting positions is controlled by parameter TBA= (see Sect. F11.4.3). If the start data are appropriate, it may be necessary to increase the value of TBA to ensure generating *npb* starting positions.

K5-106 POSIT ERROR — ILLEGAL GEOMETRY TYPE X=____ Y=____ Z=____ K1= ____ K2=____ K=____ IGEO=____

This message from subroutine POSIT is usually the result of an incorrectly specified starting point for the initial source distribution when NST=3, 4 or 6 is specified in the start data. The starting point may not be consistent with the unit's position in the global array. The message can also be caused by a code error that was introduced when changes were made to the code. X, Y, Z is the location of the neutron, K1 and K2 are the first and last regions of the unit, and K is the region whose geometry type is illegal. IGEO is the geometry type. If the geometry word (*fgeom*) for the region is correct, IGEO should be correct. See Sect. F11.4.4 for correct geometry words.

K5-107 POSIT ERROR X=____ Y=____ Z=____ K1=____ K2=____

This message from subroutine POSIT may result from precision difficulties or a code error. X, Y, Z is the location of the neutron. K1 is the region it is in and K2 is the region it is trying to go to. K5-104 is a companion message.

K5-108 POSITION (_____, _____, _____) IS NOT VALID FOR THE POSITION OF THE SPIKE FOR START TYPE 2.

This message from subroutine START2 indicates that NXS, NYS or NZS was entered as zero. See Sect. F11.4.8 for the correct start data specification.

K5-109 ***** INVALID GEOMETRY TYPE IN START. IGEO=____ *****

This message from subroutine STRTSU is probably the result of a code error.

K5-110 ***** ERROR ERROR THE PROBLEM WILL NOT BE EXECUTED BECAUSE NO FISSILE MATERIAL WAS FOUND. *****

This message from subroutine VOLFIS occurs when the volume of fissile material is found to be zero. Check to be sure the fissile material was correctly specified in the geometry data, check the volume of the fissile material in the printout to be sure it is nonzero, verify that the mixing table is correct or the correct ice mixed cross-section data set is used if a mixing table is not used. A STOP 131 is executed when this message is printed.

K5-111 ** RESTART DATA IS NOT AVAILABLE FOR RESTARTING WITH GENERATION ____ AS SPECIFIED IN THE INPUT DATA **
** HOWEVER, AVAILABLE RESTART DATA WAS ALLOWED RESTARTING WITH GENERATION _____. **

This message from subroutine RDCALC indicates that *nbas* (BEG=, Sect. F11.4.3) was not consistent with the set of restart data that was to be used. A set of restart data is written every *nrstrt* (RES=, Sect. F11.4.3). The value of *nbas* should be 1 greater than one of these generations.

K5-112 ERROR IN RESTART. PARAMETER DATA AND RESTART DATA DO NOT AGREE.
NUMBER PER GENERATION FROM RESTART, NPBT=____
NUMBER PER GENERATION FROM INPUT DATA, NPB=____
NUMBER OF ENERGY GROUPS FROM RESTART, NGPT=____
NUMBER OF ENERGY GROUPS FROM INPUT DATA, NGP=____

This message is from subroutine RDCALC. A restarted problem MUST use the same number per generation and the same number of energy groups as the parent problem that wrote the restart data. Verify that the correct data set is mounted on unit *rstrt*. (RST=, Sect. F11.4.3). This message can also be caused by a code error.

K5-113 ERROR IN RESTART. PARAMETER DATA AND RESTART DATA DO NOT AGREE.
FISSION DENSITIES, FLUXES, OR REGION DEPENDENT FISSIONS AND ABSORPTIONS WERE REQUESTED, BUT THE GEOMETRY DATA IS INCONSISTENT.
NUMBER OF GEOMETRY REGIONS FROM RESTART, KREFT=____
NUMBER OF GEOMETRY REGIONS FROM INPUT DATA, KREFM=____

This message is from subroutine RDCALC. Verify that the correct data set is mounted on unit *rstrt* (RST=, Sect. F11.4.3). A code error can also cause this message.

K5-114 *** ERROR ***** ERROR *** PARAMETER DATA SPECIFIED FLUXES BUT THE RESTART DATA DID NOT INCLUDE FLUXES.

This message is from subroutine RDCALC. The restarted problem can turn off fluxes if the parent case that wrote the restart data set calculated fluxes. However, if the parent case did not calculate fluxes, the restarted problem cannot calculate fluxes either. If the correct restart data set was mounted on *rstrt* (RST=, Sect. F11.4.3). The parameter data FLX=YES must be removed from the input data or FLX=NO must be entered later in the parameter data of the restarted problem.

K5-115 *** ERROR ***** ERROR *** PARAMETER DATA SPECIFIED REGION DEPENDENT FISSIONS AND ABSORPTIONS, BUT THEY WERE NOT INCLUDED ON RESTART.

This message is from subroutine RDCALC. The restarted problem specified FAR=YES in the parameter data block but the parent case that wrote the restart data set did not calculate region-dependent fissions and absorptions. The restarted problem can turn off region-dependent data if the parent case calculated them, but cannot turn them on if they were not calculated by the parent case. Verify that the correct restart data set is mounted on *rstrt* (RST=, Sect. F11.4.3). Remove FAR=YES from the parameter data of the restarted problem or add FAR=NO later in the parameter data. Section F11.5.3 illustrates methods of changing the parameter input data.

K5-116 *** ERROR ***** ERROR *** EXECUTION IS TERMINATED *** ERROR ***
*** ERROR ***

This message from subroutine RDCALC is a companion to messages K5-112 through K5-115. A STOP 121 is executed when this message is printed.

K5-117 THE CALCULATION WAS TERMINATED BECAUSE ERRORS WERE ENCOUNTERED IN THE START DATA.

This message is from subroutine GUIDE. It will be accompanied by one or more of messages K5-101 through K5-104 or K5-106 through K5-110. A STOP 130 is executed when this message is printed.

K5-118 EXECUTION TERMINATED. RAKBAR HAS BECOME ZERO OR NEGATIVE.

This message is from subroutine GUIDE. If this message appears without other error messages, a code error is the probable cause.

K5-119 JOB PULLED GENERATION=____ NEUTRON=____

This message from subroutine GUIDE indicates that the problem is looping, or the time allotted for each generation, *tbtch* (TBA=, Sect. F11.4.3) is too small. If *tbtch* (TBA=) is increased significantly and the message occurs again for the same generation and the same neutron, it is due to a code error. Without a functional system-dependent routine (PULL) to interrupt execution, the problem will loop indefinitely, and this message will not be printed.

K5-120 EXECUTION TERMINATED DUE TO INSUFFICIENT I/Os. APPROXIMATELY ____I/Os ARE NEEDED PER GENERATION, BUT ONLY ____ REMAIN.

This message is from subroutine GUIDE. The problem can be resubmitted if more histories are desired. Be sure to change the job control language to request sufficient I/Os to allow the problem to run. This message is inaccessible on computers lacking job control language to specify a maximum number of I/Os for a problem.

K5-121 EXECUTION TERMINATED DUE TO INSUFFICIENT TIME IN THE JOB STEP. ____ SECONDS ARE NEEDED PER GENERATION, BUT ONLY ____ REMAIN IN THE JOB STEP.

This message is from subroutine GUIDE. If more histories are desired, change the job control language to allow adequate time and resubmit the problem. This message is inaccessible on computers lacking job control language to specify the maximum execution time.

K5-122 EXECUTION TERMINATED DUE TO EXCEEDING THE TIME SPECIFIED FOR THE PROBLEM.

This message is from subroutine GUIDE. If more histories are desired, increase *tmax* (TME=, Sect. F11.4.3) to allow computation of the desired number of histories.

K5-123 EXECUTION TERMINATED DUE TO COMPLETION OF THE SPECIFIED NUMBER OF GENERATIONS.

This message from subroutine GUIDE states that the requested number of histories have been completed. If more histories are desired, increase the number of generations (GEN=, Sect. F11.4.3).

K5-124 ***** ERROR ***** THE OPTION TO USE EXTRA 1-Ds WAS SPECIFIED BUT ID NO. ____ WAS NOT FOUND IN THE EXTRA 1-D ARRAY.

This self-explanatory message is from function INDX. If extra 1-Ds are specified in the parameter data (X1D=, Sect. F11.4.3), extra 1-D IDs must be entered as data. See Sect. F11.4.9. A STOP 107 is executed when this message is printed.

K5-125 EXCEEDED NEUTRON BANK SIZE

This message from subroutine BANKER indicates that the number of banked particles exceeds the bank size. This can be corrected by increasing *nbank* (NBK=, Sect. F11.4.3).

K5-126 ***** CROSS ERROR ____ ____ ____ ____ ____ ____ ____ ____

This message from subroutine CROS indicates a code error. The printed data, left to right, are as follows: IGEO,K,X,Y,Z,X1,Y1,Z1. IGEO is the geometry type, K is the region number, X,Y,Z is the current position and X1,Y1,Z1 is the end point of the path. A STOP 103 is executed when this message is printed.

K5-127 ********** ERROR.....NHCYL____ **********

This message from subroutine CROS indicates invalid hemicylinder information as the result of a code error. A STOP 104 is executed when this message is printed.

K5-128 NEUTRON BANK IS FULL. SPLITTING NOT ALLOWED.

This message from subroutine TRACK indicates that the neutron bank is too small to allow additional splitting. This can occur if the bank size, *nbank* (NBK=, Sect. F11.4.3) is too small, if the biasing or weighting data are incorrect (Sect. F11.4.7), or if the biasing data are incorrectly utilized in the geometry description (Sect. F11.4.4).

K5-129 *** ERROR IN SUBROUTINE ALBIN *** FACE NUMBER ____ USES ____ ____
ALBEDO NUMBER=____ INCIDENT XSEC ENERGY GROUP=____
INCIDENT ALBEDO ENERGY GROUP=____
INCIDENT ANGLE INDEX=____ RANDOM NUMBER=____.

This message from subroutine ALBIN indicates that a code error was encountered when trying to determine the output energy group during the albedo treatment.

K5-130 *** ERROR IN SUBROUTINE ALBIN *** FACE NUMBER____ USES ____ ____
ALBEDO NUMBER=____ RETURNING XSEC ENERGY GROUP=____
INCIDENT ALBEDO ENERGY GROUP=____
INCIDENT ANGLE INDEX=____
RETURNING ALBEDO ENERGY GROUP=____
RANDOM NUMBER=____.

This message from subroutine ALBIN indicates that a code error was encountered while trying to compute the returning angle in the albedo treatment.

K5-131 NO FISSIONS

This message from subroutine NSTART indicates that none of the generations encountered a fissile material so no fission points were generated.

K5-132 WARNING....ONLY _____ INDEPENDENT FISSION POINTS WERE GENERATED

This message from subroutine NSTART indicates that fewer than *npb* (NPG=, Sect. F11.4.3) fission points were generated during the previous generation. Because *npb* fission points are required by the code, the remaining fission points are randomly selected from those that were generated, thus utilizing duplicate fission points. If the k-effective of the system is significantly less than 1.0, several of these messages should be expected in the first few generations. The code attempts to set RAKBAR so the message can be expected to occur about once every 100 generations. The message may occur more frequently in a correctly modeled problem. However, if the number of fission points is considerably less than *npb* for most of the generations, the answer can be affected.

K5-133 NUMBER OF GENERATIONS RUN WAS INSUFFICIENT TO EDIT

This message from subroutine KEDIT occurs if the number of generations completed is less than *nskip*+1 (NSK=, Sect. F11.4.3). In this instance, the summaries for the problem cannot be printed.

K5-134 FLUXES FOR UNIT _____ WILL NOT FIT IN CORE.

This message from subroutine FITFLX indicates that the flux array is too large for the available core space. More space can be allocated in the job control language if desired.

K5-135 GEOMETRY TYPE - IGEO - OUT OF RANGE IN CRMAX

This message from subroutine CRMAX occurs if the geometry indicator is invalid as the result of a code error. A STOP 102 is executed in conjunction with this message and a traceback may be printed from subroutine STOP.

K5-136 GEOMETRY TYPE - IGEO - OUT OF RANGE IN SRMAX

This message from subroutine SRMAX occurs if the geometry indicator is invalid as the result of a code error. A STOP 127 is executed in conjunction with this message and a traceback may be printed from subroutine STOP.

K5-137 INPUT DATA SAID TO PRINT WEIGHTS, BUT THEY WERE NOT PRINTED BECAUSE THE NUMBER OF ENERGY GROUPS EXCEEDS THE AVAILABLE SPACE.

This message is from subroutine MASTER. In order to accommodate the specified number of energy groups, more computer storage must be requested in the job control language.

K5-138 ********** A WEIGHT OF 0.0 INDICATES THAT WEIGHTS WERE NOT READ OR GENERATED FOR THE BIAS ID. **********
********** WEIGHTS OF 0.0 WILL BE DEFAULTED TO 0.5 PRIOR TO EXECUTION **********

This message is from subroutine PRTWTS. It is printed to alert the user that weights were not entered, defaulted, or generated. This message may appear as the result of a code error.

K5-139 ***** ERROR ***** NO VALID MIXTURES WERE FOUND IN THE GEOMETRY DESCRIPTION.

This message from subroutine ICEMIX indicates that the geometry data did not specify any valid mixtures. Check the geometry data (Sect. F11.4.4) and correct any errors that are found. This message can also be triggered if the unit orientation data description is not properly entered for geometry having more than one unit or box type.

K5-140 ***ERROR*** NOT ENOUGH STORAGE FOR START TYPE 6 DATA. 1ST IS AMT NEEDED, 2ND IS AMT AVAILABLE.
PERTINENT CONSTANTS (1) (2)

This message from subroutine SAVST6 indicates that more computer storage is necessary to run this problem. At least (1) words of storage are needed to run the problem, but only (2) words of storage were available. Increase the amount of computer storage requested in the job control language to correspond to the amount of storage needed. A STOP 126 is executed in conjunction with this message, and a traceback may be printed from subroutine STOP.

K5-141 DATA CANNOT BE CHANGED WHEN A PROBLEM IS RESTARTED AT A GENERATION GREATER THAN 1.

This message from subroutine DATAIN is printed if data other than parameter data are entered for a problem being restarted at a generation greater than 1. If data other than certain parameter data are to be changed, the problem must be restarted with the first generation. The error flag is set so the problem will not execute.

K5-142 NO GEOMETRY DATA HAS BEEN SPECIFIED IN THE INPUT DATA.

This message from subroutine DATAIN indicates that a geometry data block was not entered for the problem either as input data or from the restart unit. Correct the data and resubmit the problem. A STOP 135 is executed in conjunction with this message, and a traceback may be printed from subroutine STOP.

K5-143 UNIT ORIENTATION DATA IS REQUIRED IF MORE THAN ONE UNIT TYPE IS SPECIFIED IN THE GEOMETRY DATA.

This self-explanatory message is from subroutine DATAIN. Enter an array i or unit orientation data block as described in Sects. F11.4.5 and F11.5.6. A STOP 136 is executed in conjunction with this message and a traceback may be printed from subroutine STOP.

K5-144 DUE TO INCONSISTENCIES BETWEEN INPUT AND RESTART DATA, MATRIX INFORMATION BY UNIT TYPE WILL BE CALCULATED BUT NOT PRINTED. INPUT DATA SET MKU=NO, BUT DATA FROM THE RESTART UNIT SPECIFIED YES.

This self-explanatory warning message is from subroutine PARAM. The matrix information by unit type cannot be eliminated if it was calculated by the original problem (parent case) that wrote the restart data. However, printing it can be avoided. Verify that the correct problem is being used for restarting the problem (the title is printed at the bottom of the parameter tables). Also verify the specification of the restart unit, RST, in the third table of the output.

K5-145 DUE TO INCONSISTENCIES BETWEEN INPUT AND RESTART DATA, MATRIX INFORMATION BY UNIT LOCATION WILL BE CALCULATED BUT NOT PRINTED. INPUT DATA SET MKP=NO, BUT DATA FROM THE RESTART UNIT SPECIFIED YES.

This self-explanatory warning message is from subroutine PARAM. The matrix information by unit location (also called array position or position index) cannot be eliminated if it was calculated by the original problem (parent case) that wrote the restart data. However, printing it can be avoided. Verify that the correct problem is being used for restarting the problem (the title is printed at the bottom of the parameter tables). Also verify the specification of the restart unit, RST, in the third table of the output.

K5-146 *** ERROR ***** ERROR *** PARAMETER DATA SPECIFIED MATRIX INFORMATION BY UNIT TYPE BUT IT WAS NOT FOUND ON THE RESTART UNIT.

This message from subroutine RDCALC is printed if a restarted problem requests matrix information by unit type when it was not requested and calculated by the original problem (parent case) that wrote the restart data. Verify that the correct restart data are being used and that the restart unit (RST) is correctly specified. Eliminate the request for matrix information by unit type (MKU=, in the parameter data) if it is not necessary. The problem must be restarted with the first generation (BEG=1, in the parameter data) if matrix information by unit type is required and was not calculated by the parent case. A STOP 121 is executed in conjunction with this message.

K5-147 *** ERROR ***** ERROR *** PARAMETER DATA SPECIFIED MATRIX INFORMATION BY UNIT LOCATION BUT IT WAS NOT FOUND ON THE RESTART UNIT.

This message from subroutine RDCALC is printed if a restarted problem requests matrix information by unit location (also called array position or position index) when it was not requested and calculated by the original problem (parent case) that wrote the restart data. Verify that the correct restart data are being used and that the restart unit (RST) is correctly specified in the first table following the parameter table. Eliminate the request for matrix information by unit location (MKP=, in the parameter data) if it is not necessary. If matrix information by unit location is required and it was not calculated by the parent case, the problem must be restarted with the first generation (BEG=1, in the parameter data). A STOP 121 is executed in conjunction with this message.

K5-149 A CROSS SECTION LIBRARY WAS SPECIFIED FOR A RESTARTED PROBLEM, BUT MIXING TABLE DATA WAS NOT AVAILABLE. THE PROBLEM WILL NOT EXECUTE.

This message from subroutine RDRST means that LIB= was entered in the parameter data block and a mixing table data block was not available. A flag is set to terminate execution when the data reading has been completed. If cross sections are to be used from the restart unit (RST=), eliminate the LIB= or XSC= from the parameter data. If new cross sections are to be mixed, LIB= must be specified in the parameter data. The IDs in the mixing table must be available on the cross-section library specified by LIB=. A problem can be restarted using a new mixed cross-section library by specifying XSC= in the parameter data.

K5-150 *** ERROR ****** ERROR *** TOO FEW ENTRIES WERE SUPPLIED IN THE REFLECTOR GEOMETRY DESCRIPTION.

This message from subroutine KENOG indicates that too few data entries were supplied for the geometry word REFLECTOR. The mixture ID, bias ID, one of the thickness/region specifications or the number of regions to be generated was omitted or incorrectly specified. Each REFLECTOR entry requires (1) a mixture ID, (2) a bias ID, (3) N entries for the thickness/region specifications, and (4) the number of regions to be generated. The thickness/region can be obtained from the Increment Thickness column of Table F11.4.5 for the material to be used in the regions generated by the REFLECTOR specification. N is the number of thickness/region specifications required by the geometry shape: N=1 for spheres or hemispheres, N=3 for cylinders and hemicylinders, and N=6 for cubes, cuboids, and cores. A flag is set to terminate the problem when the input data reading is completed.

K5-151 *** ERROR *** ____ IS AN INVALID PARAMETER NAME IN THE ARRAY DATA

This message from subroutine ARAYIN indicates that a parameter name was misspelled or the data were out of order. See Sect. F11.4.5 for a list of the array parameter names.

K5-152 *** ERROR *** IRET=____ A PREMATURE TERMINATION WAS ENCOUNTERED WHILE READING ARRAY DATA.
IRET=1 INDICATES AN END WAS FOUND. IRET=2 INDICATES AN END OF FILE.

This message from subroutine ARAYIN indicates that an array number was specified without entering the corresponding UNIT ORIENTATION DESCRIPTION. See Sects. F11.4.5 and F11.5.6 for assistance.

KF-153 FIRST TWO NUMBERS ARE WORDS OF STORAGE NEEDED AND ALLOCATED. THIRD IS ADDITIONAL REGION SIZE NEEDED.

This message from subroutine SORTA indicates that the allocated computer storage is too small to allow loading the unit orientation data prior to determining the nesting level of the arrays defined by the unit orientation data. The first number printed is the amount of storage, in words, needed to hold these data. The second number is the amount of computer storage that was allocated, and the third number is the minimum additional region size, in units of K bytes, necessary to hold these data. Increase the region size for the "go step" in the job control language by the additional required region size (the third number) and resubmit the problem.

K5-154 *** ERROR *** LEVELA=______ IS LARGER THAN ______, THE NUMBER OF ARRAYS.

This message from subroutine SORTA indicates that the array data specified in the problem are recursively nested. An example of this is the following:

array 1 contains array 3
array 2 contains array 1
array 3 contains array 2

Thus the definition of array 1 and array 3 are intertwined in a never-ending loop. Correct the array data (Sect. F11.4.5) and resubmit the problem. If the input data did not specify recursive nesting, a code error has occurred.

K5-155 ERROR *** THE NESTING FLAG OR NUMBER OF ARRAY LEVELS HAS BEEN DESTROYED BY A CODE ERROR.
THE ORIGINAL NESTING FLAG WAS SET ______. IT IS NOW SET ______.
THE ORIGINAL NESTING LEVEL WAS ______. IT IS NOW ______.

This self-explanatory message is from subroutine LODARA. A STOP 140 is executed in conjunction with this message.

K5-156 *** ERROR *** MIXTURE CONTAINS AT LEAST ONE ZERO VALUE FOR THE TOTAL CROSS SECTION.

This message is from subroutine XSEC1D. All the total cross sections must be positive. Zero total cross sections can occur if all the components of a mixture are mixed with a zero-number density. Correct the mixing table for the specified mixture and resubmit the problem.

K5-157 *** ERROR *** THE FIRST HOLE IN A UNIT MUST FOLLOW A VALID GEOMETRY REGION.

A STOP 141 accompanies this message from subroutine READGM. If holes are to be utilized in the geometry region data (Sect. F11.4.4), they must follow the region in which they are to be placed. This message indicates that HOLE was the first geometry description in a unit or was placed inside a CORE or ARRAY description. Correct the geometry region data and resubmit the problem.

K5-158 *** ERROR *** ERROR IN THE NUMBER OF HOLES. IHOL= NUMHOL=______

This message is from subroutine READGM. A code error is the probable cause of this error.

K5-159 *** ERROR *** ARRAY NUMBER _____ SPECIFIED IN THE GEOMETRY REGION DATA WAS NOT ENTERED IN THE ARRAY DATA.

This message from subroutine SORTA or subroutine HOLE occurs if the array number specified for an. ARRAY region description of the EXTENDED GEOMETRY data (Sect. F11.4.4) did not have the corresponding UNIT ORIENTATION DATA entered in the ARRAY DATA (Sect. F11.4.5). A STOP 142 is executed in conjunction with this message when it is printed from subroutine SORTA. Correct the data and resubmit the problem.

K5-160 THE HOLES ARE RECURSIVELY NESTED.

This message from subroutine HOLE indicates that the geometry region data description (Sect. F11.4.4) specifies holes that are recursively nested. This can happen if a unit contains a hole whose definition traces back to the same unit or are defined in terms of each other. A simple example of recursive nesting is

```
UNIT 1   CUBE  0  1  2PI0.0   HOLE  2  3*0.0
UNIT 2   CUBE  0  1  2PI0.0   HOLE  1  3*0.0
```

Thus unit 1 contains unit 2 and unit 2 contains unit 1. Check the geometry region data for recursive nesting. In the absence of recursive nesting, a code error is the probable cause of this message. A STOP 143 is executed when this message occurs and a traceback is printed.

K5-161 *** ERROR *** THE GLOBAL ARRAY SPECIFIED IN THE GEOMETRY REGION DATA IS ______, BUT THE GLOBAL ARRAY SPECIFIED IN THE ARRAY DATA IS _____ EXECUTION IS TERMINATED.

This message is from subroutine FLDATA. A STOP 144 is executed in conjunction with it. The global array specified in the array data (Sect. F11.4.5) was entered using the keyword GBL=. It was not consistent with the implied global array from the geometry region data (Sect. F11.4.4). The global array number in the geometry region data is defined to be the array number of the last ARRAY or CORE description that does not immediately follow a UNIT or BOX TYPE definition (i.e., other geometry definitions occur between UNIT and CORE or ARRAY). Correct the data and resubmit the problem.

K5-162 *** ERROR *** THE GLOBAL ARRAY WAS NOT CONSISTENTLY SPECIFIED. THE ARRAY DATA SPECIFIED ARRAY NUMBER _____ AND THE GEOMETRY DATA SPECIFIED ARRAY NUMBER _____.

This message from subroutine SORTA occurs if GBL= in the array data (Sect. F11.4.5) does not agree with the global array number implicitly set in the geometry region data (Sect. F11.4.4). The geometry region data define the global array number to be the array number of the last ARRAY or CORE description that does not immediately follow a UNIT or BOX TYPE definition (i.e., other geometry definitions occur between UNIT and CORE or ARRAY). A flag is set to terminate execution when the data reading is completed.

K5-163 *** ERROR *** ARRAY NUMBER _____ IS ____ WORDS LONG, BUT ONLY _____ WORDS ARE AVAILABLE WHEN WRITING RESTART. THE RESTART FILE WILL BE INCOMPLETE. THE PROBLEM WILL NOT EXECUTE.

This message is from subroutine WRTARA. It indicates that the available computer storage is too small to hold the array data from the direct-access device. This in turn will cause the restart data file to be incomplete. Increase the requested storage space in the job control language and resubmit the problem.

K5-164 *** WARNING *** ARRAY NUMBER ______ IS _____ WORDS LONG, BUT ONLY _ ___ WORDS ARE AVAILABLE WHEN READING RESTART. THIS ARRAY WILL NOT BE AVAILABLE. THE PROBLEM WILL NOT EXECUTE IF THIS ARRAY IS USED.

This message is from subroutine RDARA. It occurs when the restart data file is being loaded from the restart unit and the allocated space is insufficient to hold the data for the specified array. Resubmit the problem, requesting more storage space in the job control language.

K5-165 *** ERROR *** GEOMETRY TYPE OUT OF RANGE IN SUBROUTINE ADJUST. IGEOH= ______.

This message from subroutine ADJUST indicates that the geometry type, IGEO, falls outside the allowable range of 1 through 19. This can occur if a HOLE references an undefined unit number. If all the unit numbers referenced by the holes are valid, a code error is the probable cause.

K5-166 HOLE NUMBER ______ INTERSECTS REGION NUMBER

This form of message K5-166 is from subroutine HOLEXT. It indicates that the specified hole intersects the region in which the bole was placed. A flag is set to terminate execution when the data reading is completed. Correct the geometry data (Sect. F11.4.4) and resubmit the problem.

K5-167 GEOMETRY TYPE - IGEO - OUT OF RANGE IN CRMIN

This message from subroutine CRMIN is accompanied by a STOP 145. The geometry type, IGEO, must be greater than zero and less than 20. If it does not fall in this range, a code error is the probable cause.

K5-168 GEOMETRY TYPE - IGEO - OUT OF RANGE IN SRMIN

This message from subroutine SRMIN is accompanied by a STOP 146. The geometry type, IGEO, must be greater than zero and less than 20. If it does not fall in this range, a code error is the probable cause.

K5-169 HOLE NUMBER ______ INTERSECTS REGION NUMBER ______

This message from subroutine HOLHOL occurs if a hole intersects the first region interior to the region in which the hole is emplaced. This may be a real intersection or it may be due to roundoff in processing the holes. If it is due to roundoff, adjust *xhole, yhole and/or zhole* and/or the dimensions as appropriate. This message indicates that the problem will be terminated when the data reading is completed.

K5-171 ERROR - DIRECTION COSINES DOWN THE PAGE WERE ALL INPUT AS ZERO.

This message from subroutine RDPLOT indicates that the values for UDN=, VDN= and WDN= were all zero. A zero-value vector does not define a direction, so an error has occurred. See Sect. F11.4.11 for information concerning direction cosines down the page.

K5-172 ERROR - DIRECTION COSINES ACROSS THE PAGE WERE ALL INPUT AS ZERO.

This message from subroutine RDPLOT indicates that the values for UAX=, VAX= and WAX= were all zero. This is an error because a zero-value vector does not define a direction. See Sect. F11.4.11 for assistance in defining direction cosines across the page.

K5-173 ERRORS WERE DETECTED IN THE INPUT DATA FOR THIS PICTURE. IT WILL NOT BE DRAWN.

This message from subroutine RDPLOT is a companion to messages K5-174, K5-180, K5-171, and K5-172. Correct the error that triggered the companion message and resubmit the problem.

K5-174 ERROR IN KENO 5 PICTURE DATA - _____ _____ SHOULD BE ENTERED AS _____YES OR _____NO.

This self-explanatory message is from subroutine RDPLOT. Correct the error and resubmit the problem. See Sect. F11.4.11 for assistance.

K5-175 DUE TO INCONSISTENCIES BETWEEN INPUT AND RESTART DATA, MATRIX INFORMATION BY HOLE WILL BE CALCULATED BUT NOT PRINTED. INPUT DATA SET MKH=NO, BUT DATA FROM THE RESTART UNIT SPECIFIED YES.

This self-explanatory warning message is from subroutine PAPAM. The matrix information by hole cannot be eliminated if it was calculated by the original problem (parent case) that wrote the restart data. However, printing it can be avoided. Verify that the correct problem is being used for restarting the problem (the title is printed at the bottom of the parameter tables). Also verify the specification of the restart unit, RST, in the third table of the computer output.

K5-176 DUE TO INCONSISTENCIES BETWEEN INPUT AND RESTART DATA, MATRIX INFORMATION BY ARRAY WILL BE CALCULATED BUT NOT PRINTED. INPUT DATA SET MKA=NO, BUT DATA FROM THE RESTART UNIT SPECIFIED YES.

This self-explanatory warning message is from subroutine PARAM. The matrix information by array cannot be eliminated if it was calculated by the original problem (parent case) that wrote the restart data. However, printing it can be avoided. Verify that the correct problem is being used for restarting the problem (the title is printed at the bottom of the parameter tables). Also verify the specification of the restart unit, RST, in the third table in the computer output.

K5-177 *** ERROR ****** ERROR *** PARAMETER DATA SPECIFIED MATRIX INFORMATION BY HOLE BUT IT WAS NOT FOUND ON THE RESTART UNIT.

This message from subroutine RDCALC is printed if a restarted problem requests matrix information by hole when it was not requested and calculated by the original problem (parent case) that wrote the restart data. Verify that the correct restart data file is being used and that the restart unit (RST) is correctly specified. Eliminate the request for matrix information by hole (MKH=, in the parameter data) if it is not necessary. The problem must be restarted with the first generation (BEG=1, in the parameter data) if matrix information by hole is required and was not calculated by the parent case. A STOP 121 is executed in conjunction with this message.

K5-178 *** ERROR ****** ERROR *** PARAMETER DATA SPECIFIED MATRIX INFORMATION BY ARRAY BUT IT WAS NOT FOUND ON THE RESTART UNIT.

This message from subroutine RDCALC is printed if a restarted problem requests matrix information by array (also called array position or position index) when it was not requested and calculated by the original problem (parent case) that wrote the restart data. Verify that the correct restart data are being used and that the restart unit (RST) is correctly specified in the first table following the parameter tables. Eliminate the request for matrix information by array (MKA=, in the parameter data) if it is not necessary. If matrix information by array is required and it was not calculated by the parent case, the problem must be restarted with the first generation (BEG=1, in the parameter data). A STOP 121 is executed in conjunction with this message.

K5-179 HOLE NUMBER ____ INTERSECTS HOLE NUMBER ____ IN REGION NUMBER ____

This message from subroutine HOLHOL indicates that the specified holes intersect. Check the dimensions and origins of the units being placed in the region.

K5-180 ERROR IN KENO 5 PICTURE DATA - KEYWORD _____ IS NOT VALID.

This message from subroutine RDPLOT indicates that the plot data are out of order or a keyword is incorrectly spelled. See Sect. F11.4.11 for a list of correct keywords.

K5-181 *** ERROR *** LPIC IS OUT OF RANGE. LPIC=______

This message from subroutine PRTPLT indicates that a code error has occurred or the type of plot (PIC=) was not properly specified. *lpic*=l for a mixture map, *lpic*=2 for a unit map, and *lpic*=3 for a bias ID map. Any other values of *lpic* are invalid.

K5-182 INSUFFICIENT SPACE FOR PLOTTING ARRAYS. FIRST NUMBERS ARE STORAGE NEEDED AND ALLOCATED. LAST IS REQUIRED ADDITIONAL REGION SIZE.

This message from subroutine MASTER is accompanied by a STOP 147. It indicates that the allocated computer storage will not hold the plot data. The first number printed is the amount of storage, in words, needed to hold this data. The second number is the amount of computer storage that was allocated, and the third number is the minimum additional region size, in units of K bytes, necessary to hold the data. Increase the region size for the "go step" in the job control language by this third number and resubmit the problem. If additional computer storage is not available, eliminate the fluxes, fissions, and absorptions by region, matrix information, or other space-consuming options (see Sect. F11.4.3).

K5-183 *** WARNING *** UNIT ____ WAS NOT CHECKED FOR GEOMETRIC CONSISTENCY. IT CONTAINS ARRAY _____ BUT WAS NOT USED IN THE PROBLEM.

This warning message is from subroutine JOMCHK. It means that a unit whose first region is an array was described in the extended geometry data but that unit was not referenced in the unit orientation data (see

Sects. F11.4.4 and F11.4.5). KENO usually checks all the geometry region data to be sure it is correct, even when the unit is not used in the problem. The code is unable to make these checks when a unit containing an array is not used in the problem. It is not necessarily an error, but the user should double check to be sure that unit was intentionally omitted from all the arrays.

K5-184 ****** ______ WORDS OF STORAGE ARE NEEDED FOR THE WEIGHTS, BUT ONLY ______ WORDS ARE AVAILABLE. ******

This message from subroutine RDBIAS indicates that the allocated computer storage is not sufficient to process the biasing or weighting data for the problem. Increase the region size in the "go step" of the job control language. The additional required region size is four times (the difference of the two numbers plus 1023) divided by 1024. A STOP 151 is executed when this message is printed.

K5-185 *** ERROR *** THE NUMBER OF SETS OF BIASING CORRELATION DATA EXCEEDSTHE NUMBER THAT WAS WRITTEN WHEN THE BIASING DATA WERE READ. ______ WERE WRITTEN, BUT AN ATTEMPT WAS MADE TO READ ______

This message from subroutine WAITIN is accompanied by a STOP 148. It is indicative of a code error, unless accompanied by error messages related to the biasing input data.

K5-186 *** ERROR *** THE NUMBER OF SETS OF BIASING AUXILIARY DATA EXCEEDS THE NUMBER THAT WAS WRITTEN WHEN THE BIASING DATA WERE READ. ______ WERE WRITTEN, BUT AN ATTEMPT WAS MADE TO READ _____

This message from subroutine WAITIN is accompanied by a STOP 149. It is indicative of a code error, unless accompanied by error messages related to the biasing input data.

K5-187 ***** WARNING - INTERVALS IN THE ABOVE RANGE WERE NOT USED. THIS COULD LEAD TO IMPROPER BIASING. *****

This message from subroutine LODNM is printed to remind the user that at least one of the specified intervals was not utilized in the problem. This can result in improper biasing. Biasing should not be used between fissile units. When biasing is used, it should be flat or increasing as distance from the fissile material increases and flat or decreasing as a history moves toward fissile material. See Sect. F11.5.8 for additional assistance.

K5-188 *** ERROR *** HOLE NUMBER _____ REFERENCES UNDEFINED UNIT NUMBER ______

This message is printed by subroutine READGM if the unit number referenced by the hole is less than 1 or greater than the largest unit number in the geometry data. The message is printed by subroutine HOLE if the unit number referenced by the hole is larger than zero and not greater than the largest unit number in the geometry data, but is a unit number for which all data are missing. The message is printed by subroutine HOLCHK if the unit number referenced by the hole is undefined. Message K5-165 may accompany this message. Specify a valid unit number (*lhole* - see EXTENDED GEOMETRY DESCRIPTION, Sect. F11.4.4) and resubmit the problem.

K5-189 *** ERROR *** ALBEDOS WERE SPECIFIED FOR A PROBLEM WHOSE OUTER BOUNDARY IS NOT A CUBE OR CUBOID.
THE PROBLEM WILL NOT EXECUTE.

This message from subroutine JOMCHK is printed if the outer geometry region is not a cube or cuboid. A REFLECTOR or REPLICATE geometry description immediately following a CORE, ARRAY or CUBOID is considered to be a cuboid. If reflection is desired on a surface of some other shape, it will be necessary to describe actual geometry regions of the appropriate material.

K5-190 ERROR IN PICTURE DATA - OPTION ______ IS NOT VALID FOR KEYWORD PIC=.

This message from subroutine RDPLOT indicates an incorrect option associated with the keyword PIC=. See Sect. F11.4.11. Acceptable options include MAT, MIX, MIXT, MEDI, BOX, BOXT, UNT, UNIT, IMP, BIAS, WTS, WGT, WGTS, or WEIGH.

K5-191 ***** ERROR ***** START TYPE ______ IS INVALID FOR A PROBLEM THAT DOES NOT HAVE A GLOBAL ARRAY. ***** ERROR *****

This message from subroutine DATAIN occurs if the start type (NST= in the start data, Sect. F11.4.8) is 2, 3, 4, or 5. A global array is required in order to use the specified start type.

K5-192 MATRIX INFORMATION BY ARRAY WAS SPECIFIED AS YES IN THE PARAMETER DATA (MKA=), BUT IS NOT ALLOWED BECAUSE ARRAY DATA WERE NOT SPECIFIED.

This warning message from subroutine DATAIN is self-explanatory. The code redefines the problem so matrix information will not be collected by array number.

K5-193 MATRIX INFORMATION BY HOLE WAS SPECIFIED AS YES IN THE PARAMETER DATA (MKH=), BUT IS NOT ALLOWED BECAUSE HOLE DATA WERE NOT SPECIFIED.

This warning message from subroutine DATAIN is self-explanatory. The code redefines the problem so matrix information will not be collected by hole number.

K5-194 MATRIX INFORMATION BY UNIT LOCATION WAS SPECIFIED AS YES IN THE PARAMETER DATA (MKP=), BUT IS NOT ALLOWED BECAUSE A GLOBAL ARRAY WAS NOT SPECIFIED.

This warning message from subroutine DATAIN is self-explanatory. The code redefines the problem so matrix information will not be collected by unit location.

K5-195 ***** ERROR ***** CHARACTER STRING EXCEEDS THE SPECIFIED LENGTH. CHECK FOR ENDING DELIMITER.

This error message is from subroutine RCHRS. It indicates that either the character string exceeds 132 characters or the ending delimiter was omitted for TTL= (plot title, Sect. F11.4.11) or for COM= (unit comment, Sect. F11.4.4, or array comment, Sect. F11.4.5). A STOP 153 is executed when this message is printed.

K5-196 A PROBLEM CANNOT BE RESTARTED WHEN RESTART DATA DO NOT EXIST FOR THE SPECIFIED GENERATION AND THE NEXT GENERATION FOR WHICH RESTART DATA ARE AVAILABLE IS LARGER THAN THE REQUESTED NUMBER OF GENERATIONS. EXECUTION IS TERMINATED.

This message from subroutine RDCALC indicates that a problem was to be restarted but the requested number of generations (GEN=, Sect. F11.4.3) was smaller than the beginning generation number (BEG=, Sect. F11.4.3). The beginning generation number for a restarted problem is the generation at which the calculation of k-effectives and associated information is resumed. Therefore, the number of generations to be run must be larger than the beginning generation number. Correct the data and resubmit the problem. A STOP 154 is executed when this message is printed.

K5-197 *** ERROR IN UNIT *** A VALID GEOMETRY REGION MUST PRECEDE A REPLICATE REGION.

This message from subroutine KENOG indicates that a replicate specification follows an invalid geometry specification (for example, REPLICATE immediately follows a UNIT specification).

K5-198 *** ERROR *** ARRAY _____ CONTAINS AN ERROR IN THE INPUT DATA.

This message from subroutine ARAYIN is printed as the result of an error in the FILL input data for the specified array. One or more messages from the library routine YREAD should immediately precede this message and indicate the nature of the error. Correct the data and resubmit the problem. Messages K5-32, K5-33, and/or K5-85 may also print as a result of this error.

K5-199 *** ERROR *** AN ARRAY WAS SPECIFIED IN THE GLOBAL UNIT, BUT ARRAY DATA WERE NOT ENTERED

This message from subroutine SORTA occurs if array data are not entered when the global unit geometry specifies an array by using either CORE or ARRAY. A STOP 156 is executed in conjunction with this message. Enter the appropriate array data and resubmit the problem.

K5-200 *** ERROR *** START TYPE 6 WAS SPECIFIED IN THE START DATA, BUT THE STARTING POINTS WERE NOT SPECIFIED.

This message from subroutine RDSTRT indicates that start type 6 was specified but the corresponding starting points were not included in the START data block. Enter the corresponding starting points or change the start type. See Sect. F11.4.8.

K5-201 *** WARNING *** NEUTRON _____ SPECIFIED A POSITION IN THE GLOBAL ARRAY. THE GLOBAL UNIT DID NOT CONTAIN AN ARRAY SO THE POSITION WAS IGNORED.

This warning message from subroutine START6 indicates that extraneous data were specified in the start type 6 data. Verify that the correct global unit is specified.

K5-202 *** ERROR *** ONLY START TYPES 0, 1, OR 6 ARE VALID FOR A PROBLEM WITHOUT AN ARRAY IN THE GLOBAL UNIT.

This message from subroutine START indicates that the specified start type is not valid for the problem. Choose an appropriate start type. A STOP 157 is executed when this message is printed.

K5-205 ***** ERROR ***** A GLOBAL UNIT MUST BE SPECIFIED FOR A SINGLE UNIT PROBLEM

This message from subroutine FLDATA is printed if a global unit is not specified for a single unit problem. A STOP 164 is executed when the message is printed. If the input data do not specify the global unit, it is defaulted to unit 1. If several units are described in a problem that does not use array data, the user must specify which unit to use to run the calculation. This is done by entering the word GLOBAL before the UNIT, BOX TYPE, or BOXTYPE of that unit.

K5-206 *** ERROR *** THE UNIT SPECIFIED FOR STARTING IS NOT USED IN THE PROBLEM OR IS UNDEFINED.

This message from subroutine START indicates that the unit in which neutrons are to be started is undefined. Verify that the global unit or array is correctly specified. Check the start data for start types 4 and 5 (Sect. F11.4.8) to be sure NBX= is correctly specified. A STOP 158 is executed when this message is printed.

K5-207 *** ERROR *** THE STARTING ARRAY POSITION IS INVALID.

This message from subroutine START indicates that the array position NXS, NYS, or NZS is not valid for start types 3 or 6. NXS, NYS, and NZS must be larger than zero and no larger than NBXMAX, NBYMAX, and NBZMAX of the global array, respectively. Verify that the global unit or array is correctly specified. Correct the start data (Sect. F11.4.8) and resubmit the problem. A STOP 159 is executed when this message is printed.

K5-208 *** WARNING *** THE FRACTION OF NEUTRONS STARTED AS A SPIKE WAS LESS THAN ZERO. IT HAS BEEN RESET TO ZERO.

This message from subroutine START indicates that FCT= was incorrectly specified in the start data for start type 2. The resultant starting distribution is a cosine distribution throughout the volume of a cuboid defined by XSM, XSP, YSM, YSP, ZSM, and ZSP (see Sect. F11.4.8). If a "spike" was desired, set FCT= to a positive number between 0.0 and 1.0. If FCT=0.0 is specified, a cosine distribution without a "spike" is used as the starting distribution. If FCT=1.0 is specified, all the neutrons are started as a "spike" (i.e., they are started uniformly in the unit located at position NXS, NYS, NZS in the global array), as noted in Sect. F11.4.8)].

K5-209 *** WARNING *** THE FRACTION OF NEUTRONS STARTED AS A SPIKE WAS GREATER THAN 1. IT HAS BEEN RESET TO 1.

This message from subroutine START indicates that FCT= was incorrectly specified in the start type 2 data. The code reset FCT=1 so all of the neutrons are started as a "spike" (i.e., they are started uniformly in the unit located at NXS, NYS, NZS in the global array), as noted in Sect. F11.4.8.

K5-210 *** ERROR *** THE UNIT SPECIFIED FOR STARTING IS NOT IN THE GLOBAL ARRAY.

This message from subroutine START indicates that the unit specified by NBX= does not occur in the global array. Verify that the global array is correctly specified and that the unit specified by NBX= is correct (see Sect. F11.4.8).

K5-211 *** ERROR *** THE NUMBER OF SETS OF BIAS FACTORS FROM CARDS EXCEEDS THE NUMBER THAT WAS WRITTEN WHEN THE BIASING DATA WAS READ.
____ WERE WRITTEN BUT AN ATTEMPT WAS MADE TO READ ____.

This message from subroutine WAITIN is indicative of a code error. A STOP 161 is executed when this message is printed.

K5-212 *** ERROR *** THE BIASING DATA SPECIFIED IBGN=6 AND IEND=2. IBGN MUST BE LARGER THAN ZERO AND IEND MUST BE AT LEAST AS LARGE AS IBGN. THE PROBLEM WILL NOT BE RUN.

This message from subroutine RDBIAS indicates an error in the biasing data. The biasing correlation data is order dependent. The order of data entry is: ID=nn ibgn iend. where nn is an ID number from Table F11.4.5 and ibgn is the beginning BIAS ID and iend is the ending BIAS ID (see Sects. F11.4.7 and F11.5.8). In order to continue checking the input data, if ibgn is less than or equal to zero, it is set to 1. Similarly, if iend is less than ibgn, it is set to ibgn.

K5-213 *** ERROR *** INSUFFICIENT SPACE ALLOCATION. MASTER WAS NOT CALLED.

This error message is printed from subroutine MAIN when KENO V.a is executed in the "stand-alone mode" and from subroutine Q#$009 when executed as part of a CSAS analytical sequence. The message indicates that the job control language must be altered to provide more computer memory to be utilized by the problem. A STOP 162 is executed when this message is printed.

K5-215 *** ERROR *** FIRST NUMBER IS WORDS OF STORAGE NEEDED.

PERTINENT CONSTANTS

This message from subroutine RDCALC indicates there is not enough computer memory available to hold the neutron bank for a restarted problem. Increase the available computer memory and re-run the problem. A STOP 163 is executed when this message is printed.

K5-216 ***** ERROR ***** XNB=____ IS THE MAXIMUM VALUE THAT CAN BE SPECIFIED IN THE PARAMETER DATA. XNB MUST BE SMALL ENOUGH TO FIT IN THE EXTRA ARRAY OF COMMON NUTRON.

This message from subroutine FLDATA indicates that the value of XNB was set too large. Decrease the size of XNB and retry the problem.

K5-217 *** ERROR *** DIFFERENTIAL ALBEDOS CANNOT BE USED IN AN ADJOINT PROBLEM.

This self-explanatory message is from subroutine FLDATA. Describe reflector material in the mixing table and the geometry instead of using differential albedos or run the problem in the forward mode.

K5-218 *** ERROR *** UNIT ____ CONTAINS ARRAY ____ WHICH WAS NOT DEFINED IN THE INPUT DATA.

This message from subroutine JOBCHK occurs if the array number specified for an ARRAY region description (see EXTENDED GEOMETRY data, Sect. F11.4.4) was not defined by the UNIT ORIENTATION DATA entered in the ARRAY DATA (Sect. F11.4.5). Correct the data and resubmit the problem.

K5-219 ***ERROR***THE START DATA SPECIFIED ________ STARTING POINTS CHOSEN FROM A COSINE DISTRIBUTION BUT NONE WERE FOUND.

This message from subroutine START is printed if start type 2 was specified, and the code was unable to start any neutrons in the cuboid defined by XSM, XSP, YSM, YSP, ZSM, ZSP. Verify that fissile material exists within that cuboid. If it doesn't, respecify the starting cuboid to contain fissile material or choose a different start type. If message K5-105 states that only 0 independent starting points were generated, it indicates that the code was unable to start any neutrons in the spike specified by start type 2. Verify that the unit specified for the spike contains fissile material. If only a very small fraction of the volume of this unit is fissile, it may be necessary to input a larger value for the KENO V.a parameter "TMAX=" or choose a different start type. The problem will not be run if message K5-219 is printed.

K5-220 ERROR IN PICTURE DATA.
IF THE COORDINATE OF THE LOWER RIGHT-HAND CORNER IS ENTERED, ONE OF THE PLOT PARAMETERS DLX, DLD, NAX, OR NDN MUST BE ENTERED. CURRENT VALUES ARE LISTED BELOW. TITLE:

problem title is listed here

	UPPER LEFT COORDINATES	UPPER RIGHT COORDINATES
X	xu	xl
Y	yu	yl
Z	zu	zl

```
             U AXIS              V AXIS
             (DOWN)              (ACROSS)

        X      ud                  ua
        Y      vd                  va
        Z      wd                  wa

NDN=      0   NAX=    0      DLD= 0.0000E+00         DLX= 0.0000E+00
```

This message from subroutine RDPLOT indicates that the coordinates of the lower right-hand corner of the plot were specified in the input data without specifying one of the following plot parameters: (1) NDN, the number of characters down the page, (2) NAX, the number of characters across the page, (3) DLD, the vertical spacing between points, or DLX, the horizontal spacing between points. The problem will not be run. To correct the error, specify NDN, NAX, DLD, or DLX in the plot data and resubmit the problem. See Sects. F11.4.11 and F11.5.9 for assistance.

K5-221 *** ERROR *** IN INPUT DATA ERROR HAS BEEN ENCOUNTERED IN THE
___________ DATA ENTERED FOR THIS PROBLEM.

This message is printed from subroutine INITAL if a reading error is encountered in the KENO parameter data. It is printed from subroutine DATAIN if a reading error is encountered in any other data block. The name of the data block is printed in the message. This message is printed if the library routine LRDERR returns a value of "TRUE," indicating that a reading error has occurred. Locate the unnumbered message **** ERROR IN INPUT. CARD IMAGE PRINTED ON NEXT LINE *****. The third line of this message identifies the incorrect character. Correct the data and resubmit the problem.

K5-222 ***_____ TRANSFERS FOR MIXTURE _____ WERE CORRECTED FOR BAD MOMENTS.

This message from subroutine MAKANG indicates moments were corrected to eliminate negative probabilities for calculated angles. If the moment was changed by more than eps times the moment, a K5-60 or K5-64 message is printed. If eps is very small, any change without an accompanying message is trivial. The K5-60 or K5-64 messages can be used to determine if the affected transfer and correction are significant. Most messages are caused by the cross-sections being in single precision and the moments calculations being done in double precision. Generally, as the number of energy groups increase so does the number of corrected transfers for a given mixture.

K5-223 *** THE PLUS CUBE FACE IS NOT GREATER THAN THE MINUS CUBE FACE PLUS
FACE = _________ MINUS FACE = _________

This message from subroutine KENOG is printed if the data for a CUBE has the positive face not greater than the negative face. This will lead to negative thicknesses in each dimension for a CUBE. Correct the data making sure that the positive face is greater than the negative face.

K5-224 *** THE PLUS FACES OF A CUBOID ARE NOT GREATER THAN THE MINUS FACES
+X=_____ -X = _____ +Y = _____ -Y = _____ +Z = _____ -Z _____

This message from subroutine KENOG is printed if the data for one or more pairs of faces for a cuboid are out of order (i.e., the positive face is not greater than the negative face). Correct the data making sure the positive face is greater than the negative face.

K5-225 *** THE CYLINDER RADIUS IS NOT GREATER THAN ZERO, OR THE + HEIGHT IS NOT GREATER THAN THE - HEIGHT
RADIUS = _____ +H = _____ -H = _____

This message from subroutine KENOG is printed if the data for a cylinder has a radius that is not greater than zero, or the top of the cylinder is not specified as greater than the bottom of the cylinder. Correct the data making sure the radius is positive, and the top is greater than the bottom.

K5-226 *** THE SPHERE RADIUS IS NOT GREATER THAN ZERO - RADIUS = _____

This message from subroutine KENOG is printed if a sphere is specified with a radius that is not greater than zero. Correct making sure the radius is greater than zero.

K5-227 *** A REPLICATE THICKNESS IS LESS THAN ZERO
SURFACE NUMBER = _____ THICKNESS = _____

This message from subroutine KENOG is printed if a REPLICATE is specified with a negative thickness. Correct making sure that none of the thicknesses are less than zero.

K5-228 *** UNIT nn IS INVALID
UNITS MUST BE GREATER THAN 0. THE PREVIOUS UNIT NUMBER IS MM.
PRINTING THE GEOMETRY AS READ WILL BE TURNED ON.

This message from subroutine KENOG is printed if a unit number, nn, is read which is not greater than 0. This unit followed unit mm in the input (if mm is 1, the unit nn may be the first unit in the geometry). The geometry following this will be printed as read as an aid in locating the error. Correct making sure all unit numbers are greater than 0.

K5-229 *** THE NUMBER OF START TYPE 6 HISTORIES SPECIFIED DOES NOT MATCH THE NUMBER OF HISTORIES PER GENERATION *nnnnn* HISTORIES WILL BE RANDOMLY SELECTED FROM THE *mmmmm* HISTORIES SPECIFIED.

This message from subroutine START6 is printed if the number of starting points specified (***mmmmm***) is less than the number of histories per generation (***nnnnnn***).

F11.7.2 STOP CODES

The STOP codes that are encountered in KENO V are listed in tabular form, indicating the subroutine where they occur and the associated error message. A STOP is executed whenever a fatal error is recognized. Look up the associated message number to determine the appropriate corrective measures. A traceback may be generated whenever subroutine STOP is called to print a message. If no traceback is indicated in the STOP CODE table, a stop is executed as soon as the associated message is printed.

STOP NUMBER	SUBROUTINE	TRACEBACK	ASSOCIATED MESSAGE
20			See page F11.7.37
100	ARAYIN	Yes	K5-25
101	ARAYIN	No	K5-31
102	CRMAX	Yes	K5-135
103	CROS	No	K5-126
104	CROS	No	K5-127
105	WRTALB	No	K5-71
106	ICEMIX	No	K5-68
107	INDX	No	K5-124
108	INITAL	No	K5-2
109	INITAL	No	K5-2
110	MAKANG	Yes	K5-57
111	MIXCRS	Yes	K5-52
112	MIXER	Yes	K5-53
113	MIXER	Yes	K5-54
114	MIXIT	Yes	K5-15
115	NORM1D	No	K5-56
116	NSUPG	Yes	K5-79
117	NSUPG	No	K5-81
118	PARAM	No	K5-7 or K5-8
119	PRTLBA	No	K5-89
120	RDALB	No	K5-49
121	RDCALC	No	K5-116 or K5-146 or K5-147 or K5-177 or K5-178

STOP NUMBER	SUBROUTINE	TRACEBACK	ASSOCIATED MESSAGE
122	RDICE	No	K5-50
123	RDRST	No	K5-48
124	RDTAPE	No	K5-69
125	READGM	No	K5-20
126	SAVST6	Yes	K5-140
127	SRMAX	Yes	K5-136
128	START	Yes	K5-101
129	MASTER	No	K5-100
130	MASTER	No	K5-117 or K5-128
131	VOLFIS	No	K5-110
132	WRTICE	No	K5-72
133	WRTRST	No	K5-70
134	TRACK	No	K5-62
135	DATAIN	Yes	K5-142
136	DATAIN	Yes	K5-143
137	ARAYIN	No	K5-151
138	SORTA	Yes	K5-153
139	SORTA	No	K5-154
140	LODARA	No	K5-155
141	READGM	No	K5-157
142	SORTA	No	K5-159
143	HOLE	No	K5-160
144	FLDATA	No	K5-161
145	CRMIN	Yes	K5-167
146	SRMIN	Yes	K5-168
147	MASTER	Yes	K5-182
148	WAITIN	No	K5-185
149	WAITIN	No	K5-186
150	WATES	Yes	K5-22
151	RDBIAS	No	K5-184
152	DATAIN	No	K5-46
153	RCHRS	Yes	K5-195
154	RDCALC	No	K5-196
155	GEOMIN	Yes	K5-25
156	SORTA	Yes	K5-25
157	START	No	K5-202
158	START	No	K5-206
159	START	No	K5-207
160	START	No	K5-210
161	WAITIN	No	K5-211
162	KENO V.a	Yes	K5-213
163	RDCALC	No	K5-215

F11.7.3 MESSAGES ASSOCIATED WITH STOP 20 IN KENO V.a

The error messages that are associated with a STOP 20 in KENO V.a are listed below in numerical order. Look up the appropriate message number to determine corrective measures.

LMP001 DA ERROR - INVALID UNIT NUMBER. THE LOGICAL UNIT NUMBER IS ____.

This message from the subroutine library direct-access routines indicates that an invalid unit number was specified as a direct-access device. In KENO V.a, this message is indicative of a code error.

LMP002 DA ERROR - FORTRAN USING THIS UNIT. THE LOGICAL UNIT NUMBER IS _____.

This message from the subroutine library direct-access routines indicates that the specified unit number is open as a sequential dataset rather than a direct-access dataset. In KENO V.a, this error may be caused by entering a direct-access unit number for LIB= or XSC=.

LMP003 DA ERROR - DCB NOT OPEN. THE LOGICAL UNIT NUMBER IS _____.

This message from the subroutine library direct-access routines indicates that the program attempted to read or write on a direct-access device but the data control block was not open. In KENO V.a this message is indicative of a code error.

LMP004 DA ERROR - UNABLE TO OPEN DCB. THE LOGICAL UNIT NUMBER IS _____.

This message from the subroutine library direct-access routines indicates that the program is unable to open the data control block for the direct-access device. This message indicates that the job control language did not include proper specification of the above named unit.

LMP005 DA ERROR - RELATIVE BLOCK NOT IN DATA SET. RELATIVE BLOCK NUMBER IS __
__.

This message from the subroutine library direct-access routines indicates that the number of direct-access blocks is too small for the problem. Increase the number of direct-access blocks in the KENO V.a parameter data by entering the parameter NB8=nnn where nnn is larger. For example, if the relative block number is 201, try increasing the number of direct-access blocks to 300 or more.

LMP006 DA ERROR - INVALID BLOCK LENGTH. THE BLOCK LENGTH IS _____.

This message from the subroutine library direct-access routines indicates that the length of the direct-access blocks is invalid. A valid block length must be positive. This message is indicative of a code error.

LMP007 DA ERROR - DCB ALREADY OPEN. THE LOGICAL UNIT NUMBER IS _____.

This message from the subroutine library direct-access routines indicates that the data control block for the above named unit was previously opened and not closed.

LMP008 DA ERROR - PERMANENT I/O ERROR.

This message from the subroutine library direct-access routines indicates that a permanent input/output error has occurred.

F11.A ALPHABETICAL INDEX OF SUBROUTINES

This section provides a convenient alphabetical index of the subroutines used in KENO V.a, the subroutine that calls it, and a list of the subroutines it calls. FORTRAN-supplied library routines are contained in parentheses.

Subroutine Name	Calling Subroutine	Called Subroutine
adjust	holchk	move
albedo	track	azirn fltrn
albrd	difalb	albuse
albuse	albrd	inquir io rite
angles	prang	q
aralba	jomity	prtlba
arasiz	corsiz	lscan
arayin	datain	aread box clear iread rchrs rdbox stop yread
badmom	prang	
banker	guide	move
box	arayin	
boxc	loadit	
chkstr	guide	move trkwrt

Subroutine Name	Calling Subroutine	Called Subroutine
choose	start4 start5	fltrn
clrnit	rdplot	
cmprs	mixmix	
corre	master	clear inquir ratio reed rite
corsiz	jomity	arasiz
crmax	holext jomchk	crsprd dotprd stop vecadd vecnrm
crmin	holhol	crsprd dotprd stop vecadd vecnrm
cros	track	
crsprd	crmax crmin srmax srmin	
datain	master	arayin aread clear extra fldata geomin idx1d iowrt lrderr mixit

Subroutine Name	Calling Subroutine	Called Subroutine
		rdbias rdplot rdref rdrst rdstrt rt rtara savst6 stop wrtplt
deviat	kedit pltkef	
difalb	fldata	albrd
dotprd	crmax crmin srmax srmin xrmin	
editor	kedit	
extra	datain	
fil2d	fillsg	rd
fillsg	nsupg	clear fil2d inquir iset rd reed rite rt sgalb sgwt

Subroutine Name	Calling Subroutine	Called Subroutine
find	getmus	q
findbx	locate	
fisflx	guide	jstime looper matk statis
fitflx	master	ioleft prtflx rd
fldata	datain	difalb reed rgused rite sorta sortr stop wates
freak	master	erf
fsabed	kedit	
geomin	datain	clear kenog readgm stop
getmus	prang	find q
gocurs	volume	hunter
gtvols	volume	

Subroutine Name	Calling Subroutine	Called Subroutine
guide	master	banker chkstr clear fisflx indx inquir ioleft jstime nstart pull rd rdcalc reed reset rite start track wrtcal
holchk	jomchk	adjust holext holhol
hole	lodara sorta	clear stop
holext	holchk	crmax srmax xxlim
holhol	holchk	crmin srmin xrmin xxmin
hunter	gocurs volume	clear move
icemix	master	rdtape
idx1d	datain	iread

Subroutine Name	Calling Subroutine	Called Subroutine
indx	guide	
inital		aread ioleft jstime k5unit lrderr mesage opnfil param scanon
iosdun	jomity master	ioleft
ixalb	limln nsupg	
jll2	mixer	
jomchk	jomity	crmax holchk srmax xxlim
jomity	master	aralba corsiz iosdun jomchk loadit prtjom volume
kedit	master	clear deviat editor fsabed jstime looper matrix pltkef rndout

Subroutine Name	Calling Subroutine	Called Subroutine
kenog	geomin	aread iread rchrs rdorgn xxin xxina
ldwrt	track	
legend	prang	
limln	nsupg	ixalb
loadit	jomity	boxc clear lodalb lodara prtara reed
locate	mesh start	findbx locbox move posit
locbox	locate lodara sorta start track	
lodalb	loadit	reed
lodara	loadit	clear hole locbox reed
lodrgc	rgused	

Subroutine Name	Calling Subroutine	Called Subroutine
lodwts	master	clear prtwts rd reed
looper	fisflx kedit	reed rite
lscan	arasiz rgused start	
makang	mixer	clear prang reed stop
maktap	mixer	io reed scoot
master		clear corre datain fitflx freak guide icemix iosdun jomity jstime kedit lodwts mixer nsupg openda point prtplt stop wrtrst

Subroutine Name	Calling Subroutine	Called Subroutine
matk	fisflx matrix	clear
matrix	kedit	jstime labl matk prtsrc
mesh	print	clear locate posit
mix1d	mixmix	
mix2d	mixmix	mgcwrd
mix2m	mixmix	
mixcrs	mixer	clear stop
mixer	master	closda jll2 makang maktap mixcrs mixmix openda reed rite stop xlnths
mixit	datain	aread fread inquir iread lread rdmixt rite stop

Subroutine Name	**Calling Subroutine**	**Called Subroutine**
mixmix	mixer	cmprs io mix1d mix2d mix2m nnitl norm1d norm2d prtmix reed rite sumsct
nnitl	mixmix	clear inquir rite
norm1d	mixmix	
norm2d	mixmix	
nstart	guide	clear fltrn move sortbk
nsupg	master	fillsg freecr inquir ixalb limln openda point prtxs reed rite rtadj stop

Subroutine Name	Calling Subroutine	Called Subroutine
param	inital	aread clear fread iread opnfil rndin rndout timfac zread
pltkef	kedit	deviat
point	master nsupg	
posit	locate mesh	
prang	makang	angles badmom getmus inquir legend rd rite
print	prtplt	filnam mesh system
prt1ds	prtxs	
prt2ds	prtxs	
prtara	loadit	
prtflx	fitflx	
prtjom	jomity	
prtlba	aralba	

Subroutine Name	**Calling Subroutine**	**Called Subroutine**
prtmix	mixmix	sortmx
prtplt	master	clear inquir print reed relate untcrs
prtsrc	matrix	
prtwts	lodwts	
prtxs	nsupg	prt1d prt1ds prt2ds reed
q	angles find getmus	
ratio	corre	
rchrs	arayin kenog rdplot	aread getptr rcrdln rstptr stop
rcolor	rdplot	aread iread
rdalb	rdrst	clear inquir io rite
rdara	rdrst	inquir io rite

Subroutine Name	Calling Subroutine	Called Subroutine
rdbias	datain	aread fread io iread waitin
rdbox	arayin	iread
rdcalc	guide	io rdgrp rite rndin shufl stop
rdgrp	rdcalc	
rdice	rdrst	clear inquir io rite
rdmixt	mixit	
rdorgn	kenog	aread fread
rdplot	datain	aread clear clrnit fread io iread rchrs rcolor
rdref	datain	aread cread

Subroutine Name	Calling Subroutine	Called Subroutine
rdrst	datain	inquir io rdalb rdara rdice rdwts rite
rdstrt	datain	aread clear fread io iread opnfil
rdtape	icemix	clear inquir io rite xsec1d xtenda
rdwts	rdrst	inquir io rite
readgm	geomin	
relate	prtplt	clear
reset	guide	move
rgused	fldata	clear lodrgc lscan
rt	datain fillsg	inquir rite
rtadj	nsupg	

Subroutine Name	Calling Subroutine	Called Subroutine
rtara	datain	clear inquir rite
savst6	datain	clear inquir io move rite stop
scoot	maktap	
sgalb	fillsg	inquir rd reed
sgwt	fillsg	rd
shufl	rdcalc	
sorta	fldata	clear hole locbox reed stop
sortbk	nstart	gtiso move
sortmx	prtmix	
sortr	fldata	
srmax	holext jomchk	crsprd dotprd stop vecadd vecnrm

Subroutine Name	**Calling Subroutine**	**Called Subroutine**
srmin	holhol	crsprd dotprd stop vecadd vecnrm
start	guide	clear fltrn gtiso jstime locate locbox lscan move start0 start1 start2 start3 start4 start5 start6 stop strtsu volfis
start0	start	fltrn
start1	start start2	fltrn
start2	start	start1 start5
start3	start	
start4	start	choose
start5	start start2	choose fltrn
start6	start	fltrn move reed

Subroutine Name	Calling Subroutine	Called Subroutine
statis	fisflx	
strtsu	start	azirn fltrn gtiso sflra
sumsct	mixmix	
track	guide	albedo azirn clear cros exprn fltrn gtiso ldwrt locbox move sflra trkwrt
trkwrt	chkstr track	rndout
untcrs	prtplt	clear
vecadd	crmax crmin srmax srmin xrmin	
vecdif		
vecnrm	crmax crmin srmax srmin	
volfis	start	

Subroutine Name	**Calling Subroutine**	**Called Subroutine**
volume	jomity	clear gocurs gtvols hunter move
waitin	rdbias	inquir io rite
wates	fldata	inquir io move reed rite stop
wrtalb	wrtrst	inquir io reed
wrtara	wrtrst	io reed
wrtcal	guide	io reed rndout wrtgrp
wrtgrp	wrtcal	
wrtice	wrtrst	inquir io reed
wrtplt	datain	inquir io rite

Subroutine Name	**Calling Subroutine**	**Called Subroutine**
wrtrst	master	inquir io reed rndout wrtalb wrtara wrtice wrtwts
wrtwts	wrtrst	io reed
xlnths	mixer	
xrmin	holhol	dotprd vecadd
xsec1d	rdtape	
xxina	kenog	fread iread
xxlim	holext jomchk	
xxmin	holhol	

F11.B ALPHABETICAL INDEX OF COMMONS

This section lists the labeled common blocks used in KENO V.a and an alphabetical listing of the subroutines that reference them.

Common Name	Referencing Subroutine
albdat	albedo
	albrd
	albuse
	corre
	datain
	difalb
	fillsg
	fldata
	guide
	kedit
	limln
	loadit
	master
	nsupg
	param
	point
	rdalb
	rdref
	rdrst
	sgalb
	track
	wrtalb
	wrtrst
albnam	albedo
	albrd
	albuse
	corre
	datain
	difalb
	fillsg
	fldata
	guide
	kedit
	limln
	loadit
	master

Common Name	Referencing Subroutine
	nsupg
	param
	point
	rdalb
	rdref
	rdrst
	sgalb
	track
	wrtalb
	wrtrst
angle	angles
	badmom
	find
	getmus
	legend
	makang
	prang
blklnc	fitflx
	guide
	master
	nsupg
	param
	rdtape
dimen	arasiz
	arayin
	box
	boxc
	corre
	corsiz
	datain
	editor
	fil2d
	fillsg
	findbx
	fisflx
	fitflx
	fldata
	geomin
	guide
	holchk
	icemix
	inital

Common Name	Referencing Subroutine
	jomchk
	jomity
	kedit
	kenog
	limln
	locate
	locbox
	lodara
	looper
	master
	matrix
	mesh
	mixer
	mixit
	nstart
	nsupg
	param
	point
	posit
	print
	prtjom
	prtplt
	rdbias
	rdbox
	rdcalc
	rdgrp
	rdice
	rdref
	rdrst
	rdstrt
	rdtape
	readgm
	reset
	rgused
	rtara
	sorta
	sortr
	start
	statis
	track
	volfis
	volume

Common Name	Referencing Subroutine
	wrtcal
	wrtgrp
	wrtice
	wrtrst
	xsec1d
drtacs	albuse
	corre
	datain
	fil2d
	fillsg
	fisflx
	fitflx
	fldata
	guide
	icemix
	inital
	loadit
	lodara
	lodwts
	looper
	master
	mixer
	mixit
	nsupg
	prang
	prtplt
	rdalb
	rdara
	rdcalc
	rdice
	rdrst
	rdtape
	rdwts
	rt
	rtara
	savst6
	sgalb
	sgwt
	sorta
	start6
	waitin
	wates

Common Name	Referencing Subroutine
	wrtalb
	wrtcal
	wrtice
	wrtplt
	wrtrst
	wrtwts
errflg	arayin
	inital
	prtlba
final	editor
	fisflx
	guide
	inital
	kedit
	master
	matrix
	start
	statis
	track
lifetm	editor
	fisflx
	guide
	kedit
	master
	rdcalc
	statis
	track
	wrtcal
logic	albrd
	arasiz
	arayin
	chkstr
	corre
	corsiz
	datain
	editor
	fil2d
	fillsg
	fisflx
	fitflx
	fldata

Common Name	**Referencing Subroutine**
	guide
	icemix
	idx1d
	inital
	jomchk
	jomity
	kedit
	kenog
	limln
	locate
	looper
	master
	matrix
	mesh
	mixer
	nsupg
	param
	point
	posit
	prtjom
	prtplt
	prtxs
	ratio
	rdbias
	rdbox
	rdcalc
	rdgrp
	rdref
	rdrst
	rdtape
	readgm
	reset
	rtara
	sorta
	sortr
	start
	statis
	track
	volfis
	volume
	wates
	wrtcal
	wrtgrp
	wrtrst

Common Name	**Referencing Subroutine**
lowbnd	albedo
	corsiz
	fillsg
	fisflx
	gocurs
	guide
	hunter
	jomchk
	kedit
	locate
	lodara
	mesh
	point
	print
	prtjom
	prtplt
	reset
	start
	track
	volume
lpnt	master
	start6
matrx	fldata
	matrix
	nsupg
	reset
	sorta
	track
nutron	albedo
	chkstr
	choose
	cros
	findbx
	fldata
	guide
	locate
	mesh
	nstart
	reset
	shufl
	start

Common Name	Referencing Subroutine
	start0
	start1
	start2
	start3
	start4
	start5
	start6
	strtsu
	track
	trkwrt
pcolor	clrnit
	mesh
	print
	prtplt
	rdplot
	rdrst
	relate
	wrtrst
pict	clrnit
	mesh
	print
	prtplt
	rdplot
	rdrst
	relate
	wrtrst
picttl	clrnit
	mesh
	print
	prtplt
	rdplot
	rdrst
	relate
	wrtrst
pointr	datain
	fillsg
	icemix
	jomity
	kedit
	loadit
	master
	nsupg

Common Name	Referencing Subroutine
	point
	rdcalc
	wrtcal
runtyp	datain
	inital
	param
stdata	chkstr
	datain
	rdrst
	rdstrt
	start
	start0
	start1
	start2
	start3
	start4
	start5
	start6
	volfis
	wrtrst
titl	arasiz
	corsiz
	datain
	editor
	fitflx
	freak
	geomin
	guide
	holchk
	holext
	holhol
	icemix
	inital
	jomchk
	kedit
	kenog
	master
	matrix
	nsupg
	param
	point

Common Name	Referencing Subroutine
	prtjom
	prtlba
	prtmix
	prtplt
	prtwts
	prtxs
	rdbox
	rdplot
	rdtape
	readgm
	track
	volume
	wrtrst
unit	albedo
	albrd
	albuse
	angles
	arasiz
	arayin
	box
	chkstr
	corre
	corsiz
	cros
	datain
	difalb
	editor
	fil2d
	fillsg
	find
	fisflx
	fitflx
	fldata
	freak
	geomin
	getmus
	guide
	icemix
	idx1d
	indx
	inital
	jomchk
	jomity

Common Name	Referencing Subroutine
	kedit
	kenog
	ldwrt
	limln
	loadit
	lodara
	lodwts
	looper
	makang
	maktap
	master
	matk
	matrix
	mixer
	mixit
	mixmix
	nnitl
	nstart
	nsupg
	param
	point
	posit
	prang
	print
	prtara
	prtflx
	prtjom
	prtlba
	prtplt
	prtsrc
	prtxs
	ratio
	rdalb
	rdbias
	rdbox
	rdcalc
	rdgrp
	rdice
	rdplot
	rdref
	rdrst
	rdstrt
	rdtape
	rdwts

Common Name	Referencing Subroutine
	readgm
	rt
	rtara
	savst6
	sgalb
	sgwt
	shufl
	sorta
	sortr
	start
	start2
	start6
	statis
	strtsu
	track
	trkwrt
	volfis
	volume
	waitin
	wates
	wrtalb
	wrtcal
	wrtgrp
	wrtice
	wrtrst
	wrtwts

F11.C KENO V.a INPUT SUMMARY

This appendix consists of a summary of the KENO V.a input data requirements.

Table F11.C.1. Summary of parameter data

TITLE: The title must be entered first (80 columns) See Sect. F11.4.3

PARAMETERS: Format: READ PARAM enter parameter data here END PARAM
If parameters are entered, they must follow the title. See Sects. F11.4.3, F11.5.2, and F11.5.3.

KEY	STD.	DEFINITION	KEY	STD.	DEFINITION	KEY	STD.	DEFINITION	KEY	STD.	DEFINITION
RND=	given	random number	RUN=	YES	execute problem	MKH=	NO	matrix by hole	XSC=	14	mixed xsecs
TME=	30 min	execution time (min)	FLX=	NO	fluxes	CKH=	NO	cofactor k by hole	ALB=	79	albedo
TBA=	0.5 min	batch time (min)	FDN=	NO	fission densities	FMH=	NO	fiss. prod. by hole	WTS=	80	weights
WTA=	0.5	average weight	ADJ=	NO	adjoint calculation	HHL=	NO	MKH at highest level	LIB=	0	working xsecs
WTH=	3.0	wt. for splitting	AMX=	NO	all mixture xsecs	MKA=	NO	matrix by array	SKT=	16	scratch
WTL=	1/WTH	Russian Roulette wt.	XAP=	NO	xsec angles & probs.	CKA=	NO	cofactor k by array	RST=	0	read restart
GEN=	103	no. of generations	XS1=	NO	1-D xsecs	FMA=	NO	fiss. prod. by array	WRS=	0	write restart
NPG=	300	no. per generation	XS2=	NO	2-D xsecs	HAL=	NO	MKA at highest level			
NSK=	3	generations skipped	PKI=	NO	fission spectrum	PLT=	YES	printer plots			
RES=	0	gens. between restart	P1D=	NO	extra 1-D xsecs	BUG=	NO	debug print			
NBK=	NPG+25	neutron bank	FAR=	NO	fiss. & abs.	TRK=	NO	print neutron tracks			
XNB=	0	positions	MKP=	NO	matrix by location	PWT=	NO	print avg. weight			
NFB=	NPG	extra bank entries	CKP=	NO	cofactor k by loc.	PGM=	NO	unprocessed			
XFB=	0	fission bank positions	FMP=	NO	fiss. prod. by loc.	SMU=	NO	geometry			
X1D=	0	extra bank entries	MKU=	NO	matrix by unit	NUB=	NO	self-multiplication			
LNG=	1000000	no. of extra 1-D's	CKU=	NO	cofactor k by unit	PAX=	NO	neutrons per fission			
BEG=	1	words of storage	FMU=	NO	fiss. prod. by unit			albedo-xsec array			
NG8=	200	restart at this gen.									
NL8=	512	blocks for d.a. unit									
		length of d.a. block									

Table F11.C.2 Summary of array data

ARRAY Format: READ ARRAY array parameters data type orientation data END ARRAY See Sects. F11.5.5, F11.5.6, and F11.5.7.

Repeat the sequence ARRAY PARAMETERS DATA TYPE ORIENTATION DATA for each array used in the problem.

ARRAY PARAMETERS		
KEYWORD	DEFAULT	DEFINITION
ARA=	1	no. defining the array
NUX=	1	no. of units in X direction
NUY=	1	no. of units in Y direction
NUZ=	1	no. of units in Z direction
GBL=	maxara	global, universe, or overall array number**
COM=	none	delim comment delim optional comment is a maximum of 132 characters

**Can be defaulted by the code. If specified, it need be entered only once per problem.

DATA TYPE
FILL LOOP

ORIENTATION DATA FOR FILL

Enter unit numbers to define every position in the array. When entering data utilizing the options in this table, the count field and option field must be adjacent with no imbedded blanks. The operand field may be separated from the option field by one or more blanks. Orientation data for FILL is terminated by entering END FILL.

COUNT FIELD	OPTION FIELD	OPERAND FIELD	COMMENTS
		j	stores j at the current position in the array
i	R	j	stores j in the next i positions in the array
i	*	j	stores j in the next i positions in the array
i	$	j	stores j in the next i positions in the array
	F	j	fills remainder of the array with unit no. j starting with the current array position
	A	j	sets the current position int he array to j
i	S		increments current position in the array by i allows skipping 1 positions. The value of i may be positive or negative
i	Q	j	repeats the previous j entries i times. The default value of i is 1
i	N	j	repeats the previous j entries i times, inverting the sequence each time. The default value of i is 1
i	B	j	starting with the entry at -i from the current position, store entries in inverse order until position -(i+j) is reached. Default value of i=1
i	P	j	alternately stores j and -j in the next i positions of the array
i	I	j k	provides the end points, j and k, with i entries linearly interpolated between them (i.e., a total of i+2 points). At least one blank must separate j and k. When used for an integer array, the I option should only be used to generate integer steps (i.e., (k-j)/(i+1) should be a whole number.
	T		terminates the data reading for the array

ORIENTATION DATA FOR LOOP

Enter the unit number and nine numbers that define the position(s) of that unit.

Data for each of these ten entries are repeated until every position in the array has been defined. Orientation data for LOOP is terminated by entering END LOOP.

ENTER DATA IN THE FORM:

DATA ENTRY	COMMENTS
LTYPE	The unit or box type. LTYPE must be greater than 1.
IX1	Starting position in the X direction. IX1 must be at least 1 and no larger than the value entered for NUX.
IX2	Ending position in the X direction. IX2 must be at least 1 and no larger than the value of NUX.
INCX	The number of units by which increments are made in the X direction.
IY1	The starting position in the Y direction. IY1 must be at least 1 and less than the value entered for NUY.
IY2	Ending position int he Y direction. IY2 must be at least 1 and no larger than the value of NUY.
INCY	The number of units by which increments are made in the positive Y direction.
IZ1	Starting position in the Z direction. IZ1 must be at least 1 and no larger than NUZ.
IZ2	Ending position in the Z direction. IZ2 must be at least 1 and no larger than NUZ.
INCZ	The number of units by which increments are made in the positive Z direction.

Table F11.C.3. Summary of biasing data

BIAS (weighting)

Format: READ BIAS keyword correlation data auxiliary END BIAS
See Sects. F11.4.7 and F11.5.8

KEYWORD	DESCRIPTION	
ID=	CORRELATION DATA will be read next.	
	id	material ID. Enter from table at right to use weighting data from the library
	ibgn	beginning bias ID
	iend	ending bias ID
WT=	AUXILIARY DATA will be read next.	
WTS=	AUXILIARY DATA will be read next.	
	wttitl	material title (12-character maximum)
	id	material ID
	nsets	number of sets of group structures
	REPEAT	(THKINC, NUMINC, NGPWT, WTAVG) NSETS TIMES
	thkinc	thickness per increment
	numinc	number of increments
	ngpwt	number of energy groups for this set of wts
	wtavg	enter numinc* ngpwt values of wtavg

MATERIAL	ID	ENERGY GROUPS	THICKNESS/ INCREMENT
concrete	301	16,27,123	5 cm
paraffin	400	16,27,123	3 cm
water	500	16,27,123,218	3 cm
graphite	6100	16,27,123	20 cm

For CORRELATION DATA, the material ID is chosen from material ID column above (the keyword is ID=).
For AUXILIARY DATA, the material ID is chosen by the user and the keyword is WT= or WTS=.
When AUXILIARY DATA are entered, CORRELATION DATA must also be entered to use the data.
Beginning and ending bias ID's are defined by the user. The geometry specificaiton that has the bias ID equal to the beginning bias ID utilizes the wt avg's from the first interval of material ID.

Table F11.C.4 Summary of boundary condition data

BNDS
(albedo or
boundary
conditions)

Format: READ BNDS face code albedo name END BNDS
See Sect. F11.4.7

The sequence FACE CODE ALBEDO NAME is entered as many times as necessary to define the appropriate albedo boundary conditions. The default for all faces is vacuum.

FACE CODES FOR ENTERING BOUNDARY (ALBEDO) CONDITIONS

FACE CODE	DEFINITION	FACE CODE	DEFINITION	FACE CODE	DEFINITION	FACE CODE	DEFINITION
+XB=	positive X face	XFC=	both X faces	+YX=	positive X and Y faces	&ZY=	positive Y and Z faces
&XB=	positive X face	YFC=	both Y faces	&XY=	positive X and Y faces	-XY=	negative X and Y faces
-XB=	negative X face	ZFC=	both Z faces	&YX=	positive X and Y faces	=XZ=	negative X and Z faces
+YB=	positive Y face	+FC=	all positive faces	+XZ=	positive X and Z faces	=YZ=	negative Y and Z faces
&YB=	positive Y face	&FC=	all positive faces	+ZX=	positive X and Z faces	YXF=	all X and Y faces
-YB=	negative Y face	-FC=	all negative faces	&XZ=	positive X and Z faces	ZXF=	all X and Z faces
+ZB=	positive Z face	XYF=	all X and Y faces	&ZX=	positive X and Z faces	ZYF=	all Y and Z faces
&ZB=	positive Z face	XZF=	all X and Z faces	+YZ=	positive Y and Z faces	-YX=	negative X and Y faces
-ZB=	negative Z face	YZF=	all Y and Z faces	+ZY=	positive Y and Z faces	-ZX=	negative X and Z faces
ALL=	all 6 faces	+XY=	positive X and Y faces	&YZ=	positive Y and Z faces	-ZY=	negative Y and Z faces

ALBEDO NAMES AVAILABLE ON THE KENO V ALBEDO LIBRARY, FOR USE WITH THE FACE CODES

ALBEDO NAME	DESCRIPTION	ALBEDO NAME	DESCRIPTION	ALBEDO NAME	DESCRIPTION
DPOH20	12-in. double PO water	CONC-4	4-in. concrete differential	VACUUM	vacuum condition
DPOH20	differential albedo with	CON4	albedo with 4 incident angles	VOID	
DPO	4 incident angles	CONC4		VACU	
DPO				VAC	
		CONC-8	8-in. concrete differential		
H20	12-in. water differential	CON8	albedo with 4 incident angles	SPECULAR	mirror image reflection
WATER	albedo with 4 incident angles	CONC8		MIRROR	
				MIRR	
PARAFFIN	12-in. paraffin differential	CONC-12	12-in. concrete differential	SPEC	
PARA	albedo with 4 incident angles	CON12	albedo with 4 incident angles	SPE	
WAX		CONC12		MIR	
CARBON	200-cm carbon differential	CONC-15	16-in. concrete differential	PERIODIC	periodic boundary condition
GRAPHITE	albedo with 4 incident angles	CON16	albedo with 4 incident angles	PERI	
C		CONC16		PER	
ETHYLENE	12-in. polyethylene	CONC-24	24-in. concrete differential		
POLY	differential albedo with	CON24	albedo with 4 incident angles		
CH2	4 incident angles	CONC24			

Table F11.C.5. Summary of geometry data

GEOMETRY (region)

Format: READ GEOM enter geometry region data here END GEOM
See Sects. F11.4.4, F11.5.1.2, F11.5.6, and F11.5.7.

GEOMETRY REGION DATA consists of SIMPLE GEOMETRY REGION DATA and EXTENDED GEOMETRY REGION DATA.

ENTER GEOMETRY REGION DATA IN THE FOLLOWING FORM:

OPTIONAL GLOBAL SPECIFICATION
UNIT n
OPTIONAL GEOMETRY COMMENT
SIMPLE GEOMETRY REGION DATA
and/or
EXTENDED GEOMETRY REGION DATA
* ENTER SIMPLE
GEOMETRY REGION DATA IN THE FOLLOWING FORM:

GLOBAL Enter only to specify this unit as the global unit.
UNIT n
COM=delim comment delim This optional comment can be up to 132 characters.
It must begin and end with a delimiter.
fgeom mix no. bias ID dimensions optional origin data (ORIGN coordinates) optional chord dat CHOD)

Enter as many geometry descripton specifications as necessary to describe the unit and as amny units as necessary to describe the system.

SIMPLE GEOMETRY REGION INPUT DATA REQUIREMENTS

TYPE OF DATA	TYPE 1 DATA	TYPE 2 DATA	TYPE 3 DATA	TYPE 4 DATA	TYPE 5 DATA	TYPE 6 DATA
fgeom	SPHERE HEMISPHERE HEMISPHE+X HEMISPHE-X HEMISPHE+Y HEMISPHE-Y HEMISPHE+Z HEMISPHE-Z	XCYLINDER XHEMICYL+Y XHEMICYL-Y XHEMICYL+Z XHEMICYL-Z	YCYLINDER YHEMICYL+X YHEMICYL-X YHEMICYL+Z YHEMICYL-Z	CYLINDER ZCYLINDER ZHEMICYL+X ZHEMICYL-X ZHEMICYL+Y ZHEMICYL-Y	CUBE	CUBOID
dimensions	R (radius)	R +H -H	R +H -H	R +H -H	+X -X	+X -X +Y -Y +Z -Z
optional origin coord.*	Enter the X Y Z coord. of origin	Enter the Y Z coord. of centerline	Enter the X Z coord. of centerline	Enter the X Y coord. of centerline	omit	omit
optional chord data**	Enter the dist. to plane	Enter the dist. to plane	Enter the dist. to plane	Enter the dist. to plane	omit	omit

Table F11.C.5 (continued)

GEOMETRY (region) (cont.)

ENTER EXTENDED GEOMETRY DATA IN THE FOLLOWING FORM:

fgeom ref. ID bias ID thickness per region origin coordinates nreg

EXTENDED GEOMETRY REGION INPUT DATA REQUIREMENTS

TYPE OF DATA	TYPE 1 DATA	TYPE 2 DATA	TYPE 3 DATA
fgeom	ARRAY CORE COREBDY COREBNDS COREBOUN	HOLE	REPLICATE REFLECTOR
ref. ID	array no.	emplaced unit number	mixture no. in generated regions
bias ID	omit for ARRAY	omit	first bias ID
thick./reg.	omit	omit	variable*
origin coord.	Enter the X Y Z coord. of most neg. pt. of array	Enter the X Y Z coord. of origin	omit
nreg	omit	omit	no. of regions to be generated

*The number of dimensions to be entered is the same as the region preceding the replicate or reflector specification because the generated regions have that shape. The value of the dimensions is the thickness of each generated region of material on that surface.

Table F11.C.6. Summary of mixing table data

MIXTURES

Format: READ MIXT xsec parameters END MIXT
These data are entered only if an AMPX working format library is being used. (LIB=) in the parameter data. Do not enter if an ICE mixed library is used, (XSC=) in the parameter data. See Sects. F11.4.10 and F11.5.5.

XSEC PARAMETERS consists of keywords and associated values.
These parameters, if entered, need be entered only once.

KEYWORD	DEFAULT	DEFINITION
SCT=	1	no. of discrete scattering angles 0 is isotropic 1 is P1 2 is P3 3 is P5
EPS=	.00003	cross-section message cutoff value use to suppress message no. K5-60

MIXING TABLE DATA consists of

(1) a keyword and mixture ID for the mixture
The keyword is MIX=
The desired mixture number follows the keyword.
(2) nuclide ID**
(3) number density**

**The sequence (2) (3) is repeated for each nuclide in the mixture.

REPEAT the sequence (1) (2)'s (3)'s until all the mixtures have been described.

Table F11.C.7 Summary of plot data

PLOT

Format: READ PLOT plot parameters END PLOT plot parameters must be entered for each plot that is to be made.
See Sects. F11.4.1 and F11.5.9

KEYWORD	DEFAULT	DEFINITION
TTL=	prob. title	delim ptitl delim delim is a one-character delimiter that signals the beginning and end of the title. ptitl is the plot title (max. 132 char.)
PIC=	MAT	Type of plot: MIXTURE, UNIT NO. or BIAS ID NO. MIXTURE ——— MAT MIX MIXT MIXTURE MEDI MEDIA UNIT NO. ---------- BOX BOXT BOXTYPE UNT UNIT UNITTYPE BIAS ID NO. ------ IMP BIAS BIASID WTS WEIG WEIGHTS WGT WGTS
XUL=	prev. plot	X coord. of upper left comer of plot
YUL=	prev. plot	Y coord. of upper left comer of plot
ZUL=	prev. plot	Z coord. of upper left comer of plot
XLR=	prev. plot	X coord. of lower right comer of plot
YLR=	prev. plot	Y coord. of lower right comer of plot
ZLR=	prev. plot	Z coord. of lower right comer of plot

KEYWORD	DEFAULT	DEFINITION
UAX=	prev. plot 0 If VAX OR WAX is read	X component of direction cosine for the AX axis of the plot (across)
VAX=	prev. plot 0 IF UAX OR WAX is read	Y component of direction cosine for the AX axis of the plot (across)
WAX=	prev. plot 0 IF UAX OR VAX is read	Z component of direction cosine for the AX axis of the plot (across)
UDN=	prev. plot 0 IF VDN OR WDN is read	X component of direction cosine for the DN axis of the plot (down)
VDN=	prev. plot 0 IF UDN OR WDN is read	Y component of direction cosine for the DN axis of the plot (down)
WDN	prev. plot 0 IF UDN OR VDN is read	Z component of direction cosine for the DN axis of the plot (down)
DLX=		Horizontal spacing between points on plot
DLD=		Vertical spacing between points on plot
NAX=		No. of intervals to be printed across page
NDN=		No. of intervals to be printed down page
LPI=	8.0 (character plots) 10 (color plots)	Vertical to horizontal scaling factor for plot proportionality.
RUN=	YES	YES allows the problem to execute NO terminates problem after data checking
PLT=	YES	YES allows the plot(s) to be made NO allows reading the plot data without making a plot
SCR=	YES	Display plot method SCR=YES utilizes color plot display SCR=NO utilizes printer plot display
NCH=	CHRS*	delim CHRS delim a one character delimiter signals the beginning and end of the character string
CLR=	Table F11.4.7	num(i) red(num(i)) green (num(i)) blue (num(i)) num(i) defines mix. no., unit no., or bias ID next 3 entries define red, green, and blue components of the color representing num(i).

PLOT ORIGIN:

(1) SINGLE UNIT - coincides with origin of geometry description.

(2) BASE ARRAY - at the most negative point of the global array

(3) REFLECTED ARRAY - coincides with the origin of the CORE or ARRAY description of the global array.

*default values of CHRS are given below:

MEDIA 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
CHRS 1 2 3 4 5 6 7 8 9 A B C D E F G H I J K L M

MEDIA 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42
CHRS N O P Q R S T U V W X Y Z # , $ - +) |

MEDIA 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58
CHRS & > ; ; . - % * " = ! (@ < / 0

Table F11.C.8. Summary of starting data

START Format: READ START enter start data here END START
The default value of start type is zero. See Sect. F11.4.8.

START TYPE	REQUIRED DATA	OPTIONAL DATA	STARTING DISTRIBUTION
0	none	NST XSM XSP YSM YSP ZSM ZSP RFL PSP	uniform
1	NST	XSM XSP YSM YSPO ZSM ZSP RFL PSP	cosine
2	NST NXS NYS NZS FCT	XSM XSP YSM YSP ZSM ZSP RFL PSP	cosine with fraction in specified unit

START TYPE	REQUIRED DATA	OPTIONAL DATA	STARTING DISTRIBUTION
3	NST TFX TFY TFZ NXS NYS NZS	KFS PSP	spike
4	NST TFX TFY TFZ NBX	KFS PSP	multiple spikes
5	NST NBX	PSP	in specified units
6	NST TFX TFY TFZ LNU*	NXS NYS NZS KFS PS6 PSP	arbitrary points

*LNU must be the last entry for each set of start 6 data. The LNU of each successive set of data must be larger than the last.

KEYWORD	DEFAULT	DEFINITION
NST=	0	start type
TFX=	0.0	X coordinate
TFY=	0.0	Y coordinate
TFZ=	0.0	Z coordinate
NXS=	0	X index of unit pos.
NYS=	0	Y index of unit pos.
NZS=	0	Z index of unit pos.
KFS=		fission spectra
LNU=	0	number of last neutron
NBX=	0	source unit number
FCT=	0	fraction
XSM=	-X	-X of source cuboid
XSP=	+X	+X of source cuboid
YSM=	-Y	-Y of source cuboid
YSP=	+Y	+Y of source cuboid
ZSM=	-Z	-Z of source cuboid
ZSP=	+Z	+Z of source cuboid
RFL=	NO	start in reflector
PS6=	NO	print start 6 input
PSP=	NO	print starting points

F11.D SAMPLE PROBLEMS

This section contains sample problems to demonstrate some of the options available in KENO V.a. All the options of KENO IV except generalized geometry and the search capability are available in KENO V.a. A search can be performed using the CSAS4 sequence (see Sect. C4 of the SCALE manual). KENO V.a does not run stacked cases. However, when KENO V.a is run as a part of SCALE, the driver allows KENO V.a to be executed each time it encounters an "=KENO5." Therefore, these KENO V.a problems are set up to run in SCALE. If the problems are to be run without using the SCALE driver, remove the "=KENO5" specification and the "END" specification after the "END DATA" in each sample problem.

F11.D.1 KENO V.a SAMPLE PROBLEM DATA

A brief problem description and the associated input data are included for each KENO V.a sample problem. Different options may be easily activated by making changes in the data. These problems are set up using a 44-group AMPX working format library which is specified on Unit 4 of the job control language. The nuclide identifiers for this library are consistent with the SCALE identifiers created by CSAS. Input data to create this library are given in Sect. F11.D.2. The unit number is defined by the parameter LIB= in the parameter data.

SAMPLE PROBLEM 1   2C8 BARE

This is a simple 2 × 2 × 2 array of uranium metal cylinders as described in the article, "Critical Three-Dimensional Arrays of U(93.2)-Metal Cylinders,"[1] by J. T. Thomas. This critical experiment is designated in Table II of that article as cylinder index 11 and reflector index 1. Figure F11.D.1 shows the critical experiment.

INPUT DATA

```
#kenova
sample problem 1  case 2c8 bare
read parameters
  flx=yes fdn=yes far=yes gas=no lib=4
  end parameters
read mixt  sct=2
mix=1 1092234 4.82716e-04 1092235 4.47971e-02 1092236 9.57231e-05
1092238 2.65767e-03
end mixt
read geometry
cylinder 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
end geometry
read array  nux=2 nuy=2 nuz=2  end array
end data
end
```

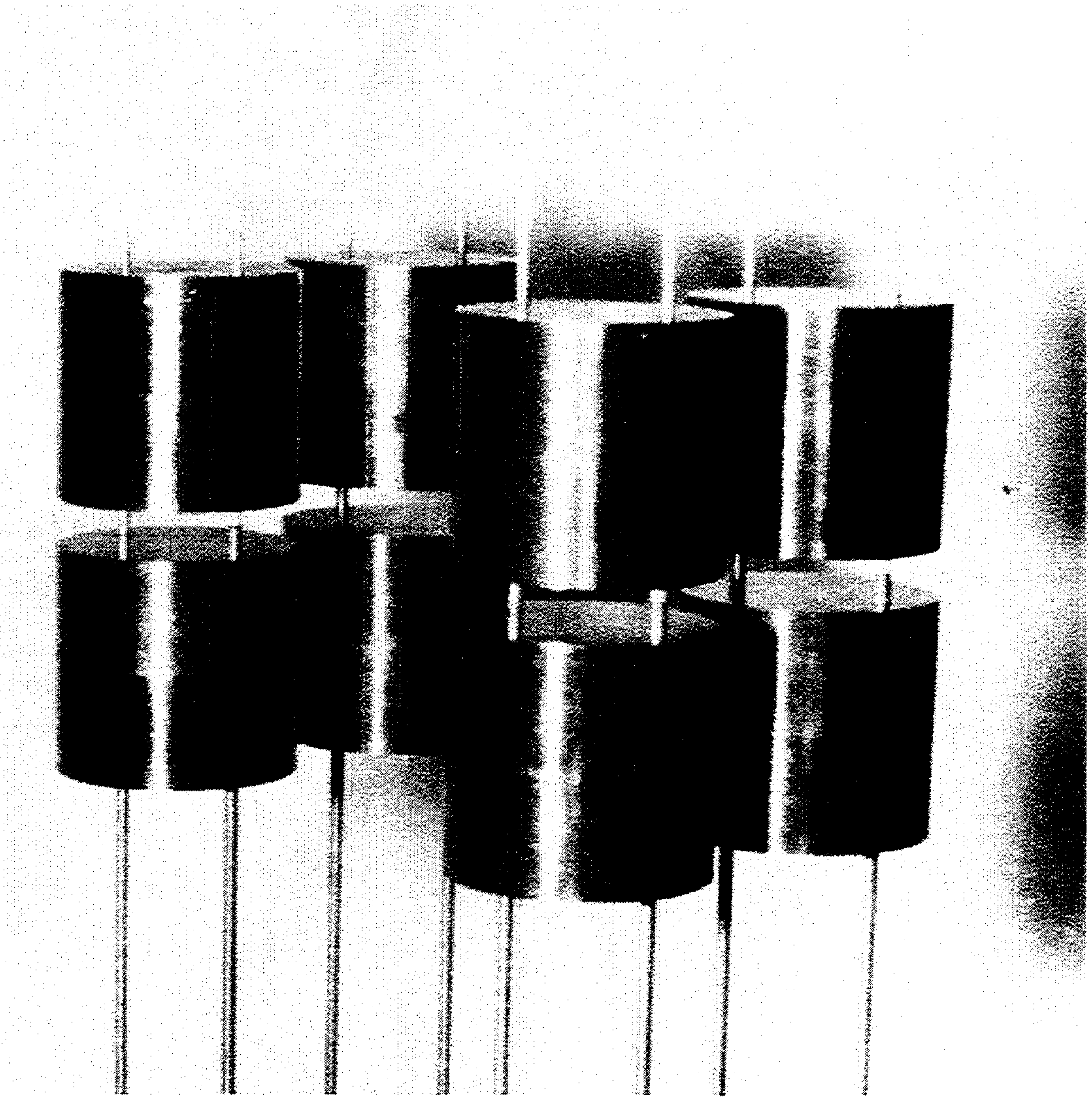


Figure F11.D.1 Critical 2C8 bare assembly

SAMPLE PROBLEM 2 CASE 2C8 BARE WITH 8 UNIT TYPES MATRIX CALCULATION

This problem is the same as sample problem 1 except it is set up as a mixed box problem with each unit of the array defined as a different unit type. Matrix k-effectives will be calculated for this problem by both unit type and array position. The print flags are set to print all matrix data.

INPUT DATA

```
#kenova
sample problem 2  2c8 bare with 8 unit types matrix calculation
read param
  lib=4  flx=yes fdn=yes
mku=yes fmu=yes mkp=yes fmp=yes
end param  read geometry  unit 1
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 2
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 3  cylinder  1 1 5.748 5.3825 -5.3825 cuboid  0 1 6.87 -6.87 6.87
-6.87 6.505 -6.505
box type 4  cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
box type  5
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 6
cylinder  1 1 5.748 5.3825 -5.3825  cuboid  0 1 6.87 -6.87 6.87 -6.87
6.505 -6.505 unit 7  cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 8
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505 end geom
read mixt  sct=2
mix=1 1092234 4.82716e-04 1092235 4.47971e-02 1092236 9.57231e-05
1092238 2.65767e-03
end mixt
read array  nux=2 nuy=2 nuz=2  loop  10*1 3*2 7*1  3 1 1 1 2 2 1
1 1 1  4 2 2 1 2 2 1 1 1 1  5 6*1 2 2 1  6 2 2 1 1 1 1 2 2 1
7  1 1 1 2 2 1 2 2 1  8 2 2 1 2 2 1 2 2 1 end array
end data
end
```

SAMPLE PROBLEM 3 2C8 15.24-CM PARAFFIN REFL

A 2 × 2 × 2 array of uranium metal cylinders is reflected by 6 in. of paraffin on all faces (Fig. F11.D.1). This critical experiment[1] is designated as cylinder index 11 and reflector index 5 in Table II of Ref. 1. Figure F11.D.2 shows half of the critical experiment, which consisted of the half shown and the mirror image of it. These two assemblies were moved together to achieve criticality. The top reflector is missing in Fig. F11.D.2, but was present when criticality was achieved.

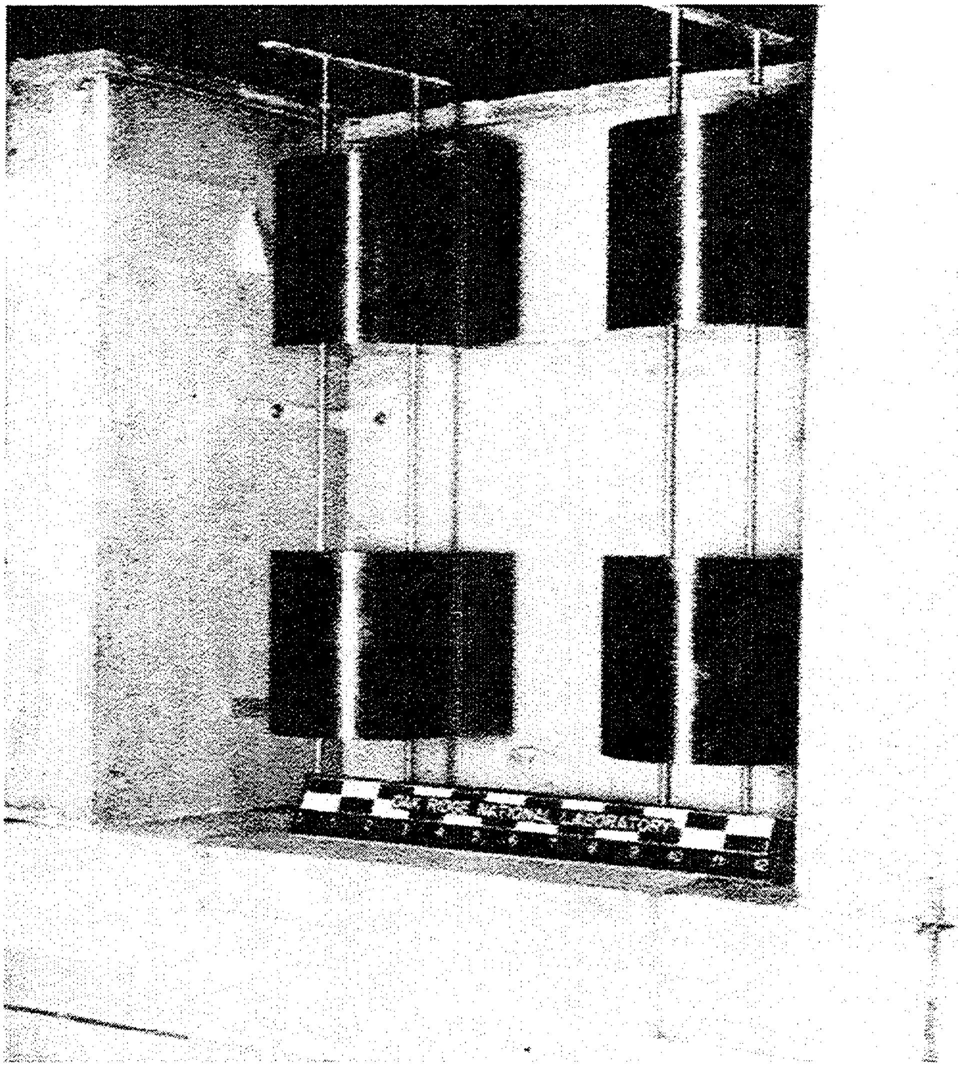


Figure F11.D.2 Half of the paraffin reflected 2C8 assembly before the top reflector was added

INPUT DATA

```
#kenova
sample problem 3  2c8  15.24 cm paraffin refl
read param
  lib=4  flx=yes fdn=yes pwt=yes end param
read array nux=2 nuy=2 nuz=2 end array
read mixt
mix=1 1092234 4.82716e-04 1092235 4.47971e-02 1092236 9.57231e-05
1092238 2.65767e-03  mix=2 10006012 3.97311e-02 10001001 8.26407e-02
sct=2
end mixt
read geom
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 11.74 -11.74 11.74 -11.74 11.375 -11.375
core  0 1 -23.48 -23.48 -22.75
cuboid  2 2 26.48 -26.48 26.48 -26.48 25.75 -25.75
cuboid  2 3 29.48 -29.48 29.48 -29.48 28.75 -28.75
cuboid  2 4 32.48 -32.48 32.48 -32.48 31.75 -31.75
cuboid  2 5 35.48 -35.48 35.48 -35.48 34.75 -34.75
cuboid  2 6 38.72 -38.72 38.72 -38.72 37.99 -37.99
end geom
read bias  id=400 2 6  end bias  end data
end
```

SAMPLE PROBLEM 4 2C8 15.24-CM PARAFFIN REFL AUTOMATIC REFL

This problem is the same as sample problem 3 except it is set up using the automatic reflector option instead of describing the reflector manually.

INPUT DATA

```
#kenova
sample problem 4  2c8 15.24 cm paraffin refl automatic refl
read param
  pwt=yes lib=4  flx=yes fdn=yes end param
read geometry  cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 11.74 -11.74 11.74 -11.74 11.375 -11.375
core  0 1 -23.48 -23.48 -22.75
reflector  2 2 6*3.0 5
reflector  2 7 6*.24 1
end geom
read mixt sct=2
mix=1 1092234 4.82716e-04 1092235 4.47971e-02 1092236 9.57231e-05
1092238 2.65767e-03  mix=2 10006012 3.97311e-02 10001001 8.26407e-02
end mixt
read arra  nux=2 nuy=2 nuz=2 end array
read bias  id=400 2 7  end bias
end data
end
```

SAMPLE PROBLEM 5 2C8 12-INCH PARAFFIN ALBEDO REFLECTOR

This problem is the same as samples problems 3 and 4 except the reflector is represented by a 12-in. paraffin albedo. Note the decrease in execution time when using an albedo reflector instead of doing actual tracking. Note also that k-effective is somewhat higher for this system, probably due to the small edge size of the system.[2]

INPUT DATA

```
#kenova
sample problem 5  2c8 12 inch paraffin albedo reflector
read para
  flx=yes far=yes gas=no fdn=yes lib=4  end para
read array  nux=2 nuy=2 nuz=2 end array
read mixt mix=1 1092234 4.82716e-04 1092235 4.47971e-02
1092236 9.57231e-05 1092238 2.65767e-03  sct=2 end mixt
read bounds  all=paraffin  end bounds
read geom  cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 11.74 -11.74 11.74 -11.74 11.375 -11.375  end geom
end data
end
```

SAMPLE PROBLEM 6 ONE 2C8 UNIT (SINGLE UNIT)

One of the 2C units[1] is described and run as a single-unit problem, and its k-effective is calculated.

INPUT DATA

```
#kenova
sample problem 6  one 2c8 unit (single unit)
read para
  lib=4  flx=yes fdn=yes far=yes gas=no end para
read mixt
sct=2 mix=1 1092234 4.82716e-04 1092235 4.47971e-02 1092236
9.57231e-05 1092238 2.65767e-03  end mixt
read geometry  cylinder  1 1 5.748 5.3825 -5.3825
end geometry  end data
end
```

SAMPLE PROBLEM 7 BARE 2C8 USING SPECULAR REFLECTION

One of the 2C units[1] is described and the 2 × 2 × 2 array is simulated by using specular reflection on the positive x, y, and z faces of the unit. This is a simulation of sample problem 1.

INPUT DATA

```
#kenova
sample problem 7  bare 2c8 using specular reflection
read para
  lib=4  flx=yes fdn=yes far=yes gas=no end parameters
read mixt mix=1 1092234 4.82716e-04 1092235 4.47971e-02
```

```
1092236 9.57231e-05 1092238 2.65767e-03  sct=2 end mixt
read geom  cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
end geom
read bounds +fc=specular end bounds  end data
end
```

SAMPLE PROBLEM 8 INFINITELY LONG CYLINDER FROM 2C8 UNIT

The fuel and cylinder radius from sample problem 1 is used. The length of the cylinder is arbitrarily chosen to be 20 cm, and the unit is specularly reflected on the top and bottom to create an infinitely long cylinder.

INPUT DATA

```
#kenova
sample problem 8  infinitely long cylinder from 2c8 unit
read param
  lib=4  end param
read mixt mix=1 1092234 4.82716e-04 1092235 4.47971e-02
1092236 9.57231e-05 1092238 2.65767e-03 sct=2
end mixtures
read geometry  cylinder  1 1 5.748 10.0 -10.0
cuboid  0 1 6.87 -6.87 6.87 -6.87 10.0 -10.0
end geometry
read bounds zfc=mirror  end bounds
end data
end
```

SAMPLE PROBLEM 9 INFINITE ARRAY OF 2C8 UNITS

The geometry description from sample problem 1 is used, and the cuboid is specularly reflected on all faces to create an infinite array of 2C8 units having an edge-to-edge spacing of 2.244 cm in the x and y directions and 2.245 cm in the z direction.

INPUT DATA

```
#kenova
sample problem 9  infinite array of 2c8 units
read param
  lib=4  gen=103 end param
read mixtures    sct=2
mix=1 1092234 4.82716e-04 1092235 4.47971e-02 1092236 9.57231e-05
1092238 2.65767e-03
end mixt
read boun  all=mir  end boun
read geom
cylinder  1 1 5.748 5.3825 -5.3825  cuboid  0 1 6.87 -6.87 6.87 -6.87
6.505 -6.505  end geom  end data
end
```

SAMPLE PROBLEM 10 2C8 BARE WRITE RESTART

This problem is the same as sample problem 1, a 2 × 2 × 2 array of metal cylinders. Restart information is written on unit 94 after the completion of every fifth generation.

INPUT DATA

```
#kenova
sample problem 10  case 2c8 bare  write restart
read parameters
  flx=yes fdn=yes far=yes gas=no lib=4  res=5 wrs=94
end parameters
read mixt mix=1 1092234 4.82716e-04 1092235 4.47971e-02
1092236 9.57231e-05 1092238 2.65767e-03 sct=2 end mixtures
read geometry
cylinder 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
end geometry
read array  nux=2 nuy=2 nuz=2  end array
end data
end
```

SAMPLE PROBLEM 11 2C8 BARE READ RESTART DATA

This problem is a restart of sample problem 10. The problem is restarted from the tenth set of restart data that was written by sample problem 10 (i.e., it restarts with the fifty-first generation).

INPUT DATA

```
#kenova
sample problem 11  2c8 bare  read restart data
read param    beg=51  rst=94 res=0 end param
end data
end
```

SAMPLE PROBLEM 12 4 AQUEOUS 4 METAL

This problem is a critical experiment consisting of a composite array[1,3] of four highly enriched uranium metal cylinders and four cylindrical plexiglas containers filled with uranyl nitrate solution. The metal units in this experiment are designated in Table II of Ref. 1 as cylinder index 11 and reflector index 1. A photograph of the experiment is given in Fig. F11.D.3.

INPUT DATA

```
#kenova
sample problem 12 4 aqueous 4 metal mixed units
read param
  lib=4 fdn=yes nub=yes smu=yes mkp=yes
mku=yes fmp=yes fmu=yes end param
```

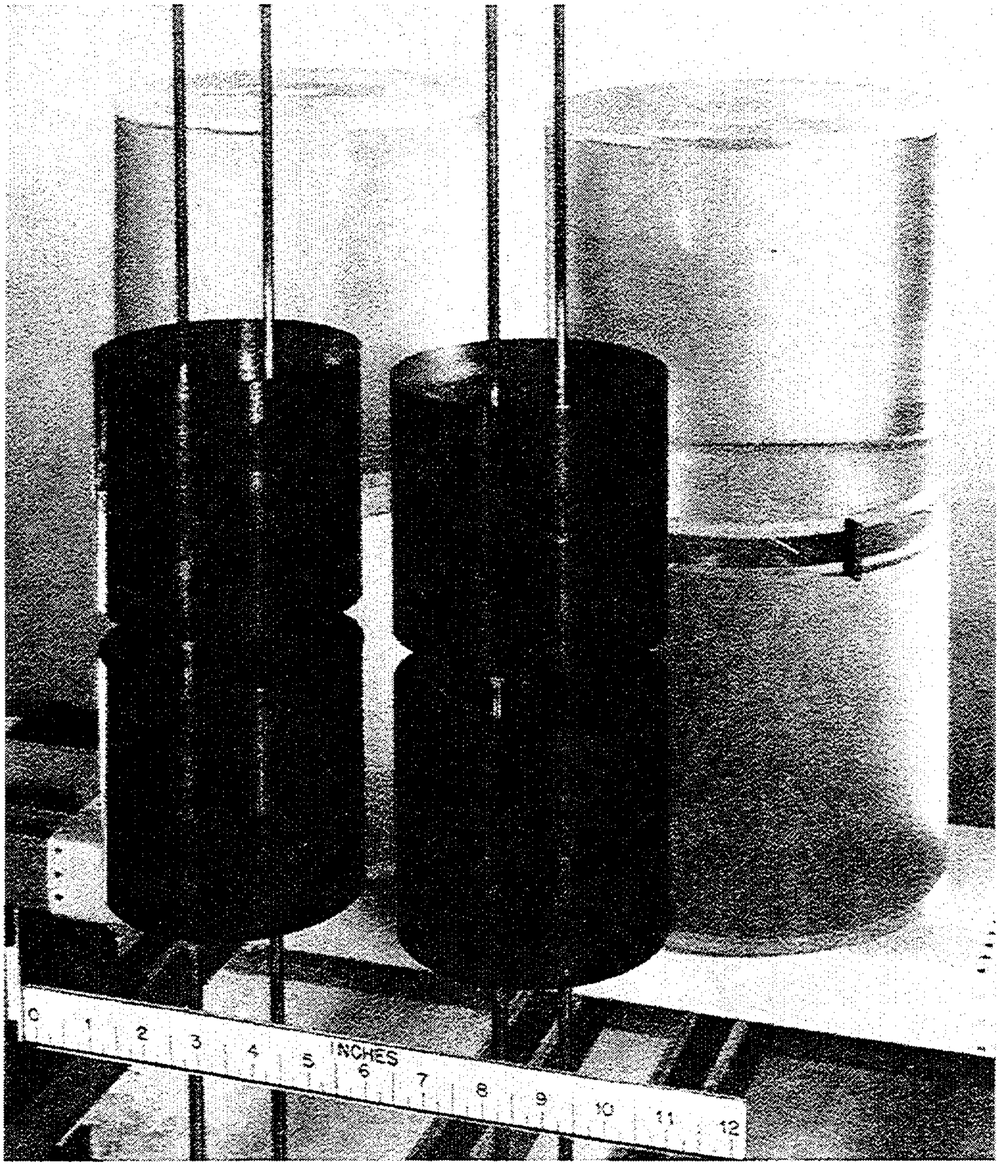
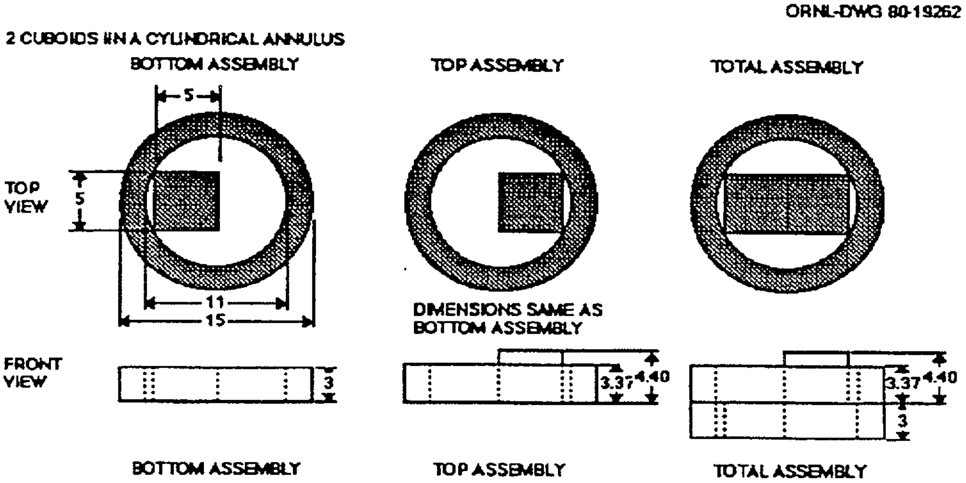


Figure F11.D.3. Critical assembly of 4 solution units and 4 metal units

```
read mixt  sct=2
mix=1 1092234 4.82716e-04 1092235 4.47971e-02 1092236 9.57231e-05
      1092238 2.65767e-03
mix=2 2001001 5.77964e-02  2007014 2.13092e-03  2008016 3.74130e-02
      2092234 1.06784e-05  2092235 9.84599e-04  2092236 5.29385e-06
      2092238 6.19413e-05
mix=3 11001001 5.68187e-02  11006012 3.55117e-02  11008016 1.42047e-02
end mixt
read geom
box type 1
cylinder  2 1 9.525 8.89 -8.89
cylinder  3 1 10.16 9.525 -9.525
cuboid  0 1 10.875 -10.875 10.875 -10.875 10.24 -10.24
box type  2
box type  2
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.59 -15.16 6.59 -15.16 6.225 -14.255
box type  3
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.59 -15.16 15.16 -6.59 6.225 -14.255
box type  4
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.59 -15.16 6.59 -15.16 14.255 -6.225
box type  5
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.59 -15.16 15.16 -6.59 14.255 -6.225
end geom  read array  nux=2 nuy=2 nuz=2  loop
1 3r2 1 2 1 1 2 1  2 9r1  3 3r1 2 2 1 3r1  4 6r1 2 2 1 5 3r1 2 2 1 2 2 1
end array
end data
end
```

SAMPLE PROBLEM 13 TWO CUBOIDS IN A CYLINDRICAL ANNULUS

This critical experiment[4] consists of two assemblies of 93.2% ^{235}U-enriched uranium metal (ρ = 18.69 g/cc) stacked vertically. The bottom assembly contains a uranium metal cuboid offset to the left within a uranium metal cylindrical annulus. The top assembly contains a uranium metal cuboid offset to the right within a uranium metal cylindrical annulus. The cuboid extends above the annulus. A drawing of the two sections and the total assembly is given in Fig. F11.D.4.



ALL DIMENSIONS ARE IN INCHES

Figure F11.D.4 Drawing of two cuboids in an annulus critical assembly

INPUT DATA

```
#kenova
sample problem 13  two cuboids in a cylindrical annulus
read param
```

```
   lib=4  end param
read geom
unit 1
cuboid  1 1 6.35 -6.35 6.35 -6.35 7.62 0.0
cylinder  0 1 13.97 7.62 0.0 orig -6.0934 0.0
cylinder  1 1 19.05 7.62 0.0 orig -6.0934 0.0
cuboid  0 1 12.9566 -25.1434 19.05 -19.05 7.62 0.0
unit 2
cuboid  1 1 6.35 -6.35 6.35 -6.35 8.56 0.0
cylinder  0 1 13.97 8.56 0.0 origin 6.0934 0.0
cylinder  1 1 19.05 8.56 0.0 origin 6.0934 0.0
cuboid  0 1 25.1434 -12.9566 19.05 -19.05 8.56 0.0
unit 3
cuboid  1 1 6.35 -6.35 6.35 -6.35 2.616 0.0
cuboid  0 1 25.1434 -12.9566 19.05 -19.05 2.616 0.0
end geom
read mixt sct=2
mix=1  3092234 4.80915e-04  3092235 4.46299e-02  3092236 9.53659e-05
3092238 2.64775e-03
end mixt
read array nux=1 nuy=1 nuz=3  fill 1 2 3 t end array end data
end
```

SAMPLE PROBLEM 14 U METAL CYLINDER IN AN ANNULUS

This critical experiment[4] consists of a 93.2 ^{235}U-enriched uranium metal cylinder within a cylindrical annulus of the same material as shown in Fig. F11.D.5. The uranium metal specification is identical to that used in sample problem 13.

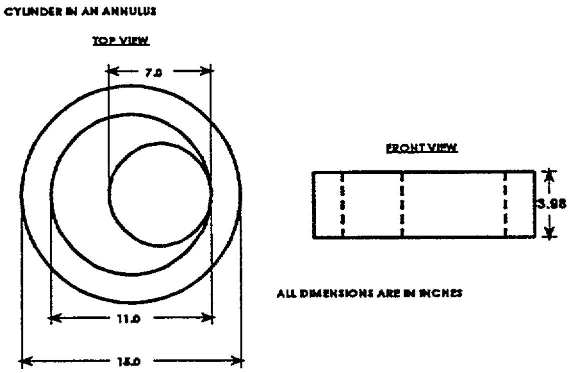


Figure F17.D.5 Drawing of the cylinder in an annulus critical assembly

INPUT DATA

```
#kenova
sample problem 14  u metal cylinder in an annulus
read param
```

```
  lib=4  end param
read mixt sct=2
mix=1  3092234 4.80915e-04  3092235 4.46299e-02  3092236 9.53659e-05
3092238 2.64775e-03
end mixt
read geom
cylinder  1 1 8.89 10.109 0.0 orig 5.0799 0.0
cylinder  0 1 13.97 10.109 0.0
cylinder  1 1 19.05 10.109 0.0
end geom
end data
end
```

SAMPLE PROBLEM 15 SMALL WATER REFLECTED SPHERE ON PLEXIGLAS COLLAR

This critical experiment[5] is a small highly enriched uranium sphere supported by a plexiglas doughnut in a tank of water. The sphere extends down through the hole of the doughnut. However, the KENO V geometry package cannot rigorously describe a doughnut. Therefore, the KENO V mockup of this problem describes the doughnut as an annular cylindrical plate and the sphere is supported by it. Both are contained in a cylindrical tank of water. A drawing of the experiment is given in Fig. F11.D.6. This drawing shows the sphere above the cylindrical collar for the sake of clarity. The sphere is actually supported by the collar and extends into the opening in its center. The actual experiment utilized a torus or doughnut instead of a cylindrical collar.

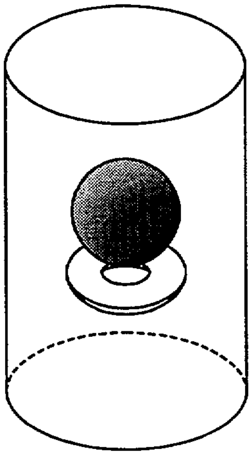


Figure F11.D.6 Drawing of a critical assembly consisting of a uranium sphere on a plexiglas collar with a cylindrical water reflector

INPUT DATA

```
#kenova
sample problem 15  small water reflected sphere on plexiglas collar
read param
  lib=4  flx=yes fdn=yes end param
read mixt  sct=2
mix=1  4092234 5.65801e-04  4092235 4.70211e-02  4092236 9.58966e-05
       4092238 4.65935e-04
mix=2 11001001 5.68187e-02  11006012 3.55117e-02  11008016 1.42047e-02
mix=3 12001001 6.67514e-02
mix=3 12008016 3.33757e-02
end mixt
read geom
unit 1
hemisphe-z 1 1 6.5537 chord -5.09066
cylinder  3 1 4.1275 -5.09066 -7.63065
cylinder  2 1 12.7   -5.09066 -7.63065
cuboid  3 1 4p12.7 -5.09066 -7.63065
unit 2
hemisphe+z 1 1 6.5537 chord 5.09066
cuboid  3 1 4p12.7 6.5537 -5.09066
core  0 1 -12.7 -12.7 -7.092175
cylinder  3 1 17.97 2p7.0922
replicate 3 2 3*3.0 5
end geom
read bias  id=500 2 6 end bias
read array nux=1 nuy=1 nuz=2  fill 1 2 end array
read plot   scr=yes lpi=10
 ttl='x-z slice through the center of the sphere'
xul=-20.0 zul=10.0 yul=0.0  xlr=20.0 ylr=0.0 zlr=-10.0
uax=1.0 wdn=-1.0 nax=400
end plot
end data
end
```

SAMPLE PROBLEM 16 UO2F2 INFINITE SLAB K-INFINITY

This problem solves for the k-infinity of an infinite number of slabs of uranyl fluoride solution contained in pyrex glass and separated by borated uranyl fluoride solution. The uranyl fluoride slab is 4.958 cm thick, 93.2% enriched, and has a density of 578.7 g U/l. The pyrex glass is 1.27 cm thick and is present on both faces of the uranyl fluoride solution. A total of 27.46 cm of borated solution separates the pyrex glass of adjacent slabs of solution. 1.482×10^{-27} atoms of boron per milliliter are present in the borated solution.

INPUT DATA

```
#kenova
sample problem 16 uo2f2 infinite slab k-infinity
read parameters
  lib=4  amx=yes xap=no        end parameters
read mixt  sct=2
mix=1  5009019 2.96286e-03  5001001 6.04824e-02  5008016 3.32041e-02
       5092235 1.38188e-03  5092238 9.95503e-05
mix=2 13011023 2.39502e-03 13013027 4.97719e-04 13014000 1.80267e-02
      13005010 9.08241e-04 13005011 3.68719e-03 13008016 4.49173e-02
mix=3  6009019 2.96286e-03  6001001 6.04824e-02  6008016 3.32041e-02
       6092235 1.38188e-03  6092238 9.95503e-05
       6005010 2.92803e-04  6005011 1.18870e-03
end mixt
read geometry
cuboid 1 1 2.479 -2.479 100 -100 100 -100
cuboid 2 1 3.749 -3.749 100 -100 100 -100
cuboid 3 1 17.479 -17.479 100 -100 100 -100
end geom
read bounds  all=mirror  end bounds    read array  end array
end data
end
```

SAMPLE PROBLEM 17 93% UO2F2 SOLUTION SPHERE ADJOINT CALCULATION

A single 93% enriched uranyl fluoride sphere is run as an adjoint calculation. The result for the forward and adjoint k-effectives should be the same within statistical error when the problem is run both ways.

INPUT DATA

```
#kenova
sample problem 17 93% uo2f2 solution sphere  adjoint calculation
read parameters
  lib=4  npg=10000 nbk=10500 adj=yes amx=yes xap=no end parameters
read mixt  sct=2
mix=1  7001001 6.54785e-02  7008016 3.34202e-02  7009019 6.80923e-04
       7092235 3.16909e-04  7092238 2.35521e-05
end mixt
read geometry
sphere 1 1 16.0
end geom
end data
end
```

SAMPLE PROBLEM 18 1F27 DEMONSTRATION OF OPTIONS

A reflected cubic array of 27 cylinders of aqueous uranyl nitrate in plexiglas bottles.[6] The walls of the bottles were 0.64-cm thick, and each bottle was filled with 5 liters of 92.6% enriched solution at a concentration of 415 g/L, a specific gravity of 1.555 and 0.39 mg excess nitrate/g soln.* The 3 × 3 × 3 array was surrounded by a 6-in. paraffin reflector. Most of the print options available in KENO V.a are exercised in this problem. A perspective of this critical experiment is shown in Fig. F11.D.7. A photograph of one of the experiments utilizing 27 of the plexiglas bottles is shown in Fig. F11.D.8. Sample problem 18 has 15.24 cm of paraffin on all six faces rather than the 2.54-cm plexiglas shown on five faces in the photograph.

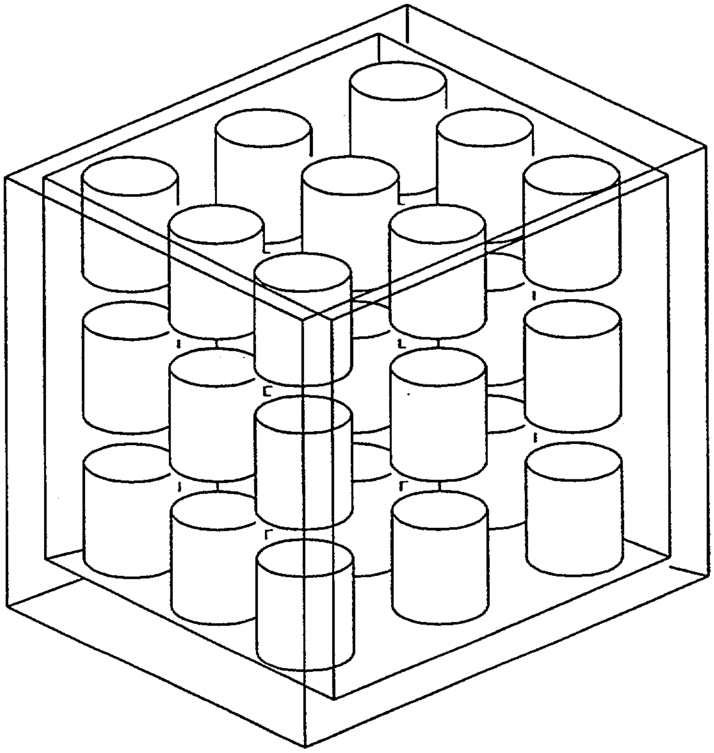


Figure F11.D.7. Perspective of critical 1F27 experiment

*From experimental facility documents. Not reported in ORNL/TM-719.

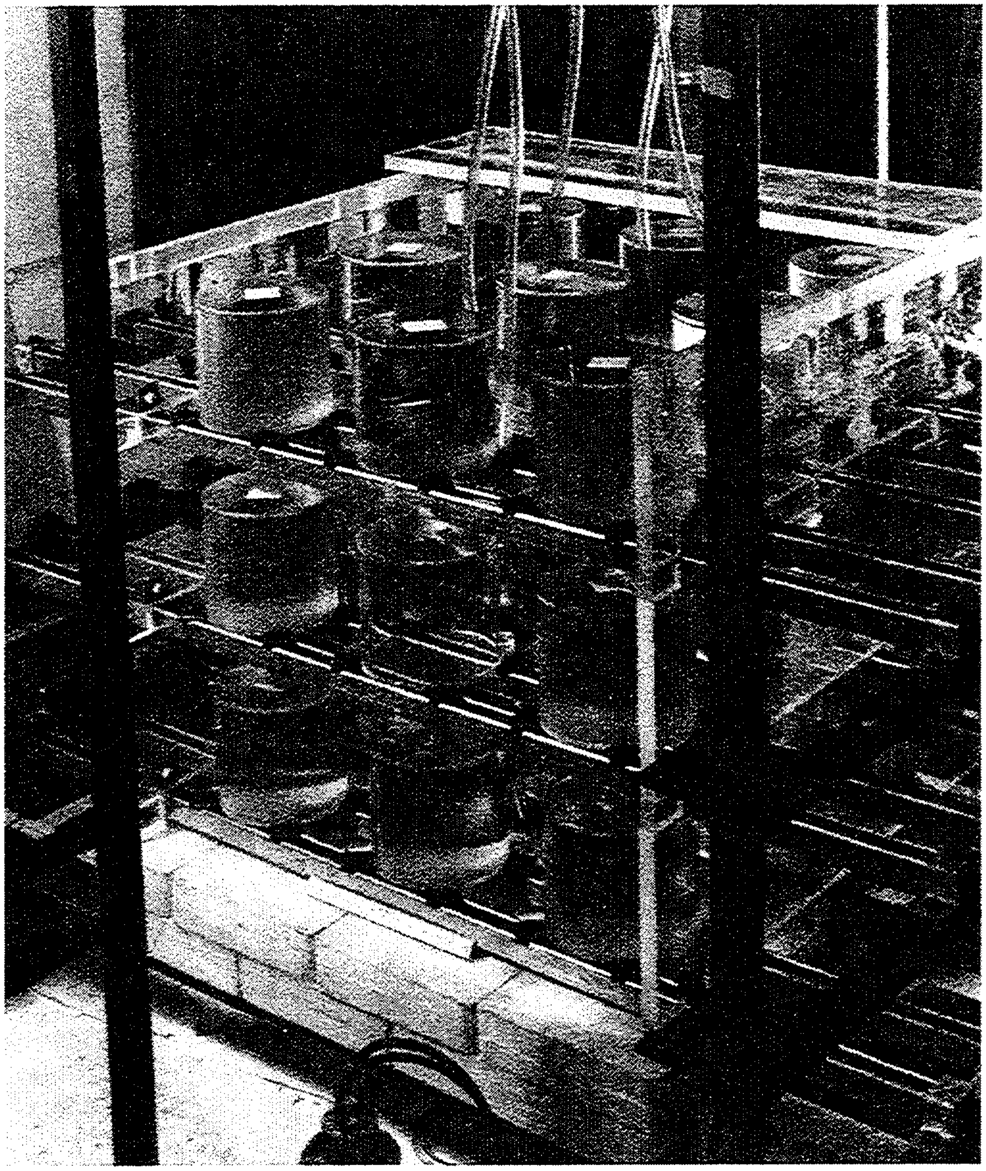


Figure F11.D.8. View of a 27-unit cubic array with 2.54-cm. thick plexiglas reflector on five sides and a 15.24-cm. thick paraffin base

INPUT DATA

```
#kenova
sample problem 18   1f27 demonstration of options problem
read para   lng=100000 gen=103 npg=500 fdn=yes nub=yes lib=4
  mku=yes fmu=yes mkh=yes fmh=yes mka=yes fma=yes rnd=f12c09ed2195
  pwt=yes far=yes flx=yes amx=yes pax=yes pgm=yes
 end para
read mixt
  sct=2
  mix=1 2001001 5.77964e-02  2007014 2.13092e-03  2008016 3.74130e-02
        2092234 1.06784e-05  2092235 9.84599e-04  2092236 5.29385e-06
        2092238 6.19413e-05
  mix=2 11001001 5.68187e-02 11006012 3.55117e-02  11008016 1.42047e-02
  mix=3 10006012 3.97311e-02 10001001 8.26407e-02
  mix=4 15008016 3.33757e-11 15001001 6.67514e-11
 end mixt
read bounds   -zb= h2o   end bounds
read geom unit 1 cylinder 1 1 9.52 8.7804 -8.7804
 cylinder 0 1 9.52 8.9896 -8.7804
 cylinder 2 1 10.16 9.6296 -9.4204
 cuboid 4 1 18.45 -18.45 18.45 -18.45 17.8946 -17.6854
 unit 2 array 1 3*0.0
 unit 3 array 2 3*0.0
 unit 4 array 3 3*0.0
 unit 5 array 4 3*0.0
 global
 unit 6 cuboid 4 1 55.3501 -55.3501 55.3501 -55.3501 53.3701 -53.3701
  hole 2 -55.35   -18.45   -17.79
  hole 3 -55.35   -18.45   -53.3701
  hole 4  18.4501 -18.45   -53.3701
  hole 5 -55.3501 -55.3501 -53.3701
 replicate 3 2 6*3 5  replicate 3 7 6*0.24 1  end geom
 read bias  id=400 2 7  end bias
read array
 ara=1 nux=2 nuy=2 nuz=2 fill f1 end fill
 ara=2 nux=2 nuy=2 nuz=1 fill f1 end fill
 ara=3 nux=1 nuy=2 nuz=3 fill f1 end fill
 ara=4 nux=3 nuy=1 nuz=3 fill f1 end fill
 end array
 read start nst=6 tfx=0.0 tfy=0.0 tfz=0.0
 lnu=500 ps6=yes
 end start
read plot          scr=yes  plt=yes lpi=10
ttl=?  1f27 xy plot at z=0.0 ?  xul=-71.0 yul=71.0 zul=0.0
 xlr=71.0 ylr=-71.0 zlr=0.0 uax=1 vdn=-1 nax=400
 run=yes end      ttl=?unit map 1f27 xy plot at z=0.0?
 pic=unit end plot    end data
end
```

SAMPLE PROBLEM 19 4 AQUEOUS 4 METAL ARRAY OF ARRAYS (SAMP PROB 12)

This critical experiment was described previously as SAMPLE PROBLEM 12. The input data given below utilize the array of arrays option. See Fig. F11.D.3.

INPUT DATA

```
#kenova
sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)
read param
  lib=4  flx=yes fdn=yes nub=yes smu=yes mkp=yes
mku=yes fmp=yes fmu=yes end param
read mixt
mix=1 1092234 4.82716e-04 1092235 4.47971e-02 1092236 9.57231e-05
      1092238 2.65767e-03
mix=2 2001001 5.77964e-02 2007014 2.13092e-03 2008016 3.74130e-02
      2092234 1.06784e-05 2092235 9.84599e-04 2092236 5.29385e-06
      2092238 6.19413e-05
mix=3 11001001 5.68187e-02  11006012 3.55117e-02  11008016 1.42047e-02
sct=2
end mixt
read geom
unit 1
com='uranyl nitrate solution in a plexiglas container'
cylinder  2 1 9.525 2p8.89
cylinder  3 1 10.16 2p9.525
cuboid  0 1 4p10.875 2p10.24
unit 2
com='uranium metal cylinder'
cylinder  1 1 5.748 2p5.3825
cuboid  0 1 4p6.59 2p6.225
unit 3
com='1x2x2 array of solution units'
array 1 3*0.0
unit 4
com='1x2x2 array of metal units padded to match solution array'
array 2 3*0.0
replicate 0 1 2*0.0 2*8.57 2*8.03 1
end geom
read array ara=1 nux=1 nuy=2 nuz=2 fill f1 end fill
ara=2 nux=1 nuy=2 nuz=2 fill f2 end fill gbl=3 ara=3 nux=2 nuy=1 nuz=1
com='composite array of solution and metal units'
fill 4 3 end fill
end array
end data
end
```

SAMPLE PROBLEM 20 TRIANGULAR PITCHED ARRAY

This problem is a critical experiment[7] consisting of seven cylinders in a triangular-pitched unreflected array. The central cylinder has six cylinders arranged around it. The surface-to-surface separation between the units is 0.15 in. Each unit consists of a 60-mil-thick aluminum can with an 8-in. inside diameter, filled with a solution of 93.2% enriched uranyl fluoride with a H/^{235}U atomic ratio of 44.3 and a density of 576.87 g U/L. The apparatus for conducting this experiment is shown in Fig. F11.D.9.

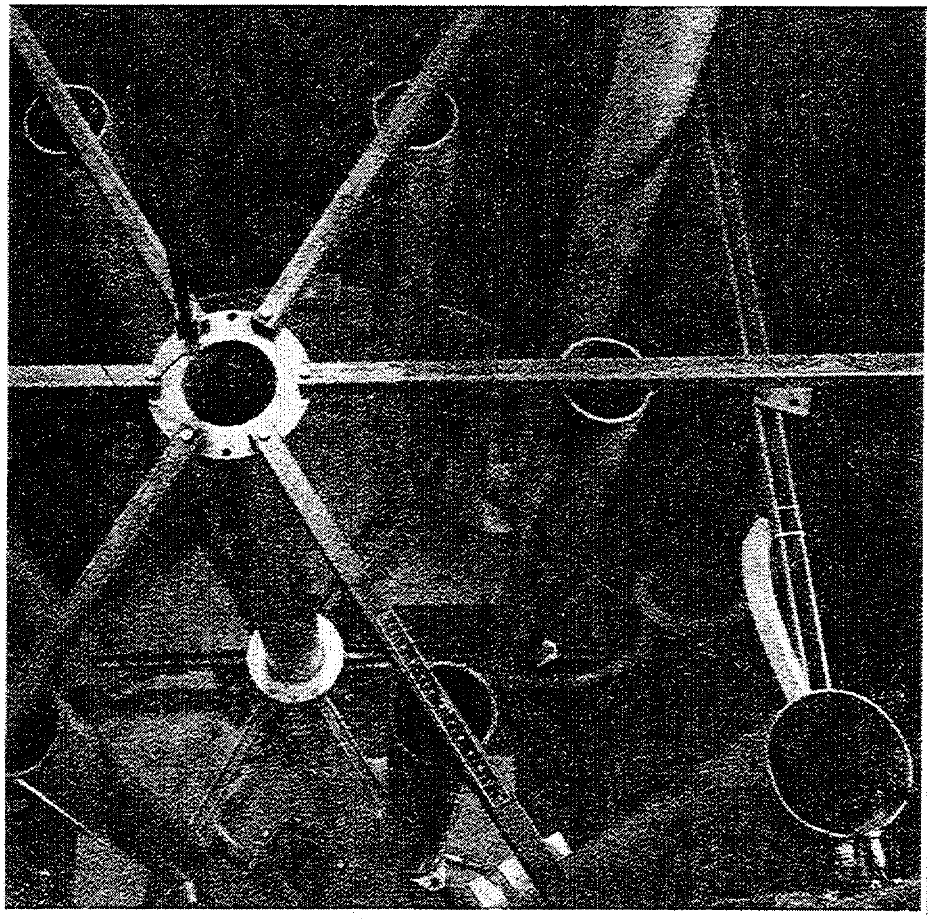


Figure F11.D.9 Typical arrangement for critical experiments with interacting arrays of aluminum cylinders containing enriched ^{235}U solutions

INPUT DATA

```
#kenova
sample problem 20 triangular pitched array
read param
  lib=4  end param
read mixt sct=2
mix=1  8092235 1.37751e-03  8092238 9.92354e-05  8008016 3.32049e-02
       8009019 2.95349e-03  8001001 6.05028e-02
mix=2 14013027 6.02374e-02
end mixt
read geom
unit 1
cylinder   1 1 10.16 18.288 0
cylinder   2 1 10.312 18.288 -.152
unit 2
cuboid     0 1 4p50 50 -.152
hole       1 3r0
hole       1 21.006 2r0
hole       1 -21.006 2r0
hole       1 10.503 18.192 0
hole       1 -10.503 18.192 0
hole       1 10.503 -18.192 0
hole       1 -10.503 -18.192 0
end geom
read array nux=1 nuy=1 nuz=1 fill 2 end fill end array
read plot ttl='hex array' pic=mix lpi=10 scr=yes
xul=0 yul=100 zul=10
xlr=100 ylr=0 zlr=10 uax=1 vdn=-1 nax=400
 end plot
end data
end
```

SAMPLE PROBLEM 21 PARTIALLY FILLED SPHERE

This critical experiment[8] consisted of a partially filled, unreflected spherical container. This aluminum container had an inside diameter of 27.244 in. and a wall thickness of 1/16 in. It is referred to in the report as the 27.3-in.-diameter vessel. The sphere was 98% filled with uranyl fluoride at an enrichment of 4.89% with an $H/^{235}U$ atomic ratio of 1099. The height of the solution in the sphere was 64.6 cm above the bottom of the sphere. A schematic diagram of the apparatus used in the experiment is given in Fig. F11.D.10. The steel tank was ignored.

INPUT DATA

```
#kenova
sample problem 21  partially filled sphere
read param
  lib=4  end param
read geom
hemisphe-z  1 1 34.6   chord 30.
sphere      0 1 34.6
sphere      2 1 34.759
end geom
read mixt sct=2
mix=1  9001001 6.15670e-02  9008016 3.32845e-02  9009019 2.50098e-03
       9092234 2.54223e-07  9092235 6.18922e-05  9092238 1.18834e-03
```

```
mix=2 14013027 6.02374e-02
end mixt
end data
end
```

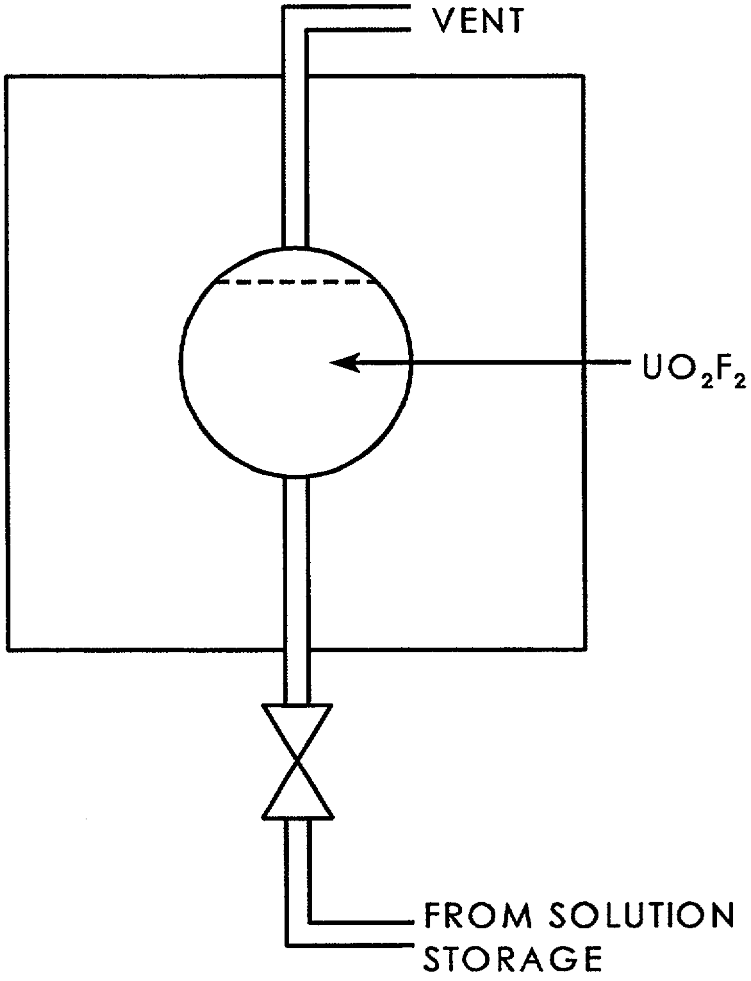


Figure F11.D.10 Schematic of bare partially filled sphere experiment inside a 9.5-ft-diameter, 9-ft-high steel tank

SAMPLE PROBLEM 22 CASE 2C8 BARE WITH 3 NESTED HOLES, EACH IS EQUAL VOLUME

The physical representation of this sample problem is the critical experiment described in sample problem 1. It is a simple 2 × 2 × 2 array of 93.2%$_{wt}$ enriched uranium metal cylinders. This sample problem defines a uranium cylinder in a void spacing cuboid using nested holes. Eight of these units are stacked together in a 2 × 2 × 2 array.

INPUT DATA

```
#kenova
sample problem 22   case 2c8 bare with 3 nested, equal volume holes
read parameters
  flx=yes fdn=yes far=yes gas=no lib=4  end parameters
read mixt
sct=2 mix=1 1092234 4.82716e-04 1092235 4.47971e-02 1092236 9.57231e-05
1092238 2.65767e-03  end mixt
read geometry

unit 1
cylinder 1 1 3.621 2p3.3907

unit 2
cylinder 1 1 4.5622 2p4.2721
hole 1 3*0.0

unit 3
cylinder 1 1 5.2224 2p4.8903
hole 2 3*0.0

unit 4
cylinder 1 1 5.748 5.3825 -5.3825
hole 3 3*0.0
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505

end geometry
read array  nux=2 nuy=2 nuz=2 fill f4 end fill end array
end data
end
```

SAMPLE PROBLEM 23 CASE 2C8 BARE AS MIXED ZHEMICYLINDERS

The physical representation of this sample problem is the critical experiment described in sample problem 1. This sample problem describes each of the eight units in the critical 2 × 2 × 2 array using Z hemicylinders.

INPUT DATA

```
#kenova
sample problem 23  case 2c8 bare as mixed zhemicylinders
read parameters
  fdn=yes lib=4  end parameters
read mixt sct=2  mix=1 1092234 4.82716e-04 1092235 4.47971e-02
1092236 9.57231e-05  1092238 2.65767e-03  end mixt
read geometry
unit 1
com='-x half of unit 3'
zhemicyl-x 1 1 5.748 5.3825 -5.3825
cuboid  0 1 0.0  -6.87 6.87 -6.87 6.505 -6.505
```

```
unit 2
com='+x half of unit 3'
zhemicyl+x 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87  0.0  6.87 -6.87 6.505 -6.505
unit 3
com='cylinder composed of equal halves (zhemicylinders with x radii)'
array 1 3*0.0
unit 4
com='-x portion (more than half) of unit 6'
zhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 3.0  -6.87 6.87 -6.87 6.505 -6.505
unit 5
com='+x portion (less than half) of unit 6'
zhemicyl+x 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.87  3.0  6.87 -6.87 6.505 -6.505
unit 6
com='cylinder composed of unequal halves (zhemicylinders with x radii)'
array 2 3*0.0
unit 7
com='cylinder of a single zhemicylinder in the -x direction'
zhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 8
com='cylinder of a single zhemicylinder in the +x direction'
zhemicyl+x 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 9
com='-y half of unit 11'
zhemicyl-y 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 0.0  -6.87 6.505 -6.505
unit 10
com='+y half of unit 11'
zhemicyl+y 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87  0.0  6.505 -6.505
unit 11
com='cylinder composed of equal halves (zhemicylinders with z radii)'
array 3 3*0.0
unit 12
com='-y portion (more than half) of unit 14'
zhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 6.87 -6.87 3.0  -6.87 6.505 -6.505
unit 13
com='+y portion (less than half) of unit 14'
zhemicyl+y 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.87 -6.87 6.87  3.0  6.505 -6.505
unit 14
com='cylinder composed of unequal halves (zhemicylinders with z radii)'
array 4 3*0.0
unit 15
com='cylinder of a single zhemicylinder in the -y direction'
zhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 16
com='cylinder of a single zhemicylinder in the +y'
zhemicyl+y 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
end geometry
read array
com='array 1 defines unit 3 (zhemicylinders with x radii)'
ara=1 nux=2 nuy=1 nuz=1 fill 1 2 end fill
com='array 2 defines unit 6 (zhemicylinders with x radii)'
ara=2 nux=2 nuy=1 nuz=1 fill 4 5 end fill
com='array 3 defines unit 11 (zhemicylinders with y radii)'
ara=3 nux=1 nuy=2 nuz=1 fill 9 10 end fill
com='array 4 defines unit 14 (zhemicylinders with y radii)'
ara=4 nux=1 nuy=2 nuz=1 fill 12 13 end fill
```

```
com='array 5 defines the total 2c8 problem'
gbl=5 ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

SAMPLE PROBLEM 24 CASE 2C8 BARE AS MIXED XHEMICYLINDERS

The physical representation of this sample problem is the critical experiment described in sample problem 1. This sample problem describes each of the eight units in the critical 2 × 2 × 2 array using hemicylinders whose axes are in the x direction.

INPUT DATA

```
#kenova
sample problem 24  case 2c8 bare as mixed xhemicylinders
read parameters
  fdn=yes lib=4  end parameters
read mixt mix=1 1092234 4.82716e-04 1092235 4.47971e-02
1092236 9.57231e-05 1092238 2.65767e-03  sct=2 end mixt
read geometry
unit 1
com='-y half of unit 3'
xhemicyl-y 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.505 -6.505 0.0 -6.87 6.87 -6.87
unit 2
com='+y half of unit 3'
xhemicyl+y 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.505 -6.505 6.87 0.0  6.87 -6.87
unit 3
com='cylinder composed of equal halves (xhemicylinders with y radii)'
array 1 3*0.0
unit 4
com='-y portion (more than half) of unit 6'
xhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 6.505 -6.505 3.0 -6.87 6.87 -6.87
unit 5
com='+y portion (less than half) of unit 6'
xhemicyl+y 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.505 -6.505 6.87 3.0  6.87 -6.87
unit 6
com='cylinder composed of unequal halves (xhemicylinders with y radii)'
array 2 3*0.0
unit 7
com='cylinder of a single xhemicylinder in the -y direction'
xhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 -6.87
unit 8
com='cylinder of a single xhemicylinder in the +y direction'
xhemicyl+y 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 -6.87
unit 9
com='-z half of unit 11'
xhemicyl-z 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.505 -6.505 6.87 -6.87 0.0 -6.87
unit 10
com='+z half of unit 11'
xhemicyl+z 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 0.0
```

```
unit 11
com='cylinder composed of equal halves (xhemicylinders with z radii)'
array 3 3*0.0
unit 12
com='-z portion (more than half) of unit 14'
xhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 6.505 -6.505 6.87 -6.87 3.0 -6.87
unit 13
com='+z portion (less than half) of unit 14'
xhemicyl+z 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 3.0
unit 14
com='cylinder composed of unequal halves (xhemicylinders with z radii)'
array 4 3*0.0
unit 15
com='cylinder of a single xhemicylinder in the -z direction'
xhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 -6.87
unit 16
com='cylinder of a single xhemicylinder in the +z direction'
xhemicyl+z 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 -6.87
end geometry
read array
com='array 1 defines unit 3 (xhemicylinders with y radii)'
ara=1 nux=1 nuy=2 nuz=1 fill 1 2 end fill
com='array 2 defines unit 6 (xhemicylinders with y radii)'
ara=2 nux=1 nuy=2 nuz=1 fill 4 5 end fill
com='array 3 defines unit 11 (xhemicylinders with z radii)'
ara=3 nux=1 nuy=1 nuz=2 fill 9 10 end fill
com='array 4 defines unit 14 (xhemicylinders with z radii)'
ara=4 nux=1 nuy=1 nuz=2 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
gbl=5 ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

SAMPLE PROBLEM 25 CASE 2C8 BARE AS MIXED YHEMICYLINDERS

The physical representation of this sample problem is the critical experiment described in sample problem 1. This sample problem describes each of the eight units in the critical 2 × 2 × 2 array using hemicylinders whose axes are in the y direction.

INPUT DATA

```
#kenova
sample problem 25  case 2c8 bare as mixed yhemicylinders
read parameters
   fdn=yes lib=4  end parameters
read mixt mix=1 1092234 4.82716e-04 1092235 4.47971e-02
1092236 9.57231e-05 1092238 2.65767e-03  sct=2 end mixt
read geometry
unit 1
com='-x half of unit 3'
yhemicyl-x 1 1 5.748 5.3825 -5.3825
cuboid  0 1 0.0  -6.87 6.505 -6.505 6.87 -6.87
unit 2
com='+x half of unit 3'
```

```
yhemicyl+x 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87  0.0  6.505 -6.505 6.87 -6.87
unit 3
com='cylinder composed of equal halves (yhemicylinders with x radii)'
array 1 3*0.0
unit 4
com='-x portion (more than half) of unit 6'
yhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 3.0  -6.87 6.505 -6.505 6.87 -6.87
unit 5
com='+x portion (less than half) of unit 6'
yhemicyl+x 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.87  3.0  6.505 -6.505 6.87 -6.87
unit 6
com='cylinder composed of unequal halves (yhemicylinders with x radii)'
array 2 3*0.0
unit 7
com='cylinder of a single yhemicylinder in the -x direction'
yhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 -6.87
unit 8
com='cylinder of a single yhemicylinder in the +x direction'
yhemicyl+x 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 -6.87
unit 9
com='-z half of unit 11'
yhemicyl-z 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.505 -6.505 0.0 -6.87
unit 10
com='+z half of unit 11'
yhemicyl+z 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 0.0
unit 11
com='cylinder composed of equal halves (yhemicylinders with z radii)'
array 3 3*0.0
unit 12
com='-z portion (more than half) of unit 14'
yhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 6.87 -6.87 6.505 -6.505 3.0 -6.87
unit 13
com='+z portion (less than half) of unit 14'
yhemicyl+z 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 3.0
unit 14
com='cylinder composed of unequal halves (yhemicylinders with z radii)'
array 4 3*0.0
unit 15
com='cylinder of a single  yhemicylinder in the -z direction'
yhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 -6.87
unit 16
com='cylinder of a single yhemicylinder in the +z direction'
yhemicyl+z 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 -6.87
end geometry
read array
com='array 1 defines unit 3 (yhemicylinders with x radii)'
ara=1 nux=2 nuy=1 nuz=1 fill 1 2 end fill
com='array 2 defines unit 6 (yhemicylinders with x radii)'
ara=2 nux=2 nuy=1 nuz=1 fill 4 5 end fill
com='array 3 defines unit 11 (yhemicylinders with z radii)'
ara=3 nux=1 nuy=1 nuz=2 fill 9 10 end fill
com='array 4 defines unit 14 (zhemicylinders with z radii)'
```

```
ara=4 nux=1 nuy=1 nuz=2 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
gbl=5 ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

SAMPLE PROBLEM 26 CASE 2C8 BARE AS MIXED ZHEMICYLINDERS WITH ORIGINS

The physical representation of this sample problem is the critical experiment described in sample problem 1. This sample problem describes each of the eight units in the critical 2 × 2 × 2 array using zhemicylinders with origins.

INPUT DATA

```
#kenova
sample problem 26  case 2c8 bare as mixed zhemicylinders with origins
read parameters
  fdn=yes lib=4  run=yes end parameters
read mixt sct=2  mix=1 1092234 4.82716e-04 1092235 4.47971e-02
1092236 9.57231e-05  1092238 2.65767e-03  end mixt
read geometry
unit 1
com='-x half of first cylinder'
zhemicyl-x 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
cuboid  0 1 6.87 0.0 6.87 -6.87 6.505 -6.505
unit 2
com='+x half of first cylinder'
zhemicyl+x 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
cuboid  0 1 13.74 6.87 6.87 -6.87 6.505 -6.505
unit 3
com='1st cylinder composed of equal portions (z hemicylinders with x radii)'
array 1 3*0.0
unit 4
com='-x portion (more than half) of second cylinder'
zhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 6.87 0.0
cuboid  0 1 9.87 0.0 6.87 -6.87 6.505 -6.505
unit 5
com='+x portion (less than half) of second cylinder'
zhemicyl+x 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 6.87 0.0
cuboid  0 1 13.74 9.87 6.87 -6.87 6.505 -6.505
unit 6
com='2nd cylinder composed of unequal portions (z hemicylinders with x radii)'
array 2 3*0.0
unit 7
com='3rd cylinder: described as a zhemicylinder in the -x direction'
zhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 13.74 0.0 6.87 -6.87 6.505 -6.505
unit 8
com='4th cylinder: described as a zhemicylinder in the +x direction'
zhemicyl+x 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 13.74 0.0 6.87 -6.87 6.505 -6.505
unit 9
com='-y half of fifth cylinder'
zhemicyl-y 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.87 0.0 6.505 -6.505
unit 10
com='+y half of fifth cylinder'
zhemicyl+y 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
```

```
cuboid  0 1 6.87 -6.87 13.74 6.87  6.505 -6.505
unit 11
com='5th cylinder composed of equal portions (zhemicylinders with y radii)'
array 3 3*0.0
unit 12
com='-y portion (more than half) of sixth cylinder'
zhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 9.87  0.0 6.505 -6.505
unit 13
com='+y portion (less than half) of sixth cylinder'
zhemicyl+y 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 13.74  9.87  6.505 -6.505
unit 14
com='6th cylinder composed of unequal portions (zhemicylinders with y radii)'
array 4 3*0.0
unit 15
com='7th cylinder: described as a zhemicylinder in the -y direction'
zhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 13.74 0.0 6.505 -6.505
unit 16
com='8th cylinder: described as a zhemicylinder in the +y direction'
zhemicyl+y 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 13.74 0.0 6.505 -6.505
global unit 17
com='complete 2c8 bare configuration'
array 5 3*0.0
end geometry
read array
com='array 1: 1st cylinder (unit 3) equal x portions of zhemicylinders'
ara=1 nux=2 nuy=1 nuz=1 fill 1 2 end fill
com='array 2: 2nd cylinder (unit 6) unequal x portions of zhemicylinders'
ara=2 nux=2 nuy=1 nuz=1 fill 4 5 end fill
com='array 3: 5th cylinder (unit 11) equal y portions of zhemicylinders'
ara=3 nux=1 nuy=2 nuz=1 fill 9 10 end fill
com='array 4: 6th cylinder (unit 14) unequal y portions of zhemicylinders'
ara=4 nux=1 nuy=2 nuz=1 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

SAMPLE PROBLEM 27 CASE 2C8 BARE AS MIXED XHEMICYLINDERS WITH ORIGINS

The physical representation of this sample problem is the critical experiment described in sample problem 1. This sample problem describes each of the eight units in the critical 2 × 2 × 2 array using hemicylinders whose axes are in the x direction. Origins are specified for each hemicylinder.

INPUT DATA

```
#kenova
sample problem 27  case 2c8 bare as mixed xhemicylinders with origins
read parameters
  fdn=yes lib=4  run=yes end parameters
read mixt mix=1 1092234 4.82716e-04 1092235 4.47971e-02
1092236 9.57231e-05 1092238 2.65767e-03  sct=2 end mixt
read geometry
unit 1
com='-y half of first cylinder'
xhemicyl-y 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
```

```
cuboid  0 1 6.505 -6.505 6.87 0.0 6.87 -6.87
unit 2
com='+y half of first cylinder'
xhemicyl+y 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
cuboid  0 1 6.505 -6.505 13.74 6.87 6.87 -6.87
unit 3
com='1st cylinder composed of equal portions (xhemicylinders with y radii)'
array 1 3*0.0
unit 4
com='-y portion (more than half) of second cylinder'
xhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 6.87 0.0
cuboid  0 1 6.505 -6.505 9.87 0.0 6.87 -6.87
unit 5
com='+y portion (less than half) of second cylinder'
xhemicyl+y 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 6.87 0.0
cuboid  0 1 6.505 -6.505 13.74 9.87  6.87 -6.87
unit 6
com='2nd cylinder composed of unequal portions (xhemicylinders with y radii)'
array 2 3*0.0
unit 7
com='3rd cylinder: described as a xhemicylinder in the -y direction'
xhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 6.505 -6.505 13.74 0.0 6.87 -6.87
unit 8
com='4th cylinder: described as a xhemicylinder in the +y direction'
xhemicyl+y 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 6.505 -6.505 13.74 0.0 6.87 -6.87
unit 9
com='-z half of fifth cylinder'
xhemicyl-z 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 0.0
unit 10
com='+z half of fifth cylinder'
xhemicyl+z 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 13.74 6.87
unit 11
com='5th cylinder composed of equal portions (xhemicylinders with z radii)'
array 3 3*0.0
unit 12
com='-z portion (more than half) of sixth cylinder'
xhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 9.87 0.0
unit 13
com='+z portion (less than half) of sixth cylinder'
xhemicyl+z 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 13.74 9.87
unit 14
com='6th cylinder composed of unequal portions (xhemicylinders with z radii)'
array 4 3*0.0
unit 15
com='7th cylinder: described as a xhemicylinder in the -z direction'
xhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 13.74 0.0
unit 16
com='8th cylinder: de3scribed as a xhemicylinder in the +z direction'
xhemicyl+z 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 13.74 0.0
global unit 17
com='complete 2c8 bare configuration'
array 5 3*0.0
end geometry
read array
com='array 1: 1st cylinder (unit 3) equal y portions of xhemicylinders'
ara=1 nux=1 nuy=2 nuz=1 fill 1 2 end fill
com='array 2: 2nd cylinder (unit 6) unequal y portions of xhemicylinders'
ara=2 nux=1 nuy=2 nuz=1 fill 4 5 end fill
```

```
com='array 3: 5th cylinder (unit 11) equal z portions of xhemicylinders'
ara=3 nux=1 nuy=1 nuz=2 fill 9 10 end fill
com='array 4: 6th cylinder (unit 14) unequal z portions of xhemicylinders'
ara=4 nux=1 nuy=1 nuz=2 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
gbl=5 ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

SAMPLE PROBLEM 28 CASE 2C8 BARE AS MIXED YHEMICYLINDERS WITH ORIGINS

The physical representation of this sample problem is the critical experiment described in sample problem 1. This sample problem describes each of the eight units in the critical 2 × 2 × 2 array using hemicylinders whose axes are in the y direction. Origins are specified for each hemicylinder.

INPUT DATA

```
#kenova
sample problem 28  case 2c8 bare as mixed yhemicylinders with origins
read parameters
  fdn=yes lib=4  run=yes end parameters
read mixt mix=1 1092234 4.82716e-04 1092235 4.47971e-02
1092236 9.57231e-05 1092238 2.65767e-03  sct=2 end mixt
read geometry
unit 1
com='-x half of first cylinder'
yhemicyl-x 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
cuboid  0 1 6.87 0.0 6.505 -6.505 6.87 -6.87
unit 2
com='+x half of unit 3'
yhemicyl+x 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
cuboid  0 1 13.74 6.87  6.505 -6.505 6.87 -6.87
unit 3
com='1st cylinder composed of equal portions (yhemicylinders with x radii)'
array 1 3*0.0
unit 4
com='-x portion (more than half) of second cylinder'
yhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 6.87 0.0
cuboid  0 1 9.87 0.0 6.505 -6.505 6.87 -6.87
unit 5
com='+x portion (less than half) of second cylinder'
yhemicyl+x 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 6.87 0.0
cuboid  0 1 13.74 9.87 6.505 -6.505 6.87 -6.87
unit 6
com='2nd cylinder composed of unequal portions (yhemicylinders with x radii)'
array 2 3*0.0
unit 7
com='3rd cylinder: described as a single yhemicylinder in the -x direction'
yhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 13.74 0.0 6.505 -6.505 6.87 -6.87
unit 8
com='4th cylinder: described as a single yhemicylinder in the +x direction'
yhemicyl+x 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 13.74 0.0 6.505 -6.505 6.87 -6.87
unit 9
com='-z half of fifth cylinder'
yhemicyl-z 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 0.0
unit 10
```

```
com='+z half of sixth cylinder'
yhemicyl+z 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 13.74 6.87
unit 11
com='5th cylinder composed of equal portions (yhemicylinders with z radii)'
array 3 3*0.0
unit 12
com='-z portion (more than half) of sixth cylinder'
yhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 9.87 0.0
unit 13
com='+z portion (less than half) of sixth cylinder'
yhemicyl+z 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 13.74 9.87
unit 14
com='6th cylinder composed of unequal portions (yhemicylinders with z radii)'
array 4 3*0.0
unit 15
com='7th cylinder: described as a yhemicylinder in the -z direction'
yhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 13.74 0.0
unit 16
com='8th cylinder: described as a yhemicylinder in the +z direction'
yhemicyl+z 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 13.74 0.0
global unit 17
com='complete 2c8 bare configuration'
array 5 3*0.0
end geometry
read array
com='array 1: 1st cylinder (unit 3) equal x portions of yhemicylinders'
ara=1 nux=2 nuy=1 nuz=1 fill 1 2 end fill
com='array 2: 2nd cylinder (unit 6) unequal x portions of yhemicylinders'
ara=2 nux=2 nuy=1 nuz=1 fill 4 5 end fill
com='array 3: 5th cyllinder (unit 11) equal z portions of yhemicylinders'
ara=3 nux=1 nuy=1 nuz=2 fill 9 10 end fill
com='array 4: 6th cylinder (unit 14) unequal z portions of yhemicylinders'
ara=4 nux=1 nuy=1 nuz=2 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
gbl=5 ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

SAMPLE PROBLEM 29 BARE CRITICAL SPHERE 3.4420-IN. RADIUS

This problem is a critical experiment[9] consisting of a critical Oralloy sphere. The density of the Oralloy is 18.747 g/cc, and the isotopic enrichment (wt %) is 93.21% ^{235}U, 5.7697% ^{238}U, 0.9844% ^{234}U, and 0.0359% ^{236}U. The critical radius was 8.74268 cm. A photograph of the experiment is given in Fig. F11.D.11. The support structure was ignored in the input data.

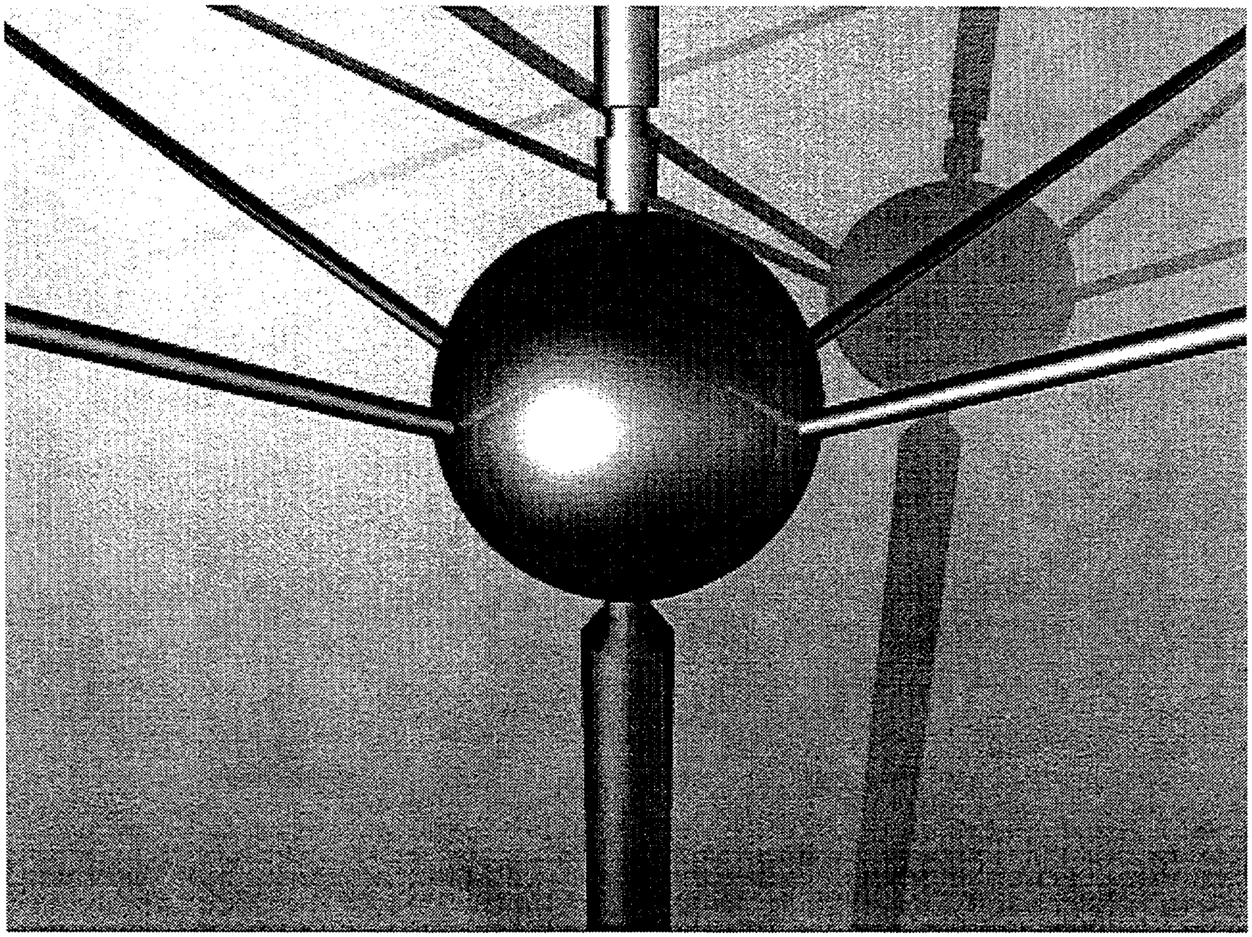


Figure F11.D.11 Critical oralloy sphere

INPUT DATA

```
#kenova
sample problem 29  bare critical sphere   3.4420" radius
read parameters
    fdn=yes lib=4
end parameters
read mixt
mix=16  16092235   4.47708e-02
        16092238   2.73631e-03
        16092234   4.74857e-04
        16092236   1.71704e-05

end mixt
read geometry
sphere   16 1 8.74268
end geometry
read plot   scr=yes lpi=10
ttl='x-y slice at z=0.0'
xul=-9 yul=9 zul=0.0 xlr=9 ylr=-9.0 zlr=0.0
uax=1 vdn=-1  nax=400 nch=  *'
end plot
end data
end
```

SAMPLE PROBLEM 30 BARE CRITICAL SPHERE Z HEMISPHERE MODEL 3.4420-IN. RADIUS

The physical representation of this sample problem is the critical experiment described in sample problem 29. This sample problem describes the sphere as two Z hemispheres, each with a chord and origin specified. One of the hemispheres is placed using the hole geometry option.

INPUT DATA

```
#kenova
sample problem 30   bare critical sphere    z hemisphere model 3.4420" radius
read parameters
    fdn=yes lib=4
end parameters
read mixt
mix=16  16092235   4.47708e-02
        16092238   2.73631e-03
        16092234   4.74857e-04
        16092236   1.71704e-05
end mixt
read geometry
unit 1
hemisphe+z   16 1 8.74268  chord +3.0 origin 8.9 8.9 8.9
global unit 2
hemisphe-z   16 1 8.74268  chord -3.0 origin 8.9 8.9 8.9
cuboid       0 1 17.8 0.0 17.8 0.0 17.8 0.0
hole 1 3*0.0
end geometry
read plot   scr=yes lpi=10
ttl=’y-z slice at x=8.9   mixture map’
xul=8.9 yul=-.5 zul=18.5 xlr=8.9 ylr=18.5 zlr=-0.5
vax=1 wdn=-1  nax=400 end plt1
ttl=’y-z slice at x=8.9   unit map’
pic=box   end plt2
end plot
end data
end
```

SAMPLE PROBLEM 31 BARE CRITICAL SPHERE X HEMISPHERE MODEL 3.4420-IN. RADIUS

The physical representation of this sample problem is the critical experiment described in sample problem 29. This sample problem describes the sphere as two X hemispheres, each with a chord and origin specified. One of the hemispheres is placed using the hole geometry option.

INPUT DATA

```
#kenova
sample problem 31   bare critical sphere    x hemisphere model 3.4420" radius
read parameters
    fdn=yes lib=4
end parameters
read mixt
mix=16  16092235   4.47708e-02
        16092238   2.73631e-03
        16092234   4.74857e-04
        16092236   1.71704e-05
```

```
end mixt
read geometry
unit 1
hemisphe-x   16 1 8.74268  chord +3.0
global unit 2
hemisphe+x   16 1 8.74268  chord -3.0 origin 8.9 8.9 8.9
cuboid        0 1 17.8 0.0 17.8 0.0 17.8 0.0
hole 1 3*8.9
end geometry
read plot     scr=yes lpi=10
ttl='x-y slice at z=8.9     mixture map'
xul=-0.5 yul=18.5 zul=8.9  xlr=18.5 ylr=-0.5 zlr=8.9
uax=1 vdn=-1  nax=400  end plt1
ttl='y-z slice at x=8.9   unit map'
pic=box   end plt2
end plot
end data
end
```

SAMPLE PROBLEM 32 BARE CRITICAL SPHERE Y HEMISPHERE MODEL 3.4420-IN. RADIUS

The physical representation of this sample problem is the critical experiment described in sample problem 29. This sample problem describes the sphere as two Y hemispheres, each with a chord and origin specified. One of the hemispheres is placed using the hole geometry option.

INPUT DATA

```
#kenova
sample problem 32   bare critical sphere    y hemisphere model 3.4420" radius
read parameters
    fdn=yes lib=4
end parameters
read mixt
mix=16  16092235   4.47708e-02
        16092238   2.73631e-03
        16092234   4.74857e-04
        16092236   1.71704e-05
end mixt
read geometry
unit 1
hemisphe-y   16 1 8.74268  chord +3.0 origin 8.9 9.9 10.9
global unit 2
hemisphe+y   16 1 8.74268  chord -3.0 origin 8.9 8.9 8.9
cuboid        0 1 17.8 0.0 17.8 0.0 17.8 0.0
hole 1 0.0 -1.0 -2.0
end geometry
read plot     scr=yes lpi=10
ttl='x-y slice at z=8.9     mixture map'
xul=-0.5 yul=18.5 zul=8.9  xlr=18.5 ylr=-0.5 zlr=8.9
uax=1 vdn=-1  nax=400  end plt1
ttl='y-z slice at x=8.9   unit map'
pic=box   end plt2
end plot
end data
end
```

SAMPLE PROBLEM 33 CRITICAL TRIANGULAR PITCHED ARRAY OF ANNULAR RODS

This sample problem represents a critical experiment[10,11] that consists of a partially flooded array of 19 low enriched uranium metal cylindrical annuli billets arranged in a triangular pitched array. The density of the uranium metal was 19.0 g/cc, and the isotopic enrichment in weight percent was 1.95% ^{235}U, 98.02% ^{238}U, 0.006% ^{236}U, and 0.002% ^{234}U. The cylindrical annuli had an inside diameter of 6.604 cm, an outside diameter of 18.288 cm, and were placed with a pitch of 20.828 cm. Each billet was 101.6 cm long. The array was positioned in a very large tank. This configuration was critical when the tank was filled to a height of 47.7 cm on a scale whose zero point was 0.6c m below the bottom of the billets. The bottom of the billets was 21.6 cm above the bottom of the tank. The tank and all support structures have been ignored in this model. The model utilizes only 15.24 cm of water reflector on all sides of the array. Figures F11.D.12 and F11.D.13 provide a representation of the model. A photograph of a single annular billet is shown in Fig. F11.D.14. See Appendix F11.D.4 for a discussion of modeling decisions and a summary of results obtained using several different cross-section libraries.

INPUT DATA

```
#kenova
sample problem 33   critical triangular pitched array of annular rods
read parameters  fdn=yes nub=yes lib=4
end parameters
read mixt
  sct=2
mix=17  17092235   9.49269e-04
        17092238   4.71245e-02
        17092234   9.77784e-07
        17092236   2.90843e-06
mix=18  18008016   3.33757e-02
        18001001   6.67514e-02
mix=19  19001001   6.67514e-02
        19008016   3.33757e-02
mix=20  20092235   9.49269e-04
        20092238   4.71245e-02
        20092234   9.77784e-07
        20092236   2.90843e-06
end mixt
read geom
unit 1
zhemicyl-x 18 1 3.302 47.7 0.6
zhemicyl-x 17 1 9.144 47.7 0.6
unit 2
zhemicyl-y 18 1 3.302 47.7 0.6
zhemicyl-y 17 1 9.144 47.7 0.6
unit 3
zhemicyl+x 18 1 3.302 47.7 0.6
zhemicyl+x 17 1 9.144 47.7 0.6
unit 4
zhemicyl+y 18 1 3.302 47.7 0.6  origin 0.0 -18.03758
zhemicyl+y 17 1 9.144 47.7 0.6  origin 0.0 -18.03758
cuboid     19 1 2p10.414 2p18.03758 47.7 0.6
hole      1       10.414    0.0      0.0
hole      2        0.0     18.03758  0.0
hole      3     -10.414     0.0      0.0
unit 5
cuboid     19 1 2p10.414 10.414 0.0 47.7 0.6
unit 6
zhemicyl-y 18 1 3.302 47.7 0.6
zhemicyl-y 17 1 9.144 47.7 0.6
```

```
cuboid     19 1 2p10.414 0.0 -10.414 47.7 0.6
unit 7
zhemicyl-y 18 1 3.302 47.7 0.6 origin 0.0 18.03758
zhemicyl-y 17 1 9.144 47.7 0.6 origin 0.0 18.03758
cuboid     19 1 2p10.414 2p18.03758 47.7 0.6
hole     3      -10.414 0.0 0.0
unit 8
zhemicyl+y 18 1 3.302 47.7 0.6 origin 0.0 -18.03758
zhemicyl+y 17 1 9.144 47.7 0.6 origin 0.0 -18.03758
cuboid     19 1 2p10.414 2p18.03758 47.7 0.6
hole     3      -10.414 0.0 0.0
unit 9
zhemicyl+y 18 1 3.302 47.7 0.6
zhemicyl+y 17 1 9.144 47.7 0.6
cuboid     19 1 2p10.414 10.414 0.0 47.7 0.6
unit 10
zhemicyl+y 18 1 3.302 47.7 0.6  origin 0.0 -18.03758
zhemicyl+y 17 1 9.144 47.7 0.6  origin 0.0 -18.03758
cuboid     19 1 2p10.414 2p18.03758 47.7 0.6
hole     1       10.414 0.0 0.0
unit 11
zhemicyl-y 18 1 3.302 47.7 0.6  origin 0.0 18.03758
zhemicyl-y 17 1 9.144 47.7 0.6  origin 0.0 18.03758
cuboid     19 1 2p10.414 2p18.03758 47.7 0.6
hole     1       10.414 0.0 0.0
unit 21
zhemicyl-x  0 1 3.302 102.2 47.7
zhemicyl-x 20 1 9.144 102.2 47.7
unit 22
zhemicyl-y  0 1 3.302 102.2 47.7
zhemicyl-y 20 1 9.144 102.2 47.7
unit 23
zhemicyl+x  0 1 3.302 102.2 47.7
zhemicyl+x 20 1 9.144 102.2 47.7
unit 24
zhemicyl+y  0 1 3.302 102.2 47.7  origin 0.0 -18.03758
zhemicyl+y 20 1 9.144 102.2 47.7  origin 0.0 -18.03758
cuboid      0 1 2p10.414 2p18.03758 102.2 47.7
hole      21      10.414    0.0      0.0
hole      22       0.0     18.03758  0.0
hole      23     -10.414    0.0      0.0
unit 25
cuboid      0 1 2p10.414 10.414 0.0 102.2 47.7
unit 26
zhemicyl-y  0 1 3.302 102.2 47.7
zhemicyl-y 20 1 9.144 102.2 47.7
cuboid      0 1 2p10.414 0.0 -10.414 102.2 47.7
unit 27
zhemicyl-y  0 1 3.302 102.2 47.7  origin 0.0 18.03758
zhemicyl-y 20 1 9.144 102.2 47.7  origin 0.0 18.03758
cuboid      0 1 2p10.414 2p18.03758 102.2 47.7
hole     23      -10.414 0.0 0.0
unit 28
zhemicyl+y  0 1 3.302 102.2 47.7  origin 0.0 -18.03758
zhemicyl+y 20 1 9.144 102.2 47.7  origin 0.0 -18.03758
cuboid      0 1 2p10.414 2p18.03758 102.2 47.7
hole     23      -10.414 0.0 0.0
unit 29
zhemicyl+y  0 1 3.302 102.2 47.7
zhemicyl+y 20 1 9.144 102.2 47.7
cuboid      0 1 2p10.414 10.414 0.0 102.2 47.7
unit 30
zhemicyl+y  0 1 3.302 102.2 47.7  origin 0.0 -18.03758
zhemicyl+y 20 1 9.144 102.2 47.7  origin 0.0 -18.03758
```

```
cuboid      0 1 2p10.414 2p18.03758 102.2 47.7
hole     21       10.414 0.0 0.0
unit 31
zhemicyl-y  0 1 3.302 102.2 47.7  origin 0.0 18.03758
zhemicyl-y 20 1 9.144 102.2 47.7  origin 0.0 18.03758
cuboid      0 1 2p10.414 2p18.03758 102.2 47.7
hole     21       10.414 0.0 0.0
unit 32
com='flooded portion of array with 15.24 cm of water in x and y'
array 1 2*0.0 0.6
replicate 19 1 4r15.24 0.0 0.6 1
replicate 19 2 5r0.0 3.0 7
unit 33
com='unflooded upper portion of array'
array 2 3*0.0
replicate 0 1 4r15.24 2*0.0 1
global
unit 34
array 3 -67.31 -61.72916 -21.0
end geom
read bias id=500 2 8 end bias
read array
ara=1 nux=5 nuy=4 nuz=1  fill
5 3r6 5      11 3r4 7      10 3r4 8       5 3r9 5     end fill
ara=2 nux=5 nuy=4 nuz=1  fill
25 3r26 25   31 3r24 27    30 3r24 28     25 3r29 25  end fill
ara=3 nux=1 nuy=1 nuz=2 fill 32 33 end fill
end array
read start nst=1 xsm=-52 xsp=52 ysm=-47 ysp=47 zsm=0.6 zsp=47.7
end start
read plot
scr=yes lpi=10
ttl='x-y plot of pins at z=45.0'
xul=-52.0 yul=47.0 zul=45.0  xlr=52.0 ylr=-47.0 zlr=45.0
uax=1.0  vdn=-1.0 nax=400 end plt1
ttl='x-z plot of pins at y=0.0'
xul=-52.0 yul=0.0 zul=102.7 xlr=52 ylr=0.0 zlr=-3.0
uax=1.0 wdn=-1.0 nax=400  end plt2
ttl='x-z plot at y=0.0'
xul=-68.0 yul=0.0 zul=102.7 xlr=70.0 ylr=0.0 zlr=-25.0
uax=1.0 wdn=-1.0 nax=400  end plt3
end plot
end data
end
```

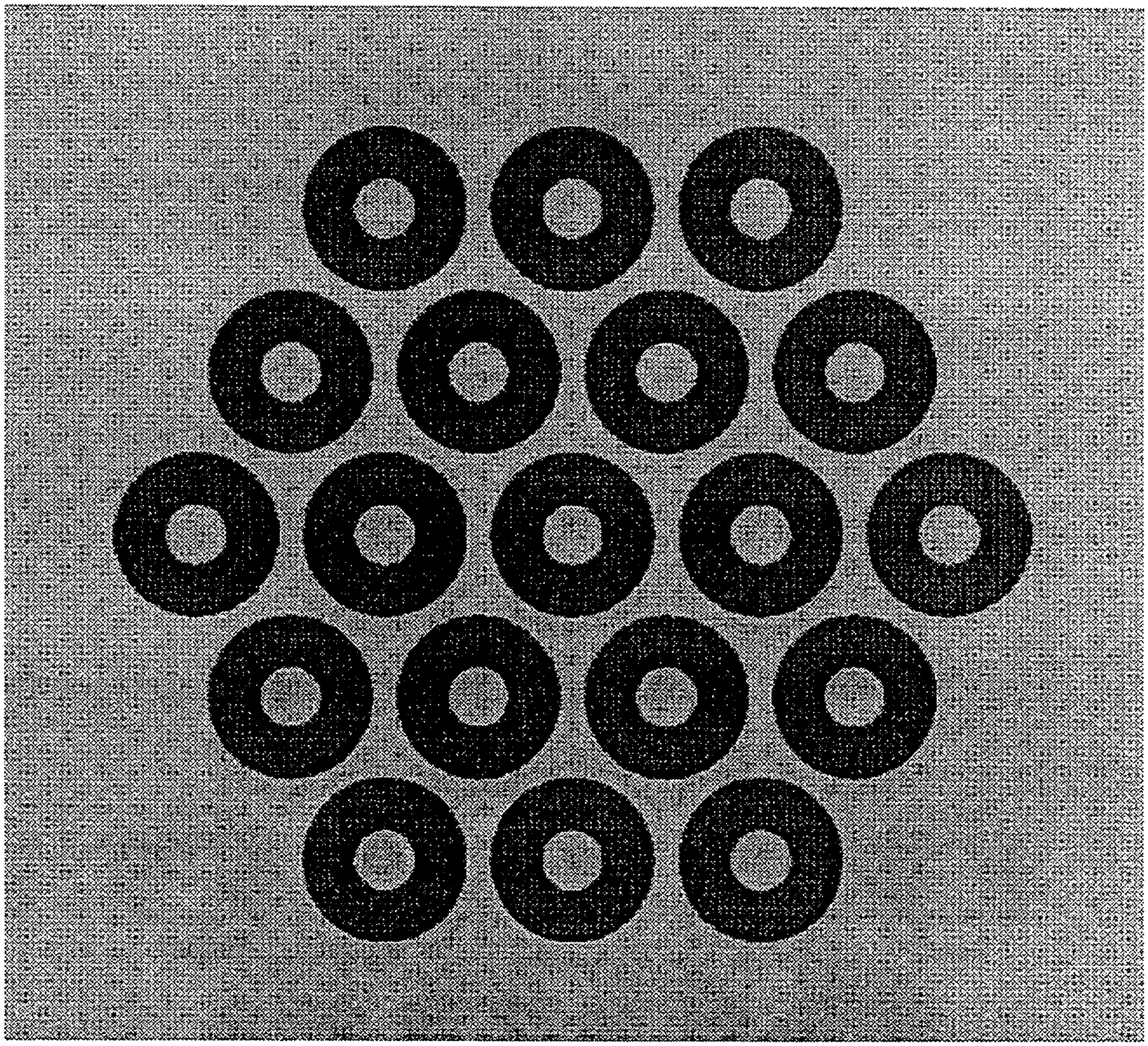


Figure F11.D.12 Horizontal slice through a critical triangular-pitched array of partially flooded 1.95% enriched uranium metal annular billets

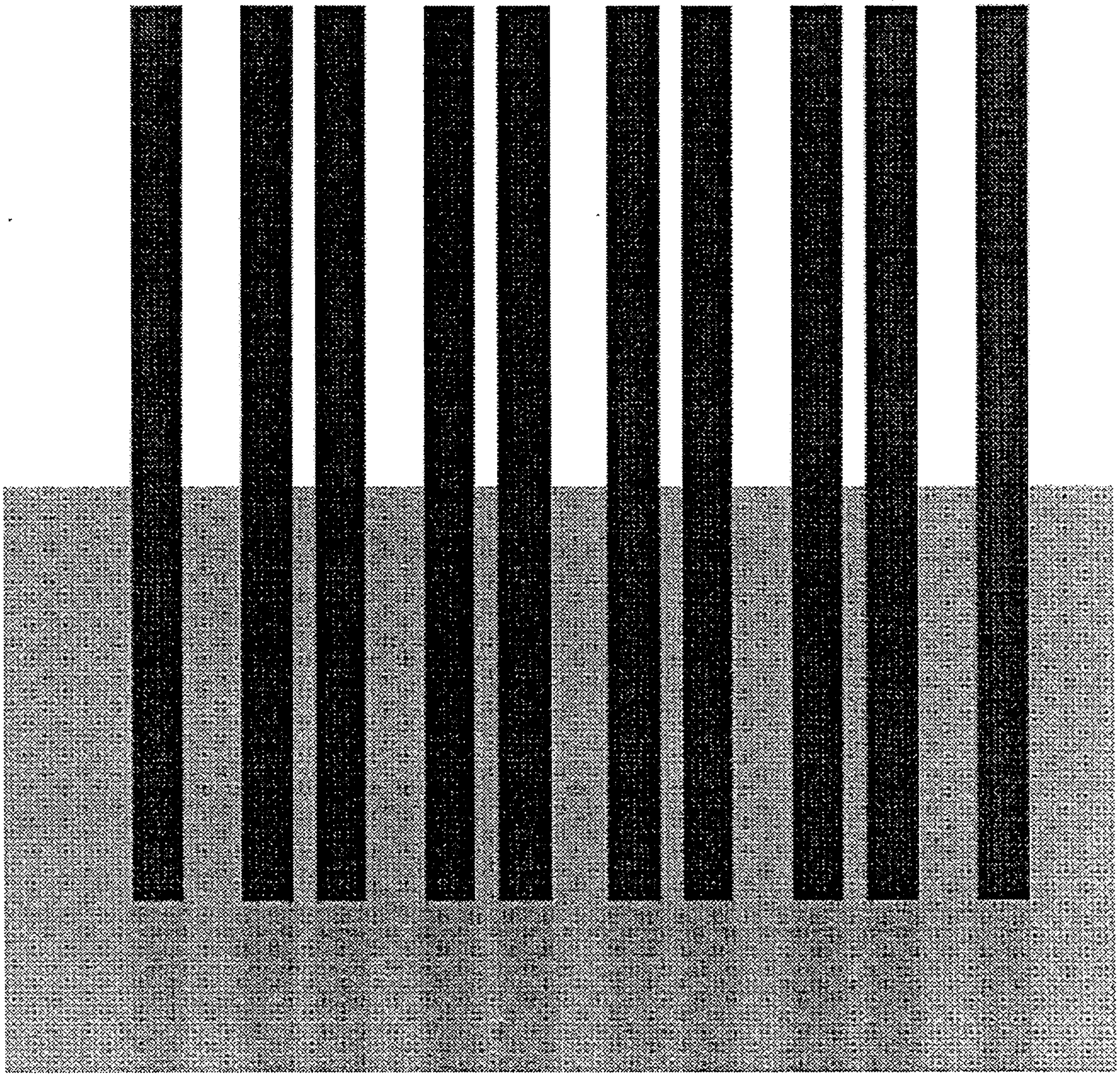


Figure F11.D.13 Vertical slice through the center of a critical triangular-pitched array of partially flooded 1.95% enriched uranium metal annular billets

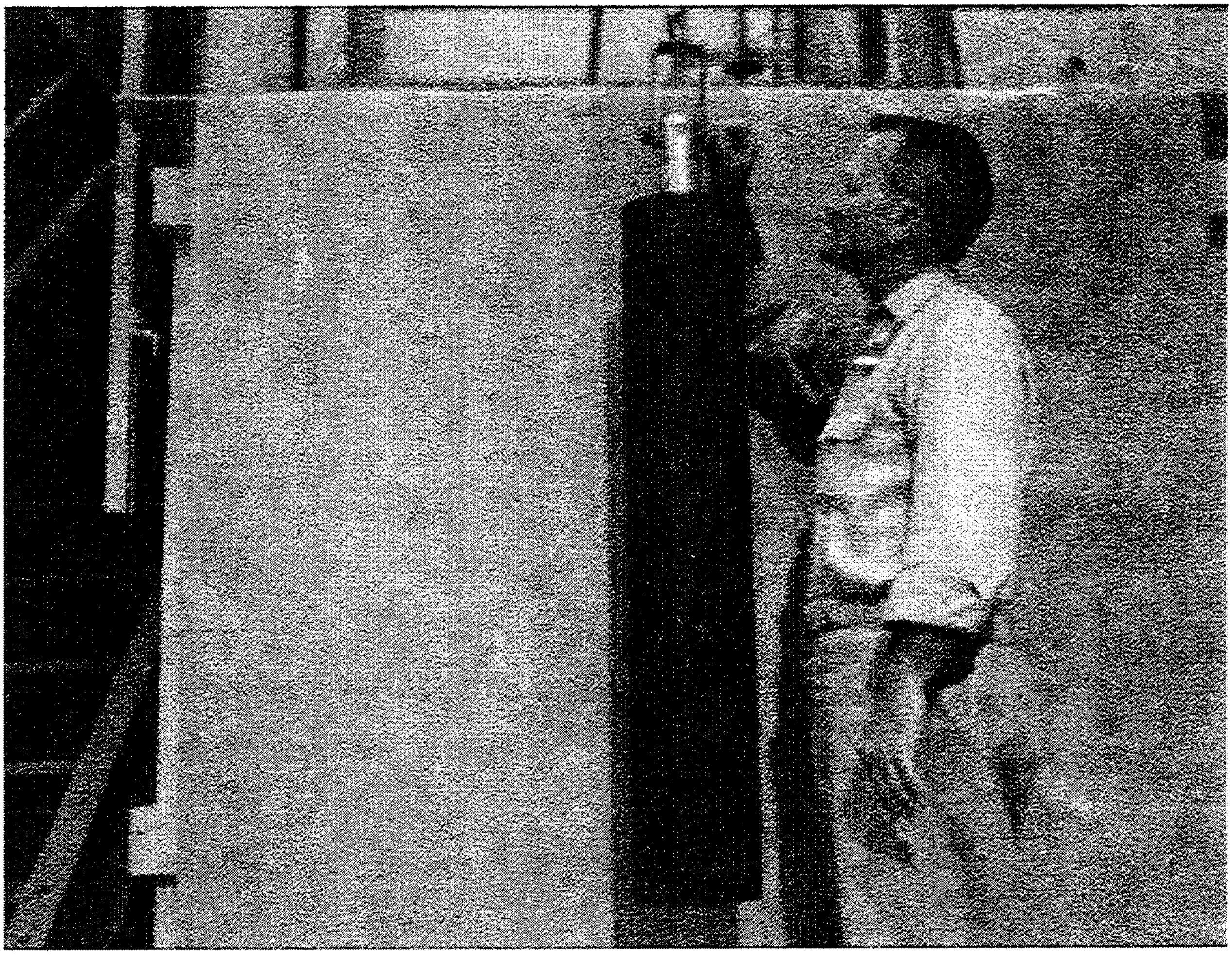


Figure F11.D.14 1.95% enriched uranium metal annular billet used in critical experiments

F11.D.2 SAMPLE PROBLEM WORKING FORMAT LIBRARY

The KENO V.a sample problems use nuclide IDs that are consistent with the SCALE CSAS nuclide ID naming convention. Nuclides are identified by the ZA number plus 1000000 times the mixture number. CSASN or BONAMI and NITAWL can be used to create a problem-dependent working format cross-section library suitable for use with the sample problems. CSASN or BONAMI and NITAWL can (1) be run alone with logical unit 4 of the job control language saved for later use with the KENO V.a sample problems, or (2) be placed in the job stream in front of the KENO V.a sample problems.

The CSASN SCALE control module is easy to use because it calculates the necessary resonance data required to create the problem-dependent AMPX working format library. It is recommended that CSASN be used if it is operational at the installation where the problems are to be run. CSASN is one of the control modules associated with CSAS4 and is automatically operational if CSAS4 is operational. See Sect. C4 of the SCALE manual for assistance in using CSASN. If the SCALE DRIVER and the control modules are not operational at your installation, BONAMI and NITAWL should be used to create the problem-dependent AMPX working format cross-section library. See Sect. F1 of the SCALE manual for assistance in using BONAMI. Obtaining accurate resonance data for NITAWL may be difficult. See Sect. F2 of the SCALE manual for assistance in using NITAWL.

The KENO V.a sample problem input data are independent of energy group structure. However, if the desired cross-section library contains Bondarenko data, BONAMI must be run prior to NITAWL to provide the Bondarenko calculations for resonance self-shielding corrections. To change cross-section libraries, change the first entry in the BONAMI 0$$ array. If CSASN is used to create the AMPX working format cross-section library, the Bondarenko corrections are automatically performed when necessary. To use a different energy group structure, simply supply the desired master cross-section library name in the CSASN or CSAS25 data. See Sects. M7.5.3, M7.5.4, and C4.5.2 for additional information and examples. See Sect. M4 for information about the master format cross-section libraries that are available in SCALE.

Data for CSASN and BONAMI and NITAWL are provided to create a problem-dependent AMPX working format cross-section library suitable for use with the KENO V.a sample problems. These data include all of the mixtures used in the 33 KENO V.a sample problems and will create an AMPX working format cross-section library with nuclide IDs matching those in the KENO V.a sample problem mixing tables. *This cross-section library is problem-specific and is not appropriate for use with other problems.*

F11.D.2.1 CSASN Data

The CSASN input data to produce an AMPX working format cross-section library for the KENO V.a sample problems are given on the following page.

```
#csasn
csasn to prepare  44 group working format xsec lib for kenova smp prbs
44group  latticecell
' uranium metal for smp prbs 1,2,3,4,5,6,7,8,9,10,11,12,19,22,23,24,25,26,27,28
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
' uranyl nitrate solution for smp prbs 12,18,19
solnuo2(no3)2  2 415 9.783-3 spg=1.555 1.0 293 92235 92.6 92238 5.9
                                               92234 1.0  92236 0.5 end
' uranium metal for smp prbs 13,14
uranium  3 den=18.69 1 293  92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
' uranium metal for smp prb 15
uranium  4 den=18.794 1 293 92235 97.67 92238 0.98
                           92234 1.17  92236 0.20 end
' uranyl fluoride solution for smp prb 16
```

```
solnuo2f2      5 578.7 0 1.0 293 92235 93.2 92238 6.8 end
' borated uranyl fluoride solution for smp prb 16
solnuo2f2      6 578.7 0 1.0 293 92235 93.2 92238 6.8 end
boron          6 den=.0266 end
' uranyl fluoride solution for smp prb 17
solnuo2f2  7 133 0 1.0 293 92235 93.0 92238 7 end
' uranyl fluoride solution for smp prb 20
solnuo2f2  8 576.87 0 1.0 293 92235 93.2 92238 6.8 end
' uranyl fluoride solution for smp prb 21
solnuo2f2  9 494 0 spg= 1.56 1 293 92235 4.89 92238 95.09 92234 0.02 end
' paraffin for smp prbs 3,4,18
para(h2o) 10 end
' plexiglas for smp prbs 12,15,18,19
plexiglas 11 end
' water for smp prbs 15
h2o 12 end
' pyrex glass for smp prb 16
pyrex 13 end
' aluminum for smp prb 20,21
al 14 end
' low density water for smp prb 18
h2o 15 1-9 end
' uranium metal for smp prbs 29 - 32
uranium  16 den=18.747 1 293  92235 93.21 92238 5.7697 92234 0.9844
                              92236 0.0359 end
'uranium metal for water moderated portion of smp prb 33
uranium  17 den=19.0 1 293  92235 1.95 92238 98.042 92234 0.002 92236 0.006  end
'internal (2nd) moderator water for smp prb 33
h2o      18 end
'external moderator water and reflector for smp prb 33
h2o      19 end
'uranium metal for bare portion of smp prb 33
uranium  20 den=19.0 1 293  92235 1.95 92238 98.042 92234 0.002 92236 0.006  end
end comp
'latticecell data for samp prb 33
atriangpitch  20.828 18.288 17 19 18 6.604 end
more data  res=20 slab 5.842  dan(20)=1.0  end more
end
```

F11.D.2.2 BONAMI/NITAWL Data

The BONAMI and NITAWL input data to produce an AMPX working format cross-section library for the KENO V.a sample problems are given below. If the SCALE DRIVER is not used, the first and last records of the data for BONAMI and NITAWL should be omitted. The first record in each input data set (=NITAWL in NITAWL) and (=BONAMI in BONAMI) are used to initiate that program in SCALE. The last record in each input data set (END starting in column 1) indicates the end of the data for the program in SCALE.

```
=bonami
0$$ 84 a4 1
1$$ 2 20 79 0 0 0
2** .0001 1.35 1t
3$$ 1 2 3 4 5 6 7 8 9 16 17 20
    1 2 3 4 5 6 7 8 9 16 17 20
    1 2 3 4 9 16 17 20
    1 2 3 4 16 17 20
    2 2 5 6 7 8 9 11 12 13 15 18 19 2 5 6 7 8 9 10 11 12 15
    18 19 5 6 7 8 9 6 13 6 13 10 11 13 14 13 13
4$$ 92235 92235 92235 92235 92235 92235 92235 92235 92235 92235 92235
```

```
    92235
    92238 92238 92238 92238 92238 92238 92238 92238 92238 92238 92238
    92238
    92234 92234 92234 92234 92234 92234 92234 92234
    92236 92236 92236 92236 92236 92236 92236
     7014  8016  8016  8016  8016  8016  8016  8016  8016
     8016  8016  8016  8016  1001  1001  1001  1001  1001  1001  1001
     1001  1001  1001  1001  1001  9019  9019  9019  9019  9019  5010
     5010  5011  5011  6012  6012 13027 13027 14000 11023
5**   4.47971e-02 9.84601e-04 4.46299e-02 4.70211e-02 1.38188e-03
      1.38188e-03 3.16910e-04 1.37751e-03 6.18924e-05 4.47708e-02
      9.49269e-04 9.49269e-04
      2.65767e-03 6.19414e-05 2.64776e-03 4.65936e-04 9.95504e-05
      9.95504e-05 2.35521e-05 9.92356e-05 1.18834e-03 2.73631e-03
      4.71245e-02 4.71245e-02
      4.82717e-04 1.06784e-05 4.80915e-04 5.65802e-04 2.54224e-07
      4.74857e-04 9.77784e-07 9.77784e-07
      9.57232e-05 5.29386e-06 9.53660e-05 9.58967e-05 1.71704e-05
      2.90843e-06 2.90843e-06
      2.13092e-03 3.74130e-02 3.32040e-02 3.32040e-02 3.34202e-02
      3.32049e-02 3.32845e-02 1.42047e-02 3.33757e-02 4.49174e-02
      3.33757e-11 3.33757e-02 3.33757e-02 5.77964e-02 6.04824e-02
      6.04824e-02 6.54785e-02 6.05029e-02 6.15670e-02 8.26407e-02
      5.68187e-02 6.67514e-02 6.67514e-11 6.67514e-02 6.67514e-02
      2.96286e-03 2.96286e-03 6.80924e-04 2.95349e-03 2.50098e-03
      2.92804e-04 9.08242e-04 1.18870e-03 3.68719e-03 3.97311e-02
      3.55117e-02 4.97719e-04 6.02374e-02 1.80268e-02 2.39502e-03
10$$  1092235  2092235  3092235  4092235  5092235  6092235  7092235
      8092235  9092235 16092235 17092235 20092235
      1092238  2092238  3092238  4092238  5092238  6092238  7092238
      8092238  9092238 16092238 17092238 20092238
      1092234  2092234  3092234  4092234  9092234 16092234 17092234
     20092234
      1092236  2092236  3092236  4092236 16092236 17092236 20092236
      2007014
      2008016  5008016  6008016  7008016  8008016  9008016 11008016
     12008016 13008016 15008016 18008016 19008016
      2001001  5001001  6001001  7001001  8001001  9001001 10001001
     11001001 12001001 15001001 18001001 19001001
      5009019  6009019  7009019  8009019  9009019
      6005010 13005010  6005011 13005011 10006012 11006012 13013027
     14013027
     13014000 13011023
6$$ 18 17 19 14i1 16 20
7** 3.302 9.144 10.9355 15i15.9355 95.9355
8** f293 9** f0 a2 .0952248 e
11$$ f0 2t
end
=nitawl
0$$ 1 e  1$$ 2001 79 a8 40 2 e  1t
2$$
 1092235  2092235  3092235  4092235  5092235  6092235  7092235  8092235
 9092235 16092235 17092235 20092235
 1092238  2092238  3092238  4092238  5092238  6092238  7092238  8092238
 9092238 16092238 17092238 20092238
 1092234  2092234  3092234  4092234  9092234 16092234 17092234 20092234
 1092236  2092236  3092236  4092236 16092236 17092236 20092236
 2007014
 2008016  5008016  6008016  7008016  8008016  9008016 11008016 12008016
13008016 15008016 18008016 19008016
 2001001  5001001  6001001  7001001  8001001  9001001 10001001 11001001
12001001 15001001 18001001 19001001
 5009019  6009019  7009019  8009019  9009019
 6005010 13005010
 6005011 13005011
10006012 11006012
13013027 14013027
13014000
13011023
3**
 1092235 293.00   0 0.0 0.0 0.0 4.47971e-02  1 238.051 4.92413e-01
 1 234.370 1.35581e-01  1   1.0000
 2092235 293.00   0 0.0 0.0 0.0 9.84601e-04  1   1.008 1.19749e+03
 1  15.767 1.64828e+02  1   1.0000
 3092235 293.00   0 0.0 0.0 0.0 4.46299e-02  1 238.051 4.92413e-01
 1 234.370 1.35581e-01  1   1.0000
 4092235 293.00   0 0.0 0.0 0.0 4.70211e-02  1 234.041 1.26346e-01
 1 237.634 1.03660e-01  1   1.0000
 5092235 293.00   0 0.0 0.0 0.0 1.38188e-03  1   1.008 8.92870e+02
 1  16.294 9.84222e+01  1   1.0000
 6092235 293.00   0 0.0 0.0 0.0 1.38188e-03  1   1.008 8.92870e+02
 1  15.951 1.02996e+02  1   1.0000
 7092235 293.00   0 0.0 0.0 0.0 3.16910e-04  1   1.008 4.21496e+03
 1  16.068 4.03814e+02  1   1.0000
 8092235 293.00   0 0.0 0.0 0.0 1.37751e-03  1   1.008 8.96007e+02
 1  16.294 9.87105e+01  1   1.0000
 9092235 293.00   0 0.0 0.0 0.0 6.18924e-05  1   1.008 2.02927e+04
 1  17.317 2.32155e+03  1   1.0000
16092235 293.00   0 0.0 0.0 0.0 4.47708e-02  1 238.051 5.07281e-01
 1 234.110 1.15394e-01  1   1.0000
17092235 293.00   2 9.14400e+00 4.10601e-03 3.30200e+00 9.49269e-04
 1 238.051 4.12036e+02  1 235.538 4.29860e-02  1   1.0000
20092235 293.00   1 5.84200e+00 1.00000e+00 0.00000e+00 9.49269e-04
 1 238.051 4.12036e+02  1 235.538 4.29860e-02  1   1.0000
 1092238 293.00   0 0.0 0.0 0.0 2.65767e-03  1 235.044 1.76985e+02
 1 234.370 2.28531e+00  1   1.0000
 2092238 293.00   0 0.0 0.0 0.0 6.19414e-05  1   1.008 1.90349e+04
 1  16.689 2.77866e+03  1   1.0000
 3092238 293.00   0 0.0 0.0 0.0 2.64776e-03  1 235.044 1.76985e+02
 1 234.370 2.28531e+00  1   1.0000
 4092238 293.00   0 0.0 0.0 0.0 4.65936e-04  1 235.044 1.05963e+03
 1 234.329 1.49116e+01  1   1.0000
 5092238 293.00   0 0.0 0.0 0.0 9.95504e-05  1   1.008 1.23941e+04
 1  17.868 1.50367e+03  1   1.0000
 6092238 293.00   0 0.0 0.0 0.0 9.95504e-05  1   1.008 1.23941e+04
 1  17.427 1.56716e+03  1   1.0000
 7092238 293.00   0 0.0 0.0 0.0 2.35521e-05  1   1.008 5.67151e+04
 1  16.448 5.56657e+03  1   1.0000
 8092238 293.00   0 0.0 0.0 0.0 9.92356e-05  1   1.008 1.24377e+04
 1  17.862 1.50767e+03  1   1.0000
 9092238 293.00   0 0.0 0.0 0.0 1.18834e-03  1   1.008 1.05690e+03
 1  16.245 1.13160e+02  1   1.0000
16092238 293.00   0 0.0 0.0 0.0 2.73631e-03  1 235.044 1.71798e+02
 1 234.110 1.88805e+00  1   1.0000
```

```
17092238 293.00   2 9.14400e+00 4.10601e-03 3.30200e+00 4.71245e-02
 1 235.044 2.11511e-01  1 235.538 8.65905e-04  1   1.0000
20092238 293.00   1 5.84200e+00 1.00000e+00 0.00000e+00 4.71245e-02
 1 235.044 2.11511e-01  1 235.538 8.65905e-04  1   1.0000
 1092234 293.00   0 0.0 0.0 0.0 4.82717e-04  1 235.044 9.74421e+02
 1 237.963 4.77791e+01  1   1.0000
 2092234 293.00   0 0.0 0.0 0.0 1.06784e-05  1   1.008 1.10414e+05
 1  16.727 1.61556e+04  1   1.0000
 3092234 293.00   0 0.0 0.0 0.0 4.80915e-04  1 235.044 9.74422e+02
 1 237.963 4.77791e+01  1   1.0000
 4092234 293.00   0 0.0 0.0 0.0 5.65802e-04  1 235.044 8.72604e+02
 1 237.634 8.61464e+00  1   1.0000
 9092234 293.00   0 0.0 0.0 0.0 2.54224e-07  1   1.008 4.94039e+06
 1  17.392 5.67741e+05  1   1.0000
16092234 293.00   0 0.0 0.0 0.0 4.74857e-04  1 235.044 9.89970e+02
 1 238.035 4.82075e+01  1   1.0000
17092234 293.00   2 9.14400e+00 4.10601e-03 3.30200e+00 9.77784e-07
 1 238.051 4.00020e+05  1 235.047 1.02250e+04  1   1.0000
20092234 293.00   1 5.84200e+00 1.00000e+00 0.00000e+00 9.77784e-07
 1 238.051 4.00020e+05  1 235.047 1.02250e+04  1   1.0000
 1092236 293.00   0 0.0 0.0 0.0 9.57232e-05  1 235.044 4.91385e+03
 1 237.291 2.83392e+02  1   1.0000
 2092236 293.00   0 0.0 0.0 0.0 5.29386e-06  1   1.008 2.22720e+05
 1  16.732 3.25986e+04  1   1.0000
 3092236 293.00   0 0.0 0.0 0.0 9.53660e-05  1 235.044 4.91385e+03
 1 237.291 2.83392e+02  1   1.0000
 4092236 293.00   0 0.0 0.0 0.0 9.58967e-05  1 235.044 5.14847e+03
 1 235.606 1.02279e+02  1   1.0000
16092236 293.00   0 0.0 0.0 0.0 1.71704e-05  1 235.044 2.73781e+04
 1 237.319 1.61309e+03  1   1.0000
17092236 293.00   2 9.14400e+00 4.10601e-03 3.30200e+00 2.90843e-06
 1 238.051 1.34482e+05  1 235.043 3.43057e+03  1   1.0000
20092236 293.00   1 5.84200e+00 1.00000e+00 0.00000e+00 2.90843e-06
 1 238.051 1.34482e+05  1 235.043 3.43057e+03  1   1.0000
13011023 293.00   0 0.0 0.0 0.0 2.39502e-03  1  15.995 7.03292e+01
 1  18.466 2.41785e+01  1   1.0000
3$$
 1092235 14s 2092235  14s 3092235  14s 4092235 14s 5092235  14s
 6092235 14s 7092235  14s 8092235  14s 9092235 14s 16092235 14s
17092235 14s 20092235 14s
 1092238 14s 2092238  14s 3092238  14s 4092238 14s 5092238  14s
 6092238 14s 7092238  14s 8092238  14s 9092238 14s 16092238 14s
17092238 14s 20092238 14s
 1092234 14s 2092234  14s 3092234  14s 4092234 14s 9092234  14s
16092234 14s 17092234 14s 20092234 14s
 1092236 14s 2092236  14s 3092236  14s 4092236 14s 16092236 14s
17092236 14s 20092236 14s
13011023 14s e
 3t
end
```

F11.D.3 KENO V.a SAMPLE PROBLEMS MODELED AS CSAS25

Note that the KENO V.a sample problems can be easily converted to run using CSAS25. The CSAS25 input data for the KENO V.a sample problems are given below.

Sample Problem 1

```
=csas25
sample problem 1  case 2c8 bare
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 1  case 2c8 bare
read parameters   flx=yes fdn=yes far=yes          end parameters
read geometry
cylinder 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
end geometry
read array  nux=2 nuy=2 nuz=2  end array
end data
end
```

Sample Problem 2

```
=csas25
sample problem 2  2c8 bare with 8 unit types matrix calculation
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 2  2c8 bare with 8 unit types matrix calculation
read param              flx=yes fdn=yes
mku=yes fmu=yes mkp=yes fmp=yes
end param  read geometry  unit 1
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 2
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 3  cylinder  1 1 5.748 5.3825 -5.3825 cuboid  0 1 6.87 -6.87 6.87
-6.87 6.505 -6.505
box type 4  cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
box type  5
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 6
cylinder  1 1 5.748 5.3825 -5.3825  cuboid  0 1 6.87 -6.87 6.87 -6.87
6.505 -6.505 unit 7  cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 8
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505 end geom
read array  nux=2 nuy=2 nuz=2  loop  10*1 3*2 7*1  3 1 1 1 2 2 1
1 1 1  4 2 2 1 2 2 1 1 1 1  5 6*1 2 2 1  6 2 2 1 1 1 1 2 2 1
7  1 1 1 2 2 1 2 2 1  8 2 2 1 2 2 1 2 2 1 end array
end data
end
```

Sample Problem 3

```
=csas25
sample problem 3  2c8  15.24 cm paraffin refl
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
para(h2o) 2 end
end comp
sample problem 3  2c8  15.24 cm paraffin refl
read param             flx=yes fdn=yes        pwt=yes end param
read array nux=2 nuy=2 nuz=2 end array
read geom
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 11.74 -11.74 11.74 -11.74 11.375 -11.375
core  0 1 -23.48 -23.48 -22.75
cuboid  2 2 26.48 -26.48 26.48 -26.48 25.75 -25.75
cuboid  2 3 29.48 -29.48 29.48 -29.48 28.75 -28.75
cuboid  2 4 32.48 -32.48 32.48 -32.48 31.75 -31.75
cuboid  2 5 35.48 -35.48 35.48 -35.48 34.75 -34.75
cuboid  2 6 38.72 -38.72 38.72 -38.72 37.99 -37.99
end geom
read bias  id=400 2 6  end bias  end data
end
```

Sample Problem 4

```
=csas25
sample problem 4  2c8 15.24 cm paraffin refl automatic refl
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
para(h2o) 2 end
end comp

sample problem 4  2c8 15.24 cm paraffin refl automatic refl
read param    pwt=yes           flx=yes fdn=yes end param
read geometry  cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 11.74 -11.74 11.74 -11.74 11.375 -11.375
core  0 1 -23.48 -23.48 -22.75
reflector  2 2 6*3.0 5
reflector  2 7 6*.24 1
end geom
read arra  nux=2 nuy=2 nuz=2 end array
read bias  id=400 2 7  end bias
end data
end
```

Sample Problem 5

```
=csas25
sample problem 5  2c8 12 inch paraffin albedo reflector
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 5  2c8 12 inch paraffin albedo reflector
read para
 flx=yes far=yes fdn=yes           end para
read array  nux=2 nuy=2 nuz=2 end array
read bounds  all=paraffin  end bounds
read geom  cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 11.74 -11.74 11.74 -11.74 11.375 -11.375  end geom
end data
end
```

Sample Problem 6

```
=csas25
sample problem 6  one 2c8 unit (single unit)
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 6  one 2c8 unit (single unit)
read para            flx=yes fdn=yes far=yes         end para
read geometry  cylinder  1 1 5.748 5.3825 -5.3825
end geometry  end data
end
```

Sample Problem 7

```
=csas25
sample problem 7  bare 2c8 using specular reflection
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 7  bare 2c8 using specular reflection
read para                    flx=yes fdn=yes  far=yes    end parameters
read geom  cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
end geom
read bounds +fc=specular end bounds  end data
end
```

Sample Problem 8

```
=csas25
sample problem 8  infinitely long cylinder from 2c8 unit
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 8  infinitely long cylinder from 2c8 unit
read param                      end param
read geometry  cylinder  1 1 5.748 10.0 -10.0
cuboid  0 1 6.87 -6.87 6.87 -6.87 10.0 -10.0
end geometry
read bounds zfc=mirror  end bounds
end data
end
```

Sample Problem 9

```
=csas25
sample problem 9  infinite array of 2c8 units
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 9  infinite array of 2c8 units
read param             gen=33 end param
read boun  all=mir  end boun
read geom
cylinder  1 1 5.748 5.3825 -5.3825  cuboid  0 1 6.87 -6.87 6.87 -6.87
6.505 -6.505  end geom  end data
end
```

Sample Problem 10

```
=csas25
sample problem 10  case 2c8 bare  write restart
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 10  case 2c8 bare  write restart
read parameters  flx=yes fdn=yes far=yes           res=5 wrs=94
end parameters
read geometry
cylinder 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
end geometry
read array  nux=2 nuy=2 nuz=2  end array
end data
end
```

Sample Problem 11

```
=kenova
sample problem 11  2c8 bare  read restart data
read param   beg=51           rst=94 res=0 end param
end data
end
```

Sample Problem 12

```
=csas25
sample problem 12 4 aqueous 4 metal mixed units
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
solnuo2(no3)2  2 415 9.783-3 spg=1.555 1.0 293 92235 92.6 92238 5.9
                                               92234 1.0  92236 0.5 end
plexiglas 3 end
end comp
sample problem 12 4 aqueous 4 metal mixed units
read param                    flx=yes fdn=yes    nub=yes smu=yes mkp=yes
mku=yes fmp=yes fmu=yes end param
read geom
box type 1
cylinder  2 1 9.525 8.89 -8.89
cylinder  3 1 10.16 9.525 -9.525
cuboid  0 1 10.875 -10.875 10.875 -10.875 10.24 -10.24
box type  2
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.59 -15.16 6.59 -15.16 6.225 -14.255
box type  3
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.59 -15.16 15.16 -6.59 6.225 -14.255
box type  4
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.59 -15.16 6.59 -15.16 14.255 -6.225
box type  5
cylinder  1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.59 -15.16 15.16 -6.59 14.255 -6.225
end geom  read array  nux=2 nuy=2 nuz=2  loop
1 3r2 1 2 1 1 2 1  2 9r1  3 3r1 2 2 1 3r1
4 6r1 2 2 1  5 3r1 2 2 1 2 2 1
end array
end data
end
```

Sample Problem 13

```
=csas25
sample problem 13  two cuboids in a cylindrical annulus
44group inf
uranium  1 den=18.69 1 293  92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 13  two cuboids in a cylindrical annulus
read param              end param
read geom
unit 1
cuboid  1 1 6.35 -6.35 6.35 -6.35 7.62 0.0
cylinder  0 1 13.97 7.62 0.0 orig -6.0934 0.0
cylinder  1 1 19.05 7.62 0.0 orig -6.0934 0.0
cuboid  0 1 12.9566 -25.1434 19.05 -19.05 7.62 0.0
unit 2
cuboid  1 1 6.35 -6.35 6.35 -6.35 8.56 0.0
cylinder  0 1 13.97 8.56 0.0 origin 6.0934 0.0
cylinder  1 1 19.05 8.56 0.0 origin 6.0934 0.0
cuboid  0 1 25.1434 -12.9566 19.05 -19.05 8.56 0.0
unit 3
cuboid  1 1 6.35 -6.35 6.35 -6.35 2.616 0.0
cuboid  0 1 25.1434 -12.9566 19.05 -19.05 2.616 0.0
end geom
read array nux=1 nuy=1 nuz=3  fill 1 2 3 t end array end data
end
```

Sample Problem 14

```
=csas25
sample problem 14  u metal cylinder in an annulus
44group inf
uranium  1 den=18.69 1 293  92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 14  u metal cylinder in an annulus
read param                   end param
read geom
cylinder  1 1 8.89 10.109 0.0 orig 5.0799 0.0
cylinder  0 1 13.97 10.109 0.0
cylinder  1 1 19.05 10.109 0.0
end geom
end data
end
```

Sample Problem 15

```
=csas25
sample problem 15  small water reflected sphere on plexiglas collar
44group inf
uranium  1 den=18.794 1 293 92235 97.67 92238 0.98
                           92234 1.17  92236 0.20 end
plexiglas 2 end
h2o 3 end
end comp
sample problem 15  small water reflected sphere on plexiglas collar
read param                     flx=yes fdn=yes end param
read geom
unit 1
hemisphe-z 1 1 6.5537 chord -5.09066
cylinder  3 1 4.1275 -5.09066 -7.63065
cylinder  2 1 12.7   -5.09066 -7.63065
cuboid  3 1 4p12.7 -5.09066 -7.63065
unit 2
hemisphe+z 1 1 6.5537 chord 5.09066
cuboid  3 1 4p12.7 6.5537 -5.09066
core  0 1 -12.7 -12.7 -7.092175
cylinder  3 1 17.97 2p7.0922
replicate 3 2 3*3.0 5
end geom
read bias  id=500 2 6 end bias
read array nux=1 nuy=1 nuz=2  fill 1 2 end array
read plot ttl='x-z slice through the center of the sphere' lpi=6
xul=-20.0 zul=10.0 yul=0.0  xlr=20.0 ylr=0.0 zlr=-10.0
uax=1.0 wdn=-1.0 nax=130    nch=' *0-'
end plot
end data
end
```

Sample Problem 16

```
=csas25
sample problem 16 uo2f2 infinite slab k-infinity
44group inf
solnuo2f2       1 578.7 0 1.0 293 92235 93.2 92238 6.8 end
solnuo2f2       3 578.7 0 1.0 293 92235 93.2 92238 6.8 end
pyrex 2 end
boron           3 den=.0266 end
end comp
sample problem 16 uo2f2 infinite slab k-infinity
read parameters              amx=yes xap=no              end parameters
read geometry
cuboid 1 1 2.479 -2.479 100 -100 100 -100
cuboid 2 1 3.749 -3.749 100 -100 100 -100
cuboid 3 1 17.479 -17.479 100 -100 100 -100
end geom
read bounds  all=mirror  end bounds    read array  end array
end data
end
```

Sample Problem 17

```
=csas25
sample problem 17 93% uo2f2 solution sphere  adjoint calculation
44group inf
solnuo2f2  1 133 0 1.0 293 92235 93.0 92238 7 end
end comp
sample problem 17 93% uo2f2 solution sphere  adjoint calculation
read parameters              adj=yes amx=yes xap=no end parameters
read geometry
sphere 1 1 16.0
end geom
end data
end
```

Sample Problem 18

```
=csas25
sample problem 18   1f27 demonstration of options problem
44group inf
solnuo2(no3)2  1 415 9.783-3 spg=1.555 1.0 293 92235 92.6 92238 5.9
                                               92234 1.0  92236 0.5 end
plexiglas 2 end
para(h2o) 3 end
h2o 4 1-9 end
end comp
sample problem 18   1f27 demonstration of options problem
read para   gen=103 npg=500 fdn=yes nub=yes
  mku=yes fmu=yes mkh=yes fmh=yes mka=yes fma=yes rnd=f12c09ed2195
  pwt=yes far=yes flx=yes amx=yes pax=yes pgm=yes
 end para
read bounds   -zb= h2o   end bounds
read geom unit 1 cylinder 1 1 9.52 8.7804 -8.7804
 cylinder 0 1 9.52 8.9896 -8.7804
 cylinder 2 1 10.16 9.6296 -9.4204
 cuboid 4 1 18.45 -18.45 18.45 -18.45 17.8946 -17.6854
 unit 2 array 1 3*0.0
 unit 3 array 2 3*0.0
 unit 4 array 3 3*0.0
 unit 5 array 4 3*0.0
 global
 unit 6 cuboid 4 1 55.3501 -55.3501 55.3501 -55.3501 53.3701 -53.3701
```

```
  hole 2 -55.35   -18.45   -17.79
  hole 3 -55.35   -18.45   -53.3701
  hole 4  18.4501 -18.45   -53.3701
  hole 5 -55.3501 -55.3501 -53.3701
 replicate 3 2 6*3 5  replicate 3 7 6*0.24 1  end geom
 read bias  id=400 2 7  end bias
read array
 ara=1 nux=2 nuy=2 nuz=2 fill f1 end fill
 ara=2 nux=2 nuy=2 nuz=1 fill f1 end fill
 ara=3 nux=1 nuy=2 nuz=3 fill f1 end fill
 ara=4 nux=3 nuy=1 nuz=3 fill f1 end fill
 end array
 read start nst=6 tfx=0.0 tfy=0.0 tfz=0.0
 lnu=350 ps6=yes
 end start
read plot          plt=yes lpi=6
ttl=?  1f27 xy plot at z=0.0 ?  Xul=-71.0 yul=71.0 zul=0.0
 xlr=71.0 ylr=-71.0 zlr=0.0 uax=1 vdn=-1 nax=130 nch=?.*-3 ?
 run=yes end      ttl=?unit map 1f27 xy plot at z=0.0?
 pic=unit nch=? 123456? end plot    end data
end
```

Sample Problem 19

```
=csas25
sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
solnuo2(no3)2  2 415 9.783-3 spg=1.555 1.0 293 92235 92.6 92238 5.9
                                               92234 1.0  92236 0.5 end
plexiglas 3 end
end comp
sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)
read param                    flx=yes fdn=yes    nub=yes smu=yes mkp=yes
mku=yes fmp=yes fmu=yes end param
read geom
unit 1
com='uranyl nitrate solution in a plexiglas container'
cylinder  2 1 9.525 2p8.89
cylinder  3 1 10.16 2p9.525
cuboid  0 1 4p10.875 2p10.24
unit 2
com='uranium metal cylinder'
cylinder  1 1 5.748 2p5.3825
cuboid  0 1 4p6.59 2p6.225
unit 3
com='1x2x2 array of solution units'
array 1 3*0.0
unit 4
com='1x2x2 array of metal units padded to match solution array'
array 2 3*0.0
replicate 0 1 2*0.0 2*8.57 2*8.03 1
end geom
read array ara=1 nux=1 nuy=2 nuz=2 fill f1 end fill
ara=2 nux=1 nuy=2 nuz=2 fill f2 end fill gbl=3 ara=3 nux=2 nuy=1 nuz=1
com='composite array of solution and metal units'
fill 4 3 end fill
end array
end data
end
```

Sample Problem 20

```
=csas25
sample problem 20 triangular pitched array
44group inf
solnuo2f2  1 576.87 0 1.0 293 92235 93.2 92238 6.8 end
al 2 end
end comp
sample problem 20 triangular pitched array
read param                end param
read geom
unit 1
cylinder   1 1 10.16 18.288 0
cylinder   2 1 10.312 18.288 -.152
unit 2
cuboid     0 1 4p50 50 -.152
hole       1 3r0
hole       1 21.006 2r0
hole       1 -21.006 2r0
hole       1 10.503 18.192 0
hole       1 -10.503 18.192 0
hole       1 10.503 -18.192 0
hole       1 -10.503 -18.192 0
end geom
read array nux=1 nuy=1 nuz=1 fill 2 end fill end array
read plot ttl='hex array' pic=mix lpi=6
xul=0 yul=100 zul=10
xlr=100 ylr=0 zlr=10 uax=1 vdn=-1 nax=120
nch=' 12' end plot
end data
end
```

Sample Problem 21

```
=csas25
sample problem 21  partially filled sphere
44group inf
solnuo2f2  1 494 0 spg= 1.56 1 293 92235 4.89 92238 95.09 92234 0.02 end
al 2 end
end comp
sample problem 21  partially filled sphere
read param                end param
read geom
hemisphe-z  1 1 34.6   chord 30.
sphere      0 1 34.6
sphere      2 1 34.759
end geom
end data
end
```

Sample Problem 22

```
=csas25
sample problem 22   case 2c8 bare with 3 nested holes, each is equal volume
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 22   case 2c8 bare with 3 nested holes, each is equal volume
read parameters   flx=yes fdn=yes far=yes           end parameters
read geometry
unit 1
```

```
cylinder 1 1 3.621 2p3.3907
unit 2
cylinder 1 1 4.5622 2p4.2721
hole 1 3*0.0
unit 3
cylinder 1 1 5.2224 2p4.8903
hole 2 3*0.0
unit 4
cylinder 1 1 5.748 5.3825 -5.3825
hole 3 3*0.0
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
end geometry
read array  nux=2 nuy=2 nuz=2 fill f4 end fill end array
end data
end
```

Sample Problem 23

```
=csas25
sample problem 23  case 2c8 bare as mixed zhemicylinders
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 23  case 2c8 bare as mixed zhemicylinders
read parameters   fdn=yes           end parameters
read geometry
unit 1
com='-x half of unit 3'
zhemicyl-x 1 1 5.748 5.3825 -5.3825
cuboid  0 1 0.0  -6.87 6.87 -6.87 6.505 -6.505
unit 2
com='+x half of unit 3'
zhemicyl+x 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87  0.0  6.87 -6.87 6.505 -6.505
unit 3
com='cylinder composed of equal halves (zhemicylinders with x radii)'
array 1 3*0.0
unit 4
com='-x portion (more than half) of unit 6'
zhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 3.0  -6.87 6.87 -6.87 6.505 -6.505
unit 5
com='+x portion (less than half) of unit 6'
zhemicyl+x 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.87  3.0  6.87 -6.87 6.505 -6.505
unit 6
com='cylinder composed of unequal halves (zhemicylinders with x radii)'
array 2 3*0.0
unit 7
com='cylinder of a single zhemicylinder in the -x direction'
zhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 8
com='cylinder of a single zhemicylinder in the +x direction'
zhemicyl+x 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 9
com='-y half of unit 11'
zhemicyl-y 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 0.0  -6.87 6.505 -6.505
unit 10
com='+y half of unit 11'
zhemicyl+y 1 1 5.748 5.3825 -5.3825
```

```
cuboid  0 1 6.87 -6.87 6.87  0.0  6.505 -6.505
unit 11
com='cylinder composed of equal halves (zhemicylinders with z radii)'
array 3 3*0.0
unit 12
com='-y portion (more than half) of unit 14'
zhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 6.87 -6.87 3.0  -6.87 6.505 -6.505
unit 13
com='+y portion (less than half) of unit 14'
zhemicyl+y 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.87 -6.87 6.87  3.0  6.505 -6.505
unit 14
com='cylinder composed of unequal halves (zhemicylinders with z radii)'
array 4 3*0.0
unit 15
com='cylinder of a single zhemicylinder in the -y direction'
zhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
unit 16
com='cylinder of a single zhemicylinder in the +y'
zhemicyl+y 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
end geometry
read array
com='array 1 defines unit 3 (zhemicylinders with x radii)'
ara=1 nux=2 nuy=1 nuz=1 fill 1 2 end fill
com='array 2 defines unit 6 (zhemicylinders with x radii)'
ara=2 nux=2 nuy=1 nuz=1 fill 4 5 end fill
com='array 3 defines unit 11 (zhemicylinders with y radii)'
ara=3 nux=1 nuy=2 nuz=1 fill 9 10 end fill
com='array 4 defines unit 14 (zhemicylinders with y radii)'
ara=4 nux=1 nuy=2 nuz=1 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
gbl=5 ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

Sample Problem 24

```
=csas25
sample problem 24  case 2c8 bare as mixed xhemicylinders
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 24  case 2c8 bare as mixed xhemicylinders
read parameters   fdn=yes           end parameters
read geometry
unit 1
com='-y half of unit 3'
xhemicyl-y 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.505 -6.505 0.0 -6.87 6.87 -6.87
unit 2
com='+y half of unit 3'
xhemicyl+y 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.505 -6.505 6.87 0.0  6.87 -6.87
unit 3
com='cylinder composed of equal halves (xhemicylinders with y radii)'
array 1 3*0.0
unit 4
com='-y portion (more than half) of unit 6'
xhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 3.0
```

```
cuboid  0 1 6.505 -6.505 3.0 -6.87 6.87 -6.87
unit 5
com='+y portion (less than half) of unit 6'
xhemicyl+y 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.505 -6.505 6.87 3.0  6.87 -6.87
unit 6
com='cylinder composed of unequal halves (xhemicylinders with y radii)'
array 2 3*0.0
unit 7
com='cylinder of a single xhemicylinder in the -y direction'
xhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 -6.87
unit 8
com='cylinder of a single xhemicylinder in the +y direction'
xhemicyl+y 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 -6.87
unit 9
com='-z half of unit 11'
xhemicyl-z 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.505 -6.505 6.87 -6.87 0.0 -6.87
unit 10
com='+z half of unit 11'
xhemicyl+z 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 0.0
unit 11
com='cylinder composed of equal halves (xhemicylinders with z radii)'
array 3 3*0.0
unit 12
com='-z portion (more than half) of unit 14'
xhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 6.505 -6.505 6.87 -6.87 3.0 -6.87
unit 13
com='+z portion (less than half) of unit 14'
xhemicyl+z 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 3.0
unit 14
com='cylinder composed of unequal halves (xhemicylinders with z radii)'
array 4 3*0.0
unit 15
com='cylinder of a single xhemicylinder in the -z direction'
xhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 -6.87
unit 16
com='cylinder of a single xhemicylinder in the +z direction'
xhemicyl+z 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 -6.87
end geometry
read array
com='array 1 defines unit 3 (xhemicylinders with y radii)'
ara=1 nux=1 nuy=2 nuz=1 fill 1 2 end fill
com='array 2 defines unit 6 (xhemicylinders with y radii)'
ara=2 nux=1 nuy=2 nuz=1 fill 4 5 end fill
com='array 3 defines unit 11 (xhemicylinders with z radii)'
ara=3 nux=1 nuy=1 nuz=2 fill 9 10 end fill
com='array 4 defines unit 14 (xhemicylinders with z radii)'
ara=4 nux=1 nuy=1 nuz=2 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
gbl=5 ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

Sample Problem 25

```
=csas25
sample problem 25  case 2c8 bare as mixed yhemicylinders
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 25  case 2c8 bare as mixed yhemicylinders
read parameters              fdn=yes  end parameters
read geometry
unit 1
com='-x half of unit 3'
yhemicyl-x 1 1 5.748 5.3825 -5.3825
cuboid  0 1 0.0  -6.87 6.505 -6.505 6.87 -6.87
unit 2
com='+x half of unit 3'
yhemicyl+x 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87  0.0  6.505 -6.505 6.87 -6.87
unit 3
com='cylinder composed of equal halves (yhemicylinders with x radii)'
array 1 3*0.0
unit 4
com='-x portion (more than half) of unit 6'
yhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 3.0  -6.87 6.505 -6.505 6.87 -6.87
unit 5
com='+x portion (less than half) of unit 6'
yhemicyl+x 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.87  3.0  6.505 -6.505 6.87 -6.87
unit 6
com='cylinder composed of unequal halves (yhemicylinders with x radii)'
array 2 3*0.0
unit 7
com='cylinder of a single yhemicylinder in the -x direction'
yhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 -6.87
unit 8
com='cylinder of a single yhemicylinder in the +x direction'
yhemicyl+x 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 -6.87
unit 9
com='-z half of unit 11'
yhemicyl-z 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.505 -6.505 0.0 -6.87
unit 10
com='+z half of unit 11'
yhemicyl+z 1 1 5.748 5.3825 -5.3825
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 0.0
unit 11
com='cylinder composed of equal halves (yhemicylinders with z radii)'
array 3 3*0.0
unit 12
com='-z portion (more than half) of unit 14'
yhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 3.0
cuboid  0 1 6.87 -6.87 6.505 -6.505 3.0 -6.87
unit 13
com='+z portion (less than half) of unit 14'
yhemicyl+z 1 1 5.748 5.3825 -5.3825 chord -3.0
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 3.0
unit 14
com='cylinder composed of unequal halves (yhemicylinders with z radii)'
array 4 3*0.0
unit 15
com='cylinder of a single  yhemicylinder in the -z direction'
```

```
yhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 -6.87
unit 16
com='cylinder of a single yhemicylinder in the +z direction'
yhemicyl+z 1 1 5.748 5.3825 -5.3825 chord 5.748
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 -6.87
end geometry
read array
com='array 1 defines unit 3 (yhemicylinders with x radii)'
ara=1 nux=2 nuy=1 nuz=1 fill 1 2 end fill
com='array 2 defines unit 6 (yhemicylinders with x radii)'
ara=2 nux=2 nuy=1 nuz=1 fill 4 5 end fill
com='array 3 defines unit 11 (yhemicylinders with z radii)'
ara=3 nux=1 nuy=1 nuz=2 fill 9 10 end fill
com='array 4 defines unit 14 (zhemicylinders with z radii)'
ara=4 nux=1 nuy=1 nuz=2 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
gbl=5 ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

Sample Problem 26

```
=csas25
sample problem 26  case 2c8 bare as mixed zhemicylinders with origins
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 26  case 2c8 bare as mixed zhemicylinders with origins
read parameters  fdn=yes lib=4  run=yes end parameters
read geometry
unit 1
com='-x half of first cylinder'
zhemicyl-x 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
cuboid  0 1 6.87 0.0 6.87 -6.87 6.505 -6.505
unit 2
com='+x half of first cylinder'
zhemicyl+x 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
cuboid  0 1 13.74 6.87 6.87 -6.87 6.505 -6.505
unit 3
com='1st cylinder composed of equal portions (z hemicylinders with x radii)'
array 1 3*0.0
unit 4
com='-x portion (more than half) of second cylinder'
zhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 6.87 0.0
cuboid  0 1 9.87 0.0 6.87 -6.87 6.505 -6.505
unit 5
com='+x portion (less than half) of second cylinder'
zhemicyl+x 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 6.87 0.0
cuboid  0 1 13.74 9.87 6.87 -6.87 6.505 -6.505
unit 6
com='2nd cylinder composed of unequal portions (z hemicylinders with x radii)'
array 2 3*0.0
unit 7
com='3rd cylinder: described as a zhemicylinder in the -x direction'
zhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 13.74 0.0 6.87 -6.87 6.505 -6.505
unit 8
com='4th cylinder: described as a zhemicylinder in the +x direction'
zhemicyl+x 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 13.74 0.0 6.87 -6.87 6.505 -6.505
unit 9
com='-y half of fifth cylinder'
```

```
zhemicyl-y 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.87 0.0 6.505 -6.505
unit 10
com='+y half of fifth cylinder'
zhemicyl+y 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 13.74 6.87  6.505 -6.505
unit 11
com='5th cylinder composed of equal portions (zhemicylinders with y radii)'
array 3 3*0.0
unit 12
com='-y portion (more than half) of sixth cylinder'
zhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 9.87  0.0 6.505 -6.505
unit 13
com='+y portion (less than half) of sixth cylinder'
zhemicyl+y 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 13.74  9.87  6.505 -6.505
unit 14
com='6th cylinder composed of unequal portions (zhemicylinders with y radii)'
array 4 3*0.0
unit 15
com='7th cylinder: described as a zhemicylinder in the -y direction'
zhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 13.74 0.0 6.505 -6.505
unit 16
com='8th cylinder: described as a zhemicylinder in the +y direction'
zhemicyl+y 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 13.74 0.0 6.505 -6.505
global unit 17
com='complete 2c8 bare configuration'
array 5 3*0.0
end geometry
read array
com='array 1: 1st cylinder (unit 3) equal x portions of zhemicylinders'
ara=1 nux=2 nuy=1 nuz=1 fill 1 2 end fill
com='array 2: 2nd cylinder (unit 6) unequal x portions of zhemicylinders'
ara=2 nux=2 nuy=1 nuz=1 fill 4 5 end fill
com='array 3: 5th cylinder (unit 11) equal y portions of zhemicylinders'
ara=3 nux=1 nuy=2 nuz=1 fill 9 10 end fill
com='array 4: 6th cylinder (unit 14) unequal y portions of zhemicylinders'
ara=4 nux=1 nuy=2 nuz=1 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

Sample Problem 27

```
=csas25
sample problem 27  case 2c8 bare as mixed xhemicylinders with origins
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 27  case 2c8 bare as mixed xhemicylinders with origins
read parameters  fdn=yes lib=4 run=yes end parameters
read geometry
unit 1
com='-y half of first cylinder'
xhemicyl-y 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
cuboid  0 1 6.505 -6.505 6.87 0.0 6.87 -6.87
unit 2
com='+y half of first cylinder'
xhemicyl+y 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
```

```
cuboid  0 1 6.505 -6.505 13.74 6.87 6.87 -6.87
unit 3
com='1st cylinder composed of equal portions (xhemicylinders with y radii)'
array 1 3*0.0
unit 4
com='-y portion (more than half) of second cylinder'
xhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 6.87 0.0
cuboid  0 1 6.505 -6.505 9.87 0.0 6.87 -6.87
unit 5
com='+y portion (less than half) of second cylinder'
xhemicyl+y 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 6.87 0.0
cuboid  0 1 6.505 -6.505 13.74 9.87  6.87 -6.87
unit 6
com='2nd cylinder composed of unequal portions (xhemicylinders with y radii)'
array 2 3*0.0
unit 7
com='3rd cylinder: described as a xhemicylinder in the -y direction'
xhemicyl-y 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 6.505 -6.505 13.74 0.0 6.87 -6.87
unit 8
com='4th cylinder: described as a xhemicylinder in the +y direction'
xhemicyl+y 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 6.505 -6.505 13.74 0.0 6.87 -6.87
unit 9
com='-z half of fifth cylinder'
xhemicyl-z 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 6.87 0.0
unit 10
com='+z half of fifth cylinder'
xhemicyl+z 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 13.74 6.87
unit 11
com='5th cylinder composed of equal portions (xhemicylinders with z radii)'
array 3 3*0.0
unit 12
com='-z portion (more than half) of sixth cylinder'
xhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 9.87 0.0
unit 13
com='+z portion (less than half) of sixth cylinder'
xhemicyl+z 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 13.74 9.87
unit 14
com='6th cylinder composed of unequal portions (xhemicylinders with z radii)'
array 4 3*0.0
unit 15
com='7th cylinder: described as a xhemicylinder in the -z direction'
xhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 13.74 0.0
unit 16
com='8th cylinder: described as a xhemicylinder in the +z direction'
xhemicyl+z 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.505 -6.505 6.87 -6.87 13.74 0.0
global unit 17
com='complete 2c8 bare configuration'
array 5 3*0.0
end geometry
read array
com='array 1: 1st cylinder (unit 3) equal y portions of xhemicylinders'
ara=1 nux=1 nuy=2 nuz=1 fill 1 2 end fill
com='array 2: 2nd cylinder (unit 6) unequal y portions of xhemicylinders'
ara=2 nux=1 nuy=2 nuz=1 fill 4 5 end fill
com='array 3: 5th cylinder (unit 11) equal z portions of xhemicylinders'
ara=3 nux=1 nuy=1 nuz=2 fill 9 10 end fill
com='array 4: 6th cylinder (unit 14) unequal z portions of xhemicylinders'
ara=4 nux=1 nuy=1 nuz=2 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
```

```
gbl=5 ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

Sample Problem 28

```
=csas25
sample problem 28  case 2c8 bare as mixed yhemicylinders with origins
44group inf
uranium  1 den=18.76 1 293 92235 93.2 92238 5.6 92234 1.0 92236 0.2 end
end comp
sample problem 27  case 2c8 bare as mixed yhemicylinders with origins
read parameters  fdn=yes lib=4 run=yes end parameters
read geometry
unit 1
com='-x half of first cylinder'
yhemicyl-x 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
cuboid  0 1 6.87 0.0 6.505 -6.505 6.87 -6.87
unit 2
com='+x half of unit 3'
yhemicyl+x 1 1 5.748 5.3825 -5.3825 origin 6.87 0.0
cuboid  0 1 13.74 6.87  6.505 -6.505 6.87 -6.87
unit 3
com='1st cylinder composed of equal portions (yhemicylinders with x radii)'
array 1 3*0.0
unit 4
com='-x portion (more than half) of second cylinder'
yhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 6.87 0.0
cuboid  0 1 9.87 0.0 6.505 -6.505 6.87 -6.87
unit 5
com='+x portion (less than half) of second cylinder'
yhemicyl+x 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 6.87 0.0
cuboid  0 1 13.74 9.87 6.505 -6.505 6.87 -6.87
unit 6
com='2nd cylinder composed of unequal portions (yhemicylinders with x radii)'
array 2 3*0.0
unit 7
com='3rd cylinder: described as a single yhemicylinder in the -x direction'
yhemicyl-x 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 13.74 0.0 6.505 -6.505 6.87 -6.87
unit 8
com='4th cylinder: described as a single yhemicylinder in the +x direction'
yhemicyl+x 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 6.87 0.0
cuboid  0 1 13.74 0.0 6.505 -6.505 6.87 -6.87
unit 9
com='-z half of fifth cylinder'
yhemicyl-z 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 6.87 0.0
unit 10
com='+z half of sixth cylinder'
yhemicyl+z 1 1 5.748 5.3825 -5.3825 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 13.74 6.87
unit 11
com='5th cylinder composed of equal portions (yhemicylinders with z radii)'
array 3 3*0.0
unit 12
com='-z portion (more than half) of sixth cylinder'
yhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 3.0 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 9.87 0.0
unit 13
com='+z portion (less than half) of sixth cylinder'
yhemicyl+z 1 1 5.748 5.3825 -5.3825 chord -3.0 origin 0.0 6.87
```

```
cuboid  0 1 6.87 -6.87 6.505 -6.505 13.74 9.87
unit 14
com='6th cylinder composed of unequal portions (yhemicylinders with z radii)'
array 4 3*0.0
unit 15
com='7th cylinder: described as a yhemicylinder in the -z direction'
yhemicyl-z 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 13.74 0.0
unit 16
com='8th cylinder: described as a yhemicylinder in the +z direction'
yhemicyl+z 1 1 5.748 5.3825 -5.3825 chord 5.748 origin 0.0 6.87
cuboid  0 1 6.87 -6.87 6.505 -6.505 13.74 0.0
global unit 17
com='complete 2c8 bare configuration'
array 5 3*0.0
end geometry
read array
com='array 1: 1st cylinder (unit 3) equal x portions of yhemicylinders'
ara=1 nux=2 nuy=1 nuz=1 fill 1 2 end fill
com='array 2: 2nd cylinder (unit 6) unequal x portions of yhemicylinders'
ara=2 nux=2 nuy=1 nuz=1 fill 4 5 end fill
com='array 3: 5th cyllinder (unit 11) equal z portions of yhemicylinders'
ara=3 nux=1 nuy=1 nuz=2 fill 9 10 end fill
com='array 4: 6th cylinder (unit 14) unequal z portions of yhemicylinders'
ara=4 nux=1 nuy=1 nuz=2 fill 12 13 end fill
com='array 5 defines the total 2c8 problem'
gbl=5 ara=5 nux=2 nuy=2 nuz=2 fill 3 7 6 8 11 15 14 16 end fill
end array
end data
end
```

Sample Problem 29

```
=csas25
sample problem 29  bare critical sphere   3.4420" radius
44group inf
uranium  16 den=18.747 1 293  92235 93.21 92238 5.7697 92234 0.9844
                             92236 0.0359 end
end comp
sample problem 29  bare critical sphere   3.4420" radius
read parameters    fdn=yes lib=4
end parameters
read geometry
sphere   16 1 8.74268
end geometry
read plot ttl='x-y slice at z=0.0'
xul=-9 yul=9 zul=0.0 xlr=9 ylr=-9.0 zlr=0.0
uax=1 vdn=-1  nax=119 ndn=63 nch=' *'
end plot
end data
end
```

Sample Problem 30

```
=csas25
sample problem 30   bare critical sphere    z hemisphere model 3.4420" radius
44group inf
uranium  16 den=18.747 1 293  92235 93.21 92238 5.7697 92234 0.9844
                             92236 0.0359 end
```

```
end comp
sample problem 30   bare critical sphere    z hemisphere model 3.4420" radius
read parameters    fdn=yes lib=4
end parameters
read geometry
unit 1
hemisphe+z   16 1 8.74268  chord +3.0 origin 8.9 8.9 8.9
global unit 2
hemisphe-z   16 1 8.74268  chord -3.0 origin 8.9 8.9 8.9
cuboid       0 1 17.8 0.0 17.8 0.0 17.8 0.0
hole 1 3*0.0
end geometry
read plot ttl='y-z slice at x=8.9   mixture map'
xul=8.9 yul=-.5 zul=18.5 xlr=8.9 ylr=18.5 zlr=-0.5
vax=1 wdn=-1  nax=119 ndn=63 nch=' *' end plt1
ttl='y-z slice at x=8.9   unit map'
pic=box   nch=' 12' end plt2
end plot
end data
end
```

Sample Problem 31

```
=csas25
sample problem 31   bare critical sphere    x hemisphere model 3.4420" radius
44group inf
uranium  16 den=18.747 1 293  92235 93.21 92238 5.7697 92234 0.9844
                             92236 0.0359 end
end comp
sample problem 31   bare critical sphere    x hemisphere model 3.4420" radius
read parameters    fdn=yes lib=4
end parameters
read geometry
unit 1
hemisphe-x   16 1 8.74268  chord +3.0
global unit 2
hemisphe+x   16 1 8.74268  chord -3.0 origin 8.9 8.9 8.9
cuboid       0 1 17.8 0.0 17.8 0.0 17.8 0.0
hole 1 3*8.9
end geometry
read plot ttl='x-y slice at z=8.9    mixture map'
xul=-0.5 yul=18.5 zul=8.9  xlr=18.5 ylr=-0.5 zlr=8.9
uax=1 vdn=-1  nax=119 ndn=63 nch=' *'  end plt1
ttl='y-z slice at x=8.9   unit map'
pic=box   nch=' 12' end plt2
end plot
end data
end
```

Sample Problem 32

```
=csas25
sample problem 32   bare critical sphere    y hemisphere model 3.4420" radius
44group inf
uranium  16 den=18.747 1 293  92235 93.21 92238 5.7697 92234 0.9844
                              92236 0.0359 end
end comp
sample problem 32   bare critical sphere    y hemisphere model 3.4420" radius
read parameters    fdn=yes lib=4
end parameters
read geometry
unit 1
hemisphe-y   16 1 8.74268  chord +3.0 origin 8.9 9.9 10.9
global unit 2
hemisphe+y   16 1 8.74268  chord -3.0 origin 8.9 8.9 8.9
cuboid       0 1 17.8 0.0 17.8 0.0 17.8 0.0
hole 1 0.0 -1.0 -2.0
end geometry
read plot ttl=’y-z slice at x=8.9     mixture map’
xul=8.9 yul=-.5 zul=18.5 xlr=8.9 ylr=18.5 zlr=-0.5
vax=1 wdn=-1  nax=119 ndn=63 nch=’ *’  end plt1
ttl=’y-z slice at x=8.9   unit map’
pic=box   nch=’ 12’ end plt2
end data
end
```

Sample Problem 33

```
=csas25
sample problem 33   critical triangular pitched array of annular rods
44group latticecell
uranium  17 den=19.0 1 293  92235 1.95 92238 98.042 92234 0.002 92236 0.006  end
h2o      18 end
h2o      19 end
uranium  20 den=19.0 1 293  92235 1.95 92238 98.042 92234 0.002 92236 0.006  end
end comp
atriangpitch  20.828 18.288 17 19 18 6.604 end
more data  res=20 slab 5.842  dan(20)=1.0  end more
end
sample problem 33   critical triangular pitched array of annular rods
read parameters  fdn=yes nub=yes gen=153 npg=500 lib=4
end parameters
read geom
unit 1
zhemicyl-x 18 1 3.302 47.7 0.6
zhemicyl-x 17 1 9.144 47.7 0.6
unit 2
zhemicyl-y 18 1 3.302 47.7 0.6
zhemicyl-y 17 1 9.144 47.7 0.6
unit 3
zhemicyl+x 18 1 3.302 47.7 0.6
zhemicyl+x 17 1 9.144 47.7 0.6
unit 4
zhemicyl+y 18 1 3.302 47.7 0.6  origin 0.0 -18.03758
zhemicyl+y 17 1 9.144 47.7 0.6  origin 0.0 -18.03758
cuboid     19 1 2p10.414 2p18.03758 47.7 0.6
hole       1      10.414    0.0      0.0
hole       2        0.0     18.03758  0.0
hole       3     -10.414    0.0      0.0
unit 5
```

```
cuboid     19 1 2p10.414 10.414 0.0 47.7 0.6
unit 6
zhemicyl-y 18 1 3.302 47.7 0.6
zhemicyl-y 17 1 9.144 47.7 0.6
cuboid     19 1 2p10.414 0.0 -10.414 47.7 0.6
unit 7
zhemicyl-y 18 1 3.302 47.7 0.6 origin 0.0 18.03758
zhemicyl-y 17 1 9.144 47.7 0.6 origin 0.0 18.03758
cuboid     19 1 2p10.414 2p18.03758 47.7 0.6
hole     3      -10.414 0.0 0.0
unit 8
zhemicyl+y 18 1 3.302 47.7 0.6 origin 0.0 -18.03758
zhemicyl+y 17 1 9.144 47.7 0.6 origin 0.0 -18.03758
cuboid     19 1 2p10.414 2p18.03758 47.7 0.6
hole     3      -10.414 0.0 0.0
unit 9
zhemicyl+y 18 1 3.302 47.7 0.6
zhemicyl+y 17 1 9.144 47.7 0.6
cuboid     19 1 2p10.414 10.414 0.0 47.7 0.6
unit 10
zhemicyl+y 18 1 3.302 47.7 0.6  origin 0.0 -18.03758
zhemicyl+y 17 1 9.144 47.7 0.6  origin 0.0 -18.03758
cuboid     19 1 2p10.414 2p18.03758 47.7 0.6
hole     1       10.414 0.0 0.0
unit 11
zhemicyl-y 18 1 3.302 47.7 0.6  origin 0.0 18.03758
zhemicyl-y 17 1 9.144 47.7 0.6  origin 0.0 18.03758
cuboid     19 1 2p10.414 2p18.03758 47.7 0.6
hole     1       10.414 0.0 0.0
unit 21
zhemicyl-x  0 1 3.302 102.2 47.7
zhemicyl-x 20 1 9.144 102.2 47.7
unit 22
zhemicyl-y  0 1 3.302 102.2 47.7
zhemicyl-y 20 1 9.144 102.2 47.7
unit 23
zhemicyl+x  0 1 3.302 102.2 47.7
zhemicyl+x 20 1 9.144 102.2 47.7
unit 24
zhemicyl+y  0 1 3.302 102.2 47.7  origin 0.0 -18.03758
zhemicyl+y 20 1 9.144 102.2 47.7  origin 0.0 -18.03758
cuboid      0 1 2p10.414 2p18.03758 102.2 47.7
hole      21      10.414    0.0      0.0
hole      22        0.0     18.03758  0.0
hole      23     -10.414    0.0      0.0
unit 25
cuboid      0 1 2p10.414 10.414 0.0 102.2 47.7
unit 26
zhemicyl-y  0 1 3.302 102.2 47.7
zhemicyl-y 20 1 9.144 102.2 47.7
cuboid      0 1 2p10.414 0.0 -10.414 102.2 47.7
unit 27
zhemicyl-y  0 1 3.302 102.2 47.7  origin 0.0 18.03758
zhemicyl-y 20 1 9.144 102.2 47.7  origin 0.0 18.03758
cuboid      0 1 2p10.414 2p18.03758 102.2 47.7
hole     23      -10.414 0.0 0.0
unit 28
zhemicyl+y  0 1 3.302 102.2 47.7  origin 0.0 -18.03758
zhemicyl+y 20 1 9.144 102.2 47.7  origin 0.0 -18.03758
cuboid      0 1 2p10.414 2p18.03758 102.2 47.7
hole     23      -10.414 0.0 0.0
unit 29
zhemicyl+y  0 1 3.302 102.2 47.7
zhemicyl+y 20 1 9.144 102.2 47.7
```

```
cuboid      0 1 2p10.414 10.414 0.0 102.2 47.7
unit 30
zhemicyl+y  0 1 3.302 102.2 47.7  origin 0.0 -18.03758
zhemicyl+y 20 1 9.144 102.2 47.7  origin 0.0 -18.03758
cuboid      0 1 2p10.414 2p18.03758 102.2 47.7
hole     21       10.414 0.0 0.0
unit 31
zhemicyl-y  0 1 3.302 102.2 47.7  origin 0.0 18.03758
zhemicyl-y 20 1 9.144 102.2 47.7  origin 0.0 18.03758
cuboid      0 1 2p10.414 2p18.03758 102.2 47.7
hole     21       10.414 0.0 0.0
unit 32
com='flooded portion of array with 15.24 cm of water in x and y'
array 1 2*0.0 0.6
replicate 19 1 4r15.24 0.0 0.6 1
replicate 19 2 5r0.0 3.0 7
unit 33
com='unflooded upper portion of array'
array 2 3*0.0
replicate 0 1 4r15.24 2*0.0 1
global
unit 34
array 3 -67.31 -61.72916 -21.0
end geom
read bias id=500 2 8 end bias
read array
ara=1 nux=5 nuy=4 nuz=1  fill
5 3r6 5      11 3r4 7      10 3r4 8       5 3r9 5     end fill
ara=2 nux=5 nuy=4 nuz=1  fill
25 3r26 25   31 3r24 27    30 3r24 28     25 3r29 25  end fill
ara=3 nux=1 nuy=1 nuz=2 fill 32 33 end fill
end array
read start nst=1 xsm=-52 xsp=52 ysm=-47 ysp=47 zsm=0.6 zsp=47.7
end start
read plot
ttl='x-y plot of pins at z=45.0'
xul=-52.0 yul=47.0 zul=45.0  xlr=52.0 ylr=-47.0 zlr=45.0
uax=1.0  vdn=-1.0 nax=130 ndn=62 nch=' *.-' end plt1
ttl='x-z plot of pins at y=0.0'
xul=-52.0 yul=0.0 zul=102.7 xlr=52 ylr=0.0 zlr=-3.0
uax=1.0 wdn=-1.0 nax=130 ndn=89  end plt2
ttl='x-z plot at y=0.0'
xul=-68.0 yul=0.0 zul=102.7 xlr=70.0 ylr=0.0 zlr=-25.0
uax=1.0 wdn=-1.0 nax=130 ndn=64  end plt3
end plot
end data
end
```

F11.D.4 SAMPLE PROBLEM 33 MODELING TECHNIQUES

This critical experiment[10,11] consists of an array of 19 billets of cylindrical uranium metal annuli (1.95% enriched) arranged in a triangular-pitched array creating an array in the shape of a regular hexagon with 3 billets on a side (Figs. F11.D.12 and F11.D.13). The array was constructed on a metal grid supported by wooden beams in a very large stainless steel tank. This support structure and the tank itself were ignored in the problem model. Because the critical experiment was only partially flooded, the cross-section processing for the flooded fuel should be different from that for the nonflooded fuel. In addition, it is good practice to assign different mixture numbers to the water inside the annulus and the water that is between the billets. Therefore, two fuel mixtures and two water mixtures are defined in this model. If the problem is to be run using cell-weighted cross sections (CSAS2X), an additional water mixture should be defined for the external water reflector.

Section F11.D.1 contains the stand-alone KENOV.a input data for the KENOV.a sample problems. For Sample Problem 33, mixture 17 represents the uranium metal in the flooded region, mixture 18 represents the water inside the annuli, mixture 19 represents the water between the billets, and mixture 20 represents the unmoderated uranium metal that extends above the water. The cross-section processing for the KENOV.a sample problems that are distributed with SCALE is done using BONAMI and NITAWL.

Section F11.D.2.2 contains the BONAMI and NITAWL data used to create the problem-dependent, working format, cross-section library for use with the KENOV.a sample problems. The problem-dependent, working format, cross-section library is written on "unit 4" which KENOV.a utilizes by specifying the parameter **LIB=4** and entering a mixing table data block to create the mixtures in KENOV.a. The input data for BONAMI and NITAWL was obtained by running CSASN with "PARM=CHECK." For Sample Problem 33, mixture 17 represents the uranium metal in the flooded region, mixture 18 represents the water inside the annuli, mixture 19 represents the water between the billets, and mixture 20 represents the unmoderated uranium metal that extends above the water.

Section F11.D.2.1 contains the CSASN data that can be used to create a problem-dependent, working format, cross-section library for use with the KENOV.a sample problems. The problem-dependent working format cross-section library is written on "unit 4" which KENOV.a utilizes by specifying the parameter **LIB=4** and entering a mixing table data block to create the mixtures in KENO V.a. For Sample Problem 33, mixture 17 represents the uranium metal in the flooded region, mixture 18 represents the water inside the annuli, mixture 19 represents the water between the billets, and mixture 20 represents the unmoderated uranium metal that extends above the water. The geometry used for cross-section processing is an annular latticecell with a triangular pitch. The latticecell option is specified in order to account for the lattice effects in the array. The latticecell data that are entered are:

ATRIANGPITCH 20.828 18.288 17 19 18 6.604 END

Because the fuel that extends above the water is not moderated and because the current version of CSAS allows only one type of cell for cross-section processing, the cross-section processing for this fuel mixture is done using "**MORE DATA**," utilizing the parameters **RES=20 SLAB 5.842** and **DAN(20)=1.0**. The resonance processing for mixture 20 is done using a slab of mixture 20 that is 5.842 cm thick. A slab was chosen as the best approximation for the annular cylinder because the only other options are cylindrical and spherical. The slab thickness is identical to the thickness of the cylindrical annulus. The Dancoff factor is 1.0 because there is no moderator present between fuel. Because the Dancoff factor is 1.0, the cross-section processing is identical to the default processing which is done using the infinite homogeneous media treatment. Therefore, the "**MORE DATA**" could be omitted for this problem. It has been included in order to emphasize the importance of providing adequate data for cross-section processing. If a different moderating material were present in the upper portion of the array, **"MORE DATA"** would be required in order to provide the data required to process the cross sections properly.

Section F11.D.3 contains the KENO V.a sample problems modeled as individual CSAS25 problems. The cross-section processing is done in CSAS and the processed cross sections and mixing table data are passed directly to KENO V.a. The parameter **lib=** and the mixing table data must be omitted from the KENOV.a data. In this section, sample problem 33 is modeled using mixture 1 to represent the uranium metal in the flooded region, mixture 2 to represent the water inside the annuli, mixture 3 to represent the water between the billets, and mixture 4 to represent the unmoderated uranium metal that extends above the water. The geometry for cross-section processing is modeled as an annular latticecell with a triangular pitch. The latticecell option is specified in order to account for the lattice effects in the array. The latticecell data entered are:

ATRIANGPITCH 20.828 18.288 1 3 2 6.604 END

Because the fuel that extends above the water is not moderated and because the current version of CSAS allows only one type of cell for cross-section processing, the cross-section processing for this fuel mixture is done using **"MORE DATA,"** utilizing the parameters **RES=4 SLAB 5.842** and **DAN(4)=1.0**. The resonance processing for mixture 4 is done using a slab of mixture 4 that is 5.842 cm thick. A slab was chosen as the best approximation for the annular cylinder because the only other options are cylindrical and spherical. The slab thickness is identical to the thickness of the cylindrical annulus. The Dancoff factor is 1.0 because there is no moderator present between fuel lumps.

Sample Problem 33 demonstrates the need for experimental data to establish the code/cross-section bias for specific types of systems. It is composed of large regions of low-enriched uranium metal in which slightly less than half of the length of the metal billets are water moderated. The cross-section preparation is complicated by the fact that each annulus has a water moderator in the interior region, the thickness of the annulus is large, and the billets are arranged relatively close together in a triangular-pitched array. There is no moderating material mixed with the uranium metal so the flux within the fuel has a relatively hard spectrum and the unresolved resonance range contributes significantly to k-effective. This is a problem in which heterogeneous effects are also important. A summary of results obtained for this problem using several SCALE cross-section libraries is given in Table F11.D.4.1.

Table F11.D.1 Sample Problem 33 results

Cross-section library	Average value of k-effective
16 group	0.9672 + or - 0.0023
27 group	0.9880 + or - 0.0023
44 group	1.0082 + or - 0.0023
123 group	0.9869 + or - 0.0024
238 group	1.0068 + or - 0.0024

F11.D.5 REFERENCES

1. J. T. Thomas, "Critical Three-Dimensional Arrays of U(93.2)-Metal Cylinders," *Nucl. Sci. Eng.* **52**, 350 (November 1973).

2. G. E. Whitesides and J. T. Thomas, "The Use of Differential Current Albedos in Monte Carlo Criticality Calculations," *Trans. Am. Nucl. Soc.* **12**, 889 (November 1969).

3. J. T. Thomas, *Critical Three-Dimensional Arrays of Neutron-Interacting Units, Part II*, ORNL/TM-868, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., July 1964.

4. D. C. Irving and J. T. Mihalczo, "Monte Carlo Calculations for Enriched Uranium Metal Assemblies," *Trans. Am. Nucl. Soc.* **7**, 284 (November 1964).

5. Cleo C. Byers et al., "Critical Measurements of a Water-Reflected Enriched Uranium Sphere," *Trans. Am. Nucl. Soc.* **27**, 412 (November 1977).

6. J. T. Thomas, *Critical Three-Dimensional Arrays of Neutron-Interacting Units*, ORNL/TM-719, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., 1963.

7. J. K. Fox, L. W. Gilley, D. Callihan, *Critical Mass Studies, Part IX Aqueous U-235 Solutions*, ORNL-2367, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., February 1958.

8. D. F. Cronin, *Critical Mass Studies, Part X Uranium of Intermediate Enrichment*, ORNL-2968, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., September 1960.

9. J. T. Mihalczo, J. J. Lynn, J. R. Taylor, and G. E. Hansen, "Measurements with an Unreflected Uranium (93.2%) Metal Sphere," *1993 Topical Meeting on Physics and Methods in Criticality Safety*, pp. 26–33, September 19–23, 1993.

10. E. B. Johnson, "Critical Parameters of U(1.95) Metal Cylindrical Annuli," *Neutron Physics Division Annual Progress Report, September 1966*, ORNL-3973, Union Carbide Corp., Nucl. Div., Oak Ridge Natl. Lab., 1966.

11. E. B. Johnson, "Critical Parameters of U(1.95) Metal Cylindrical Annuli," *Trans. Am. Nucl. Soc.* **9**, 185–186 (1966).

F11.E KENO V.a SAMPLE PROBLEM OUTPUT

F11.E.1 SELECTED OUTPUT FROM SAMPLE PROBLEMS

The computer output on the following pages is an example of the output for sample problems 4 and 19 described in Sect. F11.D. This output may not be identical to that on the SCALE-4 distribution tape since these problems were run prior to final testing.

```
                kk          kk  eeeeeeeeeeeee   nn          nn   ooooooooooo                    vv          vv
                kk         kk   eeeeeeeeeeeee   nnn         nn  ooooooooooooo                   vv          vv
                kk        kk    ee              nnnn        nn  oo         oo                   vv          vv
                kk       kk     ee              nn nn       nn  oo         oo                   vv          vv
                kk      kk      ee              nn   nn     nn  oo         oo                   vv          vv
                kkkkkkkk        eeeeeeeee       nn    nn    nn  oo         oo   -------------   vv          vv
                kkkkkkkk        eeeeeeeee       nn     nn   nn  oo         oo   -------------    vv        vv
                kk      kk      ee              nn      nn  nn  oo         oo                      vv     vv
                kk       kk     ee              nn       nn nn  oo         oo                       vv   vv
                kk        kk    ee              nn        nnnn  oo         oo                        vv vv
                kk         kk   eeeeeeeeeeeee   nn         nnn  ooooooooooooo                         vvv
                kk          kk  eeeeeeeeeeeee   nn          nn   ooooooooooo                           v

xx          xx           44      sssssssssss
 xx        xx           444     sssssssssssss
   xx     xx           4444     ss         ss
    xx   xx           44 44     ss
     xx xx           44  44     ss
      xxx           44   44     ssssssssssss
      xxx          44    44      ssssssssssss
     xx xx       444444444444              ss
    xx   xx     4444444444444              ss
   xx     xx             44     ss         ss
 xx        xx            44     sssssssssssss
xx          xx           44      sssssssssss

   0000000       99999999999                //     0000000      7777777777777               //   99999999999    5555555555555
  000000000     9999999999999              //     000000000     777777777777               //   9999999999999   5555555555555
 00       00    99         99             //     00       00    77        77              //    99         99   55
00         00   99         99            //     00         00            77              //     99         99   55
00         00   99         99           //      00         00           77              //      99         99   55
00         00   9999999999999          //       00         00          77              //       9999999999999   555555555555
00         00    999999999999         //        00         00          77             //         999999999999   5555555555555
00         00              99        //         00         00         77             //                    99              55
00         00              99       //          00         00         77            //                     99              55
 00       00               99      //            00       00         77            //                      99   55         55
  000000000     9999999999999     //              000000000          77           //            9999999999999   5555555555555
   0000000      999999999999     //                0000000           77          //             999999999999     55555555555

   0000000               44                      33333333333        0000000                         0000000         0000000
  000000000             444                     3333333333333      000000000                       000000000       000000000
 00       00           4444           :::       33         33     00       00          :::        00       00     00       00
00         00         44 44           :::                  33    00         00         :::       00         00   00         00
00         00        44  44           :::                  33    00         00         :::       00         00   00         00
00         00       44   44                              333     00         00                   00         00   00         00
00         00      44    44                              333     00         00                   00         00   00         00
00         00    444444444444         :::                  33    00         00         :::       00         00   00         00
00         00   4444444444444         :::                  33    00         00         :::       00         00   00         00
 00       00             44           :::       33         33     00       00          :::        00       00     00       00
  000000000              44                     3333333333333      000000000                       000000000       000000000
   0000000               44                      33333333333        0000000                         0000000         0000000
 sssssssssss     ccccccccccc       aaaaaaaaa     ll              eeeeeeeeeeeee
sssssssssssss   ccccccccccccc    aaaaaaaaaaa    ll              eeeeeeeeeeeee
ss         ss   cc         cc   aa         aa   ll              ee
ss              cc              aa         aa   ll              ee
ss              cc              aa         aa   ll              ee
ssssssssssss    cc              aaaaaaaaaaaaa   ll              eeeeeeeee
 ssssssssssss   cc              aaaaaaaaaaaaa   ll              eeeeeeeee
           ss   cc              aa         aa   ll              ee
           ss   cc              aa         aa   ll              ee
ss         ss   cc         cc   aa         aa   ll              ee
sssssssssssss   ccccccccccccc   aa         aa   lllllllllllll   eeeeeeeeeeeee
 sssssssssss     ccccccccccc    aa         aa   lllllllllllll   eeeeeeeeeeeee
```

```
*********************************************************************************
*********************************************************************************
*********************************************************************************
*****                                                                       *****
*****                  program verification information                     *****
*****                                                                       *****
*****                 code system:    scale  version:     4.3               *****
*****                                                                       *****
*********************************************************************************
*********************************************************************************
*****                                                                       *****
*****                                                                       *****
*****              program:  kenova                                         *****
*****                                                                       *****
*****        creation date:  08/13/96                                       *****
*****                                                                       *****
*****              library:  /scale4.3/bin                                  *****
*****                                                                       *****
*****                                                                       *****
*****      production code:  kenova                                         *****
*****                                                                       *****
*****              version:  3.7                                            *****
*****                                                                       *****
*****              jobname:  x4s                                            *****
*****                                                                       *****
*****    date of execution:  09/07/95                                       *****
*****                                                                       *****
*****    time of execution:  04:30:00                                       *****
*****                                                                       *****
*****                                                                       *****
*********************************************************************************
*********************************************************************************
*********************************************************************************
```

```
********************************************************************************************************************
***                                                                                                              ***
***                              sample problem 4  2c8 15.24 cm paraffin refl automatic refl                     ***
***                                                                                                              ***
********************************************************************************************************************
***                                 ******       numeric parameters       ******                                 ***
***                                                                                                              ***
***                                                                                                              ***
***                     tme         maximum problem time (min)                        120.00                     ***
***                                                                                                              ***
***                     tba         time per generation (min)                           0.50                     ***
***                                                                                                              ***
***                     gen         number of generations                                103                     ***
***                                                                                                              ***
***                     npg         number per generation                                300                     ***
***                                                                                                              ***
***                     nsk         number of generations to be skipped                    3                     ***
***                                                                                                              ***
***                     beg         beginning generation number                            1                     ***
***                                                                                                              ***
***                     res         generations between checkpoints                        0                     ***
***                                                                                                              ***
***                     x1d         number of extra 1-d cross sections                     1                     ***
***                                                                                                              ***
***                     nbk         neutron bank size                                    325                     ***
***                                                                                                              ***
***                     xnb         extra positions in neutron bank                        0                     ***
***                                                                                                              ***
***                     nfb         fission bank size                                    300                     ***
***                                                                                                              ***
***                     xfb         extra positions in fission bank                        0                     ***
***                                                                                                              ***
***                     wta         default value of weight average                   0.5000                     ***
***                                                                                                              ***
***                     wth         weight high for splitting                         3.0000                     ***
***                                                                                                              ***
***                     wtl         weight low for russian roulette                   0.3333                     ***
***                                                                                                              ***
***                     rnd         starting random number                       BB827100001                     ***
***                                                                                                              ***
***                     nb8         number of d.a. blocks on unit  8                     200                     ***
***                                                                                                              ***
***                     nl8         length of d.a. blocks on unit  8                     512                     ***
***                                                                                                              ***
***                     adj         mode of calculation                              forward                     ***
***                                                                                                              ***
***                                 input data written on restart unit                    no                     ***
***                                                                                                              ***
***                                 binary data interface                                 no                     ***
***                                                                                                              ***
***                                                                                                              ***
********************************************************************************************************************
********************************************************************************************************************
```

```
*********************************************************************************************************************
*********************************************************************************************************************
***                                                                                                               ***
***                              sample problem 4  2c8 15.24 cm paraffin refl automatic refl                      ***
***                                                                                                               ***
*********************************************************************************************************************
***                                 ******      logical parameters        ******                                  ***
***                                                                                                               ***
***  run  execute problem after checking data    yes            plt  plot picture map(s)                      yes ***
***                                                                                                               ***
***  flx  compute flux                           yes            fdn  compute fission densities                yes ***
***                                                                                                               ***
***  smu  compute avg unit self-multiplication    no            nub  compute nu-bar & avg fission group       yes ***
***                                                                                                               ***
***  mku  compute matrix k-eff by unit number     no            mkp  compute matrix k-eff by unit location     no ***
***                                                                                                               ***
***  cku  compute cofactor k-eff by unit number   no            ckp  compute cofactor k-eff by unit location   no ***
***                                                                                                               ***
***  fmu  print fiss prod matrix by unit number   no            fmp  print fiss prod matrix by unit location   no ***
***                                                                                                               ***
***  mkh  compute matrix k-eff by hole number     no            mka  compute matrix k-eff by array number      no ***
***                                                                                                               ***
***  ckh  compute cofactor k-eff by hole number   no            cka  compute cofactor k-eff by array number    no ***
***                                                                                                               ***
***  fmh  print fiss prod matrix by hole number   no            fma  print fiss prod matrix by array number    no ***
***                                                                                                               ***
***  hhl  collect matrix by highest hole level    no            hal  collect matrix by highest array level     no ***
***                                                                                                               ***
***  amx  print all mixed cross sections          no            far  print fis. and abs. by region             no ***
***                                                                                                               ***
***  xs1  print 1-d mixture x-sections            no            gas  print far by group                        no ***
***                                                                                                               ***
***  xs2  print 2-d mixture x-sections            no            pax  print xsec-albedo correlation tables      no ***
***                                                                                                               ***
***  xap  print mixture angles & probabilities    no            pwt  print weight average array               yes ***
***                                                                                                               ***
***  pki  print fission spectrum                  no            pgm  print input geometry                      no ***
***                                                                                                               ***
***  p1d  print extra 1-d cross sections          no            bug  print debug information                   no ***
***                                                                                                               ***
***                                                             trk  print tracking information                no ***
***                                                                                                               ***
***                                                                                                               ***
*********************************************************************************************************************
*********************************************************************************************************************
*********************************************************************************************************************
                                      parameter input completed

                        ........    0 io's were used reading the parameter data      ........

                       *************** data reading completed ***************
```

```
*********************************************************************************************************
***                                                                                                   ***
***                           sample problem 4  2c8 15.24 cm paraffin refl automatic refl              ***
***                                                                                                   ***
*********************************************************************************************************
*********************************************************************************************************
***                                                                                                   ***
***      unit                                               volume                                    ***
***     number              data set name                    name       unit function                 ***
***     ------              -------------                    ----       -------------                 ***
***                                                                                                   ***
***   xsc  14     ft14f001                                              mixed cross sections          ***
***                                                                                                   ***
***   alb  79     /scale4.3/data/albedos                                input albedos                 ***
***                                                                                                   ***
***   wts  80     /scale4.3/data/weights                                input weights                 ***
***                                                                                                   ***
***   skt  16      unknown                                              write scratch data            ***
***                                                                                                   ***
***   lib   4     ft04f001                                              input ampx working library    ***
***                                                                                                   ***
***         8     ft08f001                                              input data direct access      ***
***                                                                                                   ***
***         9      unknown                                              super grouped direct access   ***
***                                                                                                   ***
***        10      unknown                                              xsec mixing direct access     ***
***                                                                                                   ***
*********************************************************************************************************
                         ........    0 io's were used preparing input data    ........


                          cross sections read from the ampx working library on unit    4
```

```
                                      sample problem 4  2c8 15.24 cm paraffin refl automatic refl
                                                 mixing table

                                       number of scattering angles =  2
                                    cross section message threshold =3.0e-05


 mixture =      1          density(g/cc) =  18.760
 nuclide    atom-dens.   wgt. frac.     za      awt              nuclide title
 1092234  4.82716e-04  9.99999e-03   92234   234.0405      uranium-234    endf/b-iv mat 1043                      updated 08/12/94
 1092235  4.47971e-02  9.32000e-01   92235   235.0441      uranium-235    endf/b-iv mat 1261                      updated 08/12/94
 1092236  9.57231e-05  2.00000e-03   92236   236.0458     u-236 1163 sigo=5+4 newxlacs p-3 293k f-1/e-m(1.+5)     updated 08/12/94
 1092238  2.65767e-03  5.60000e-02   92238   238.0510      uranium-238    endf/b-iv mat 1262                      updated 08/12/94

 mixture =      2          density(g/cc) = 0.92999
 nuclide    atom-dens.   wgt. frac.     za      awt              nuclide title
 10001001  8.26407e-02  1.48689e-01   1001     1.0077      hydrogen       endf/b-iv mat 1269/thrm1002              updated 08/12/94
 10006012  3.97311e-02  8.51311e-01   6000    12.0001      carbon-12      endf/b-iv mat 1274/thrm1065              updated 08/12/94


                                   10001001     hydrogen       endf/b-iv mat 1269/thrm1002            updated 08/12/94
                                   10006012     carbon-12      endf/b-iv mat 1274/thrm1065            updated 08/12/94
                                    1092234     uranium-234    endf/b-iv mat 1043                     updated 08/12/94
                                    1092235     uranium-235    endf/b-iv mat 1261                     updated 08/12/94
                                    1092236    u-236 1163 sigo=5+4 newxlacs p-3 293k f-1/e-m(1.+5)    updated 08/12/94
                                    1092238     uranium-238    endf/b-iv mat 1262                     updated 08/12/94

 keno message number k5-222       2 transfers for mixture     2 were corrected for bad moments.

                                      ........     0 io's were used mixing cross-sections        ........

                                       1-d cross section array id numbers
                                          1  2002  1452    27    18  1018

                                      ........     0 io's were used preparing the cross sections     ........
```

```
**************************************************************************************************
***                                                                                            ***
***          sample problem 4  2c8 15.24 cm paraffin refl automatic refl                       ***
***                                                                                            ***
**************************************************************************************************
**************************************************************************************************
***                                                                                            ***
***                               ****** additional information ******                         ***
***                                                                                            ***
***  number of energy groups              27       use lattice geometry                   yes  ***
***                                                                                            ***
***  no. of fission spectrum source group  1       global array number                      1  ***
***                                                                                            ***
***  no. of scattering angles in xsecs     2       number of units in the global x dir.     2  ***
***                                                                                            ***
***  entries/neutron in the neutron bank  16       number of units in the global y dir.     2  ***
***                                                                                            ***
***  entries/neutron in the fission bank   9       number of units in the global z dir.     2  ***
***                                                                                            ***
***  number of mixtures used               2       use a global reflector                 yes  ***
***                                                                                            ***
***  number of bias id's used              7       use nested holes                        no  ***
***                                                                                            ***
***  number of differential albedos used   0       number of holes                          0  ***
***                                                                                            ***
***  total input geometry regions          9       maximum hole nesting level               0  ***
***                                                                                            ***
***  number of geometry regions used       9       use nested arrays                       no  ***
***                                                                                            ***
***  largest geometry unit number          2       number of arrays used                    1  ***
***                                                                                            ***
***  largest array number                  1       maximum array nesting level              1  ***
***                                                                                            ***
***                                                                                            ***
***  +x boundary condition            vacuum       -x boundary condition               vacuum  ***
***                                                                                            ***
***  +y boundary condition            vacuum       -y boundary condition               vacuum  ***
***                                                                                            ***
***  +z boundary condition            vacuum       -z boundary condition               vacuum  ***
***                                                                                            ***
**************************************************************************************************
```

```
*********************************************************************************************
***                                                                                       ***
***           sample problem 4  2c8 15.24 cm paraffin refl automatic refl                 ***
***                                                                                       ***
*********************************************************************************************
*********************************************************************************************
***                                                                                       ***
***                        ****** space and supergroup information ******                 ***
***                                                                                       ***
***         40000 words is the total space available.                                     ***
***                                                                                       ***
***          8819 words were used for non-supergroup storage.                             ***
***                                                                                       ***
***         31181 words of storage are available for supergrouped data.                   ***
***                                                                                       ***
***         39882 words of storage are available for constructing the supergroups.        ***
***                                                                                       ***
***         31121 words of storage are available to each supergroup.                      ***
***                                                                                       ***
***           340 words are needed for the largest group.                                 ***
***                                                                                       ***
***          9375 words of storage is sufficient to run this problem.                     ***
***                                                                                       ***
***         13535 words of storage will allow the problem to run with one supergroup.     ***
***                                                                                       ***
***         13888 words of storage will be used to run this problem.                      ***
***                                                                                       ***
*********************************************************************************************
*********************************************************************************************
***                                                                                       ***
***                        starting      ending       xsec        albedo      total       ***
***     supergroup          group         group       length      length      length      ***
***                                                                                       ***
***                                                                                       ***
***          1                1            27          678           0         4656       ***
***                                                                                       ***
*********************************************************************************************

             ........    0 io's were used in supergrouping    ........

             ........    0 io's were used loading the data    ........
```

```
                                        sample problem 4  2c8 15.24 cm paraffin refl automatic refl
                    media bias      geometry description for those units utilized in this problem
 region              num   id


                                                    -----   unit     1   -----

  1 cylinder          1   1  radius =  5.7480     +z =  5.3825     -z = -5.3825     centerline is at  x =  0.0000      y =  0.0000

  2 cuboid            0   1      +x =  11.740     -x = -11.740     +y =  11.740     -y = -11.740     +z =  11.375     -z = -11.375

                                        *******************   global   *******************
                                        -----   unit     2  external to lattice  1   -----

  1 array number      1          +x =  23.480     -x = -23.480     +y =  23.480     -y = -23.480     +z =  22.750     -z = -22.750

  2 cuboid            2   2      +x =  26.480     -x = -26.480     +y =  26.480     -y = -26.480     +z =  25.750     -z = -25.750

  3 cuboid            2   3      +x =  29.480     -x = -29.480     +y =  29.480     -y = -29.480     +z =  28.750     -z = -28.750

  4 cuboid            2   4      +x =  32.480     -x = -32.480     +y =  32.480     -y = -32.480     +z =  31.750     -z = -31.750

  5 cuboid            2   5      +x =  35.480     -x = -35.480     +y =  35.480     -y = -35.480     +z =  34.750     -z = -34.750

  6 cuboid            2   6      +x =  38.480     -x = -38.480     +y =  38.480     -y = -38.480     +z =  37.750     -z = -37.750

  7 cuboid            2   7      +x =  38.720     -x = -38.720     +y =  38.720     -y = -38.720     +z =  37.990     -z = -37.990
```

```
                 sample problem 4  2c8 15.24 cm paraffin refl automatic refl
                volumes for those units utilized in  this problem

                     geometry                              cumulative
unit     region       region          volume                 volume

  1         1           1         1.11737e+03 cm**3       1.11737e+03 cm**3
            2           2         1.14249e+04 cm**3       1.25423e+04 cm**3

surrounding geometry volumes -  geometry region      3 is an array placement boundary region

  2         1           3         1.00338e+05 cm**3       1.00338e+05 cm**3
            2           4         4.41067e+04 cm**3       1.44445e+05 cm**3
            3           5         5.54410e+04 cm**3       1.99886e+05 cm**3
            4           6         6.80712e+04 cm**3       2.67957e+05 cm**3
            5           7         8.19974e+04 cm**3       3.49955e+05 cm**3
            6           8         9.72197e+04 cm**3       4.47175e+05 cm**3
            7           9         8.47406e+03 cm**3       4.55649e+05 cm**3


                unit      uses     region      mixture        total volume

                  1         8        1            1         8.93897e+03 cm**3
                                     2            0         9.13995e+04 cm**3

                  2         1        1                      1.00338e+05 cm**3
                                     2            2         4.41067e+04 cm**3
                                     3            2         5.54410e+04 cm**3
                                     4            2         6.80712e+04 cm**3
                                     5            2         8.19974e+04 cm**3
                                     6            2         9.72197e+04 cm**3
                                     7            2         8.47406e+03 cm**3


                              total mixture volumes
                        mixture         total volume              mass(g)
                           0          9.13995e+04 cm**3
                           1          8.93897e+03 cm**3         1.67695e+05
                           2          3.55310e+05 cm**3         3.30433e+05
*************************************************************************************************************
***                                                                                                       ***
***                                           biasing information                                         ***
***                                                                                                       ***
***     weighting intervals   1 to    6 for paraffin    ,mat id=  400 will be used for bias id's    2 to    7       ***
***                                                                                                       ***
***     a default weight of    0.500 will be used for all other bias id's.                                ***
***                                                                                                       ***
*************************************************************************************************************
```

```
                              sample problem 4  2c8 15.24 cm paraffin refl automatic refl

                                                 group dependent weights

energy        bias          bias          bias          bias          bias          bias          bias
group          id    1       id    2       id    3       id    4       id    5       id    6       id    7
-----      ----------    ----------    ----------    ----------    ----------    ----------    ----------
   1       5.00000e-01   6.18390e-01   8.85090e-01   1.25430e+00   1.80190e+00   2.61220e+00   3.80230e+00
   2       5.00000e-01   5.91790e-01   8.52020e-01   1.33350e+00   2.23270e+00   3.89200e+00   6.93920e+00
   3       5.00000e-01   5.90830e-01   8.85190e-01   1.52270e+00   2.90060e+00   5.87640e+00   1.23420e+01
   4       5.00000e-01   5.86770e-01   9.05420e-01   1.68610e+00   3.57900e+00   8.22490e+00   1.98220e+01
   5       5.00000e-01   5.81890e-01   9.30070e-01   1.88130e+00   4.43390e+00   1.14320e+01   3.10120e+01
   6       5.00000e-01   5.74820e-01   9.87290e-01   2.28780e+00   6.30330e+00   1.90190e+01   5.96620e+01
   7       5.00000e-01   5.87290e-01   1.15630e+00   3.20150e+00   1.04420e+01   3.60420e+01   1.23200e+02
   8       5.00000e-01   6.20770e-01   1.41080e+00   4.44510e+00   1.57830e+01   5.64360e+01   1.91970e+02
   9       5.00000e-01   6.57610e-01   1.64320e+00   5.50540e+00   2.00300e+01   7.15900e+01   2.40440e+02
  10       5.00000e-01   6.87950e-01   1.83600e+00   6.38830e+00   2.34660e+01   8.35140e+01   2.77740e+02
  11       5.00000e-01   6.92740e-01   1.94220e+00   6.96040e+00   2.56820e+01   9.09270e+01   2.99680e+02
  12       5.00000e-01   6.98060e-01   2.06680e+00   7.59760e+00   2.80730e+01   9.87930e+01   3.22680e+02
  13       5.00000e-01   7.01860e-01   2.17810e+00   8.12680e+00   3.00000e+01   1.05060e+02   3.40830e+02
  14       5.00000e-01   7.12050e-01   2.33100e+00   8.78950e+00   3.23810e+01   1.12840e+02   3.63710e+02
  15       5.00000e-01   7.34300e-01   2.53080e+00   9.60390e+00   3.53030e+01   1.22520e+02   3.92830e+02
  16       5.00000e-01   7.84110e-01   2.80270e+00   1.06540e+01   3.91200e+01   1.35510e+02   4.33480e+02
  17       5.00000e-01   7.83000e-01   2.82230e+00   1.07360e+01   3.93890e+01   1.36290e+02   4.35350e+02
  18       5.00000e-01   8.11260e-01   2.96800e+00   1.12930e+01   4.14180e+01   1.43220e+02   4.57120e+02
  19       5.00000e-01   8.32890e-01   3.08460e+00   1.17400e+01   4.30330e+01   1.48700e+02   4.74160e+02
  20       5.00000e-01   8.41560e-01   3.14930e+00   1.19880e+01   4.38940e+01   1.51420e+02   4.81790e+02
  21       5.00000e-01   8.37120e-01   3.15740e+00   1.20190e+01   4.39320e+01   1.51180e+02   4.79580e+02
  22       5.00000e-01   8.36030e-01   3.17070e+00   1.20670e+01   4.40680e+01   1.51460e+02   4.79740e+02
  23       5.00000e-01   8.61360e-01   3.32330e+00   1.26400e+01   4.60960e+01   1.58130e+02   4.99640e+02
  24       5.00000e-01   8.86690e-01   3.46960e+00   1.31820e+01   4.79880e+01   1.64220e+02   5.17330e+02
  25       5.00000e-01   9.02200e-01   3.55250e+00   1.34830e+01   4.90100e+01   1.67360e+02   5.25870e+02
  26       5.00000e-01   9.20350e-01   3.64340e+00   1.38170e+01   5.01620e+01   1.71010e+02   5.36250e+02
  27       5.00000e-01   9.43470e-01   3.74870e+00   1.42020e+01   5.14930e+01   1.75220e+02   5.48200e+02

                          ........   0 io's were used in keno-v before tracking      .......

                          ........  0.00000 minutes were used processing data.       ........

volume fraction of fissile material in the core= 8.90881e-02

start type 0 was used.

the neutrons were started with a flat distribution in a cuboid defined by:
                     +x= 2.34800e+01  -x=-2.34800e+01  +y= 2.34800e+01  -y=-2.34800e+01  +z= 2.27500e+01  -z=-2.27500e+01
the flag to start neutrons in the reflector was turned off

0.00000 minutes were required for starting.   total elapsed time is 0.00000 minutes.
```

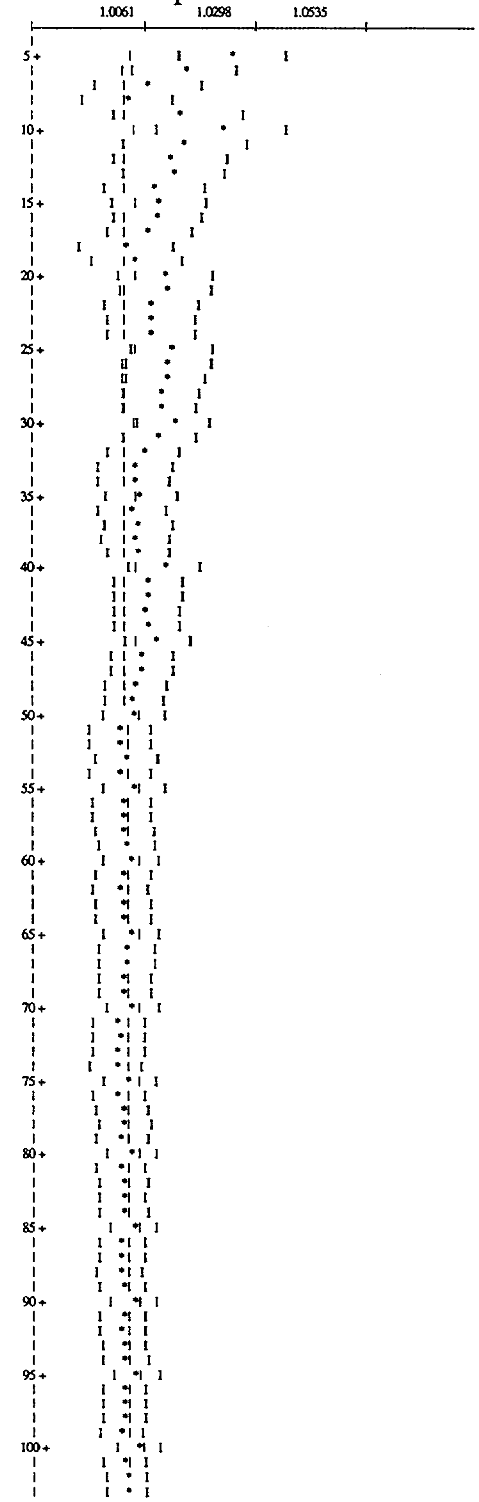
```
sample problem 4  2c8 15.24 cm paraffin refl automatic refl

                generation     elapsed time      average        avg k-eff        matrix       matrix k-eff
  generation    k-effective      minutes       k-effective      deviation      k-effective      deviation
       1        1.07614e+00    2.13333e-02    1.00000e+00     0.00000e+00    0.00000e+00    0.00000e+00
       2        1.00904e+00    4.26667e-02    1.00000e+00     0.00000e+00    0.00000e+00    0.00000e+00
       3        1.05608e+00    4.26667e-02    1.05608e+00     0.00000e+00    0.00000e+00    0.00000e+00
       4        1.05349e+00    4.26667e-02    1.05478e+00     1.29694e-03    0.00000e+00    0.00000e+00
       5        1.02349e+00    6.40000e-02    1.04435e+00     1.04579e-02    0.00000e+00    0.00000e+00
       6        1.00405e+00    8.53333e-02    1.03428e+00     1.24976e-02    0.00000e+00    0.00000e+00
       7        9.81694e-01    8.53333e-02    1.02376e+00     1.42938e-02    0.00000e+00    0.00000e+00
       8        9.87205e-01    1.06667e-01    1.01767e+00     1.31654e-02    0.00000e+00    0.00000e+00
       9        1.10194e+00    1.06667e-01    1.02971e+00     1.63932e-02    0.00000e+00    0.00000e+00
      10        1.09279e+00    1.06667e-01    1.03759e+00     1.62398e-02    0.00000e+00    0.00000e+00
      11        9.70654e-01    1.28000e-01    1.03015e+00     1.61382e-02    0.00000e+00    0.00000e+00
      12        9.91371e-01    1.28000e-01    1.02628e+00     1.49464e-02    0.00000e+00    0.00000e+00
      13        1.03029e+00    1.49333e-01    1.02664e+00     1.35245e-02    0.00000e+00    0.00000e+00
      14        9.58084e-01    1.70667e-01    1.02093e+00     1.36039e-02    0.00000e+00    0.00000e+00
      15        9.96819e-01    1.70667e-01    1.01907e+00     1.26504e-02    0.00000e+00    0.00000e+00
      16        1.05540e+00    1.92000e-01    1.02167e+00     1.19961e-02    0.00000e+00    0.00000e+00
      17        9.80279e-01    1.92000e-01    1.01891e+00     1.15036e-02    0.00000e+00    0.00000e+00
      18        9.09329e-01    1.92000e-01    1.01206e+00     1.27552e-02    0.00000e+00    0.00000e+00
      19        1.06193e+00    2.13333e-01    1.01499e+00     1.23353e-02    0.00000e+00    0.00000e+00
      20        1.10516e+00    2.34667e-01    1.02000e+00     1.26628e-02    0.00000e+00    0.00000e+00
      21        1.08266e+00    2.34667e-01    1.02330e+00     1.24235e-02    0.00000e+00    0.00000e+00
      22        9.27772e-01    2.56000e-01    1.01852e+00     1.27171e-02    0.00000e+00    0.00000e+00
      23        1.02337e+00    2.56000e-01    1.01876e+00     1.20986e-02    0.00000e+00    0.00000e+00
      24        1.01905e+00    2.56000e-01    1.01877e+00     1.15355e-02    0.00000e+00    0.00000e+00
      25        1.07737e+00    2.56000e-01    1.02132e+00     1.13132e-02    0.00000e+00    0.00000e+00
      26        1.07599e+00    2.77333e-01    1.02359e+00     1.10685e-02    0.00000e+00    0.00000e+00
      27        1.00694e+00    2.98667e-01    1.02293e+00     1.06374e-02    0.00000e+00    0.00000e+00
      28        9.83438e-01    2.98667e-01    1.02141e+00     1.03324e-02    0.00000e+00    0.00000e+00
      29        1.00594e+00    3.20000e-01    1.02084e+00     9.95883e-03    0.00000e+00    0.00000e+00
      30        1.04384e+00    3.41333e-01    1.02166e+00     9.63168e-03    0.00000e+00    0.00000e+00
      31        9.91331e-01    3.41333e-01    1.02061e+00     9.35227e-03    0.00000e+00    0.00000e+00
      32        8.82335e-01    3.41333e-01    1.01600e+00     1.01429e-02    0.00000e+00    0.00000e+00
      33        9.52266e-01    3.41333e-01    1.01395e+00     1.00234e-02    0.00000e+00    0.00000e+00
      34        9.88702e-01    3.62667e-01    1.01316e+00     9.73715e-03    0.00000e+00    0.00000e+00
      35        9.77159e-01    3.84000e-01    1.01207e+00     9.50031e-03    0.00000e+00    0.00000e+00
      36        1.02642e+00    3.84000e-01    1.01249e+00     9.22632e-03    0.00000e+00    0.00000e+00
      37        1.07467e+00    4.05333e-01    1.01427e+00     9.13331e-03    0.00000e+00    0.00000e+00
      38        9.87657e-01    4.26667e-01    1.01353e+00     8.90670e-03    0.00000e+00    0.00000e+00
      39        1.04541e+00    4.26667e-01    1.01439e+00     8.70539e-03    0.00000e+00    0.00000e+00
      40        1.17598e+00    4.26667e-01    1.01864e+00     9.48045e-03    0.00000e+00    0.00000e+00
      41        9.48441e-01    4.48000e-01    1.01684e+00     9.40796e-03    0.00000e+00    0.00000e+00
      42        1.03695e+00    4.48000e-01    1.01734e+00     9.18352e-03    0.00000e+00    0.00000e+00
      43        9.74939e-01    4.69333e-01    1.01631e+00     9.01625e-03    0.00000e+00    0.00000e+00
      44        1.03266e+00    4.90667e-01    1.01670e+00     8.80757e-03    0.00000e+00    0.00000e+00
      45        1.00083e+00    4.90667e-01    1.01633e+00     8.60822e-03    0.00000e+00    0.00000e+00
      46        9.73328e-01    4.90667e-01    1.01535e+00     8.46689e-03    0.00000e+00    0.00000e+00
      47        1.00088e+00    4.90667e-01    1.01503e+00     8.28285e-03    0.00000e+00    0.00000e+00
      48        9.40533e-01    5.12000e-01    1.01341e+00     8.26109e-03    0.00000e+00    0.00000e+00
      49        9.68779e-01    5.33333e-01    1.01246e+00     8.13900e-03    0.00000e+00    0.00000e+00
      50        8.67000e-01    5.33333e-01    1.00943e+00     8.52448e-03    0.00000e+00    0.00000e+00
      51        9.72944e-01    5.54667e-01    1.00869e+00     8.38184e-03    0.00000e+00    0.00000e+00
```

```
   52      9.88456e-01     5.76000e-01     1.00828e+00     8.22246e-03     0.00000e+00     0.00000e+00
   53      1.11720e+00     5.76000e-01     1.01042e+00     8.33777e-03     0.00000e+00     0.00000e+00
   54      9.24298e-01     5.76000e-01     1.00876e+00     8.34191e-03     0.00000e+00     0.00000e+00
   55      1.05073e+00     5.97333e-01     1.00955e+00     8.22124e-03     0.00000e+00     0.00000e+00
   56      9.88329e-01     5.97333e-01     1.00916e+00     8.07713e-03     0.00000e+00     0.00000e+00
   57      1.01124e+00     6.18667e-01     1.00920e+00     7.92900e-03     0.00000e+00     0.00000e+00
   58      1.03400e+00     6.18667e-01     1.00964e+00     7.79871e-03     0.00000e+00     0.00000e+00
   59      1.03338e+00     6.40000e-01     1.01006e+00     7.67198e-03     0.00000e+00     0.00000e+00
   60      9.05864e-01     6.40000e-01     1.00826e+00     7.74964e-03     0.00000e+00     0.00000e+00
   61      1.05193e+00     6.40000e-01     1.00900e+00     7.65303e-03     0.00000e+00     0.00000e+00
   62      9.56608e-01     6.61333e-01     1.00813e+00     7.57490e-03     0.00000e+00     0.00000e+00
   63      1.08604e+00     6.82667e-01     1.00941e+00     7.55837e-03     0.00000e+00     0.00000e+00
   64      1.01393e+00     6.82667e-01     1.00948e+00     7.43582e-03     0.00000e+00     0.00000e+00
   65      9.26224e-01     7.04000e-01     1.00816e+00     7.43522e-03     0.00000e+00     0.00000e+00
   66      1.12147e+00     7.25333e-01     1.00993e+00     7.52926e-03     0.00000e+00     0.00000e+00
   67      1.00325e+00     7.25333e-01     1.00982e+00     7.41323e-03     0.00000e+00     0.00000e+00
   68      9.95913e-01     7.25333e-01     1.00961e+00     7.30309e-03     0.00000e+00     0.00000e+00
   69      9.95462e-01     7.25333e-01     1.00940e+00     7.19636e-03     0.00000e+00     0.00000e+00
   70      9.54698e-01     7.46667e-01     1.00860e+00     7.13524e-03     0.00000e+00     0.00000e+00
   71      9.34212e-01     7.46667e-01     1.00752e+00     7.11324e-03     0.00000e+00     0.00000e+00
   72      1.05476e+00     7.68000e-01     1.00820e+00     7.04329e-03     0.00000e+00     0.00000e+00
   73      9.48724e-01     7.89333e-01     1.00736e+00     6.99372e-03     0.00000e+00     0.00000e+00
   74      9.65425e-01     7.89333e-01     1.00678e+00     6.92045e-03     0.00000e+00     0.00000e+00
   75      1.06326e+00     8.10667e-01     1.00755e+00     6.86871e-03     0.00000e+00     0.00000e+00
   76      1.00716e+00     8.10667e-01     1.00754e+00     6.77525e-03     0.00000e+00     0.00000e+00
   77      1.10511e+00     8.10667e-01     1.00884e+00     6.80971e-03     0.00000e+00     0.00000e+00
   78      1.05953e+00     8.32000e-01     1.00951e+00     6.75253e-03     0.00000e+00     0.00000e+00
   79      9.32674e-01     8.53333e-01     1.00851e+00     6.73855e-03     0.00000e+00     0.00000e+00
   80      9.82992e-01     8.74667e-01     1.00819e+00     6.65964e-03     0.00000e+00     0.00000e+00
   81      1.03351e+00     8.74667e-01     1.00851e+00     6.58261e-03     0.00000e+00     0.00000e+00
   82      1.05775e+00     8.74667e-01     1.00912e+00     6.52889e-03     0.00000e+00     0.00000e+00
   83      9.70738e-01     8.96000e-01     1.00865e+00     6.46517e-03     0.00000e+00     0.00000e+00
   84      1.03701e+00     8.96000e-01     1.00899e+00     6.39520e-03     0.00000e+00     0.00000e+00
   85      9.82943e-01     9.17333e-01     1.00868e+00     6.32547e-03     0.00000e+00     0.00000e+00
   86      9.78578e-01     9.17333e-01     1.00832e+00     6.25998e-03     0.00000e+00     0.00000e+00
   87      1.00319e+00     9.38667e-01     1.00826e+00     6.18619e-03     0.00000e+00     0.00000e+00
   88      9.51774e-01     9.60000e-01     1.00760e+00     6.14902e-03     0.00000e+00     0.00000e+00
   89      1.11055e+00     9.60000e-01     1.00879e+00     6.19204e-03     0.00000e+00     0.00000e+00
   90      1.02180e+00     9.60000e-01     1.00894e+00     6.12306e-03     0.00000e+00     0.00000e+00
   91      9.96239e-01     9.60000e-01     1.00879e+00     6.05555e-03     0.00000e+00     0.00000e+00
   92      9.60628e-01     9.81333e-01     1.00826e+00     6.01176e-03     0.00000e+00     0.00000e+00
   93      1.07787e+00     1.00267e+00     1.00902e+00     5.99434e-03     0.00000e+00     0.00000e+00
   94      1.02474e+00     1.00267e+00     1.00919e+00     5.93128e-03     0.00000e+00     0.00000e+00
   95      1.01674e+00     1.02400e+00     1.00927e+00     5.86772e-03     0.00000e+00     0.00000e+00
   96      9.66139e-01     1.02400e+00     1.00882e+00     5.82307e-03     0.00000e+00     0.00000e+00
   97      1.01563e+00     1.02400e+00     1.00889e+00     5.76190e-03     0.00000e+00     0.00000e+00
   98      9.96937e-01     1.04533e+00     1.00876e+00     5.70292e-03     0.00000e+00     0.00000e+00
   99      9.44801e-01     1.04533e+00     1.00810e+00     5.68221e-03     0.00000e+00     0.00000e+00
  100      1.05055e+00     1.06667e+00     1.00854e+00     5.64059e-03     0.00000e+00     0.00000e+00
  101      1.02279e+00     1.08800e+00     1.00868e+00     5.58518e-03     0.00000e+00     0.00000e+00
  102      1.08458e+00     1.10933e+00     1.00944e+00     5.58090e-03     0.00000e+00     0.00000e+00
  103      1.00571e+00     1.10933e+00     1.00940e+00     5.52549e-03     0.00000e+00     0.00000e+00
keno message number k5-123         execution terminated due to completion of the specified number of generations.
```

```
                                             sample problem 4  2c8 15.24 cm paraffin refl automatic refl

lifetime =  1.46804e-04 + or -  8.15112e-06             generation time =  8.48176e-05 + or -  2.27110e-06
nu bar   =  2.56274e+00 + or -  1.40640e-03       average fission group =  9.87690e+00 + or -  6.14742e-02
                     energy(ev) of the average lethargy causing fission =  9.65268e+03 + or -  5.05317e+02

no. of initial
 generations      average                          67 per cent           95 per cent           99 per cent       number of
   skipped      k-effective      deviation     confidence interval   confidence interval   confidence interval   histories

      3           1.00894     + or - 0.00556   1.00338 to 1.01450    0.99781 to 1.02006    0.99225 to 1.02562      30000
      4           1.00849     + or - 0.00584   1.00265 to 1.01432    0.99681 to 1.02016    0.99097 to 1.02600      29700
      5           1.00833     + or - 0.00565   1.00268 to 1.01399    0.99702 to 1.01964    0.99137 to 1.02530      29400
      6           1.00838     + or - 0.00596   1.00242 to 1.01434    0.99646 to 1.02030    0.99050 to 1.02625      29100
      7           1.00866     + or - 0.00577   1.00289 to 1.01442    0.99712 to 1.02019    0.99136 to 1.02595      28800
      8           1.00888     + or - 0.00606   1.00282 to 1.01495    0.99675 to 1.02101    0.99069 to 1.02707      28500
      9           1.00789     + or - 0.00580   1.00209 to 1.01369    0.99629 to 1.01949    0.99049 to 1.02529      28200
     10           1.00698     + or - 0.00588   1.00110 to 1.01286    0.99521 to 1.01874    0.98933 to 1.02463      27900
     11           1.00737     + or - 0.00584   1.00153 to 1.01321    0.99570 to 1.01905    0.98986 to 1.02489      27600
     12           1.00755     + or - 0.00599   1.00156 to 1.01353    0.99558 to 1.01952    0.98959 to 1.02550      27300
     17           1.00775     + or - 0.00618   1.00157 to 1.01392    0.99539 to 1.02010    0.98921 to 1.02628      25800
     22           1.00715     + or - 0.00624   1.00091 to 1.01339    0.99467 to 1.01963    0.98843 to 1.02587      24300
     27           1.00495     + or - 0.00641   0.99854 to 1.01137    0.99213 to 1.01778    0.98572 to 1.02419      22800
     32           1.00661     + or - 0.00661   1.00001 to 1.01322    0.99340 to 1.01983    0.98679 to 1.02644      21300
     37           1.00682     + or - 0.00696   0.99986 to 1.01378    0.99290 to 1.02074    0.98594 to 1.02771      19800
     42           1.00420     + or - 0.00687   0.99733 to 1.01106    0.99046 to 1.01793    0.98359 to 1.02480      18300
     47           1.00488     + or - 0.00743   0.99745 to 1.01231    0.99002 to 1.01974    0.98259 to 1.02718      16800
     52           1.01050     + or - 0.00748   1.00302 to 1.01798    0.99555 to 1.02546    0.98807 to 1.03294      15300
     57           1.00965     + or - 0.00767   1.00198 to 1.01731    0.99432 to 1.02498    0.98665 to 1.03264      13800
     62           1.01127     + or - 0.00810   1.00317 to 1.01937    0.99507 to 1.02747    0.98697 to 1.03556      12300
     67           1.00864     + or - 0.00795   1.00069 to 1.01660    0.99273 to 1.02455    0.98478 to 1.03250      10800
     72           1.01213     + or - 0.01008   1.00205 to 1.02221    0.99198 to 1.03229    0.98190 to 1.04236       9300
     77           1.01102     + or - 0.00885   1.00217 to 1.01986    0.99332 to 1.02871    0.98447 to 1.03756       7800
     82           1.01047     + or - 0.01013   1.00034 to 1.02060    0.99022 to 1.03073    0.98009 to 1.04086       6300
     87           1.01547     + or - 0.01198   1.00349 to 1.02744    0.99151 to 1.03942    0.97954 to 1.05140       4800
     92           1.01877     + or - 0.01333   1.00545 to 1.03210    0.99212 to 1.04542    0.97880 to 1.05875       3300
     97           1.01756     + or - 0.02347   0.99409 to 1.04104    0.97061 to 1.06451    0.94714 to 1.08798       1800
```

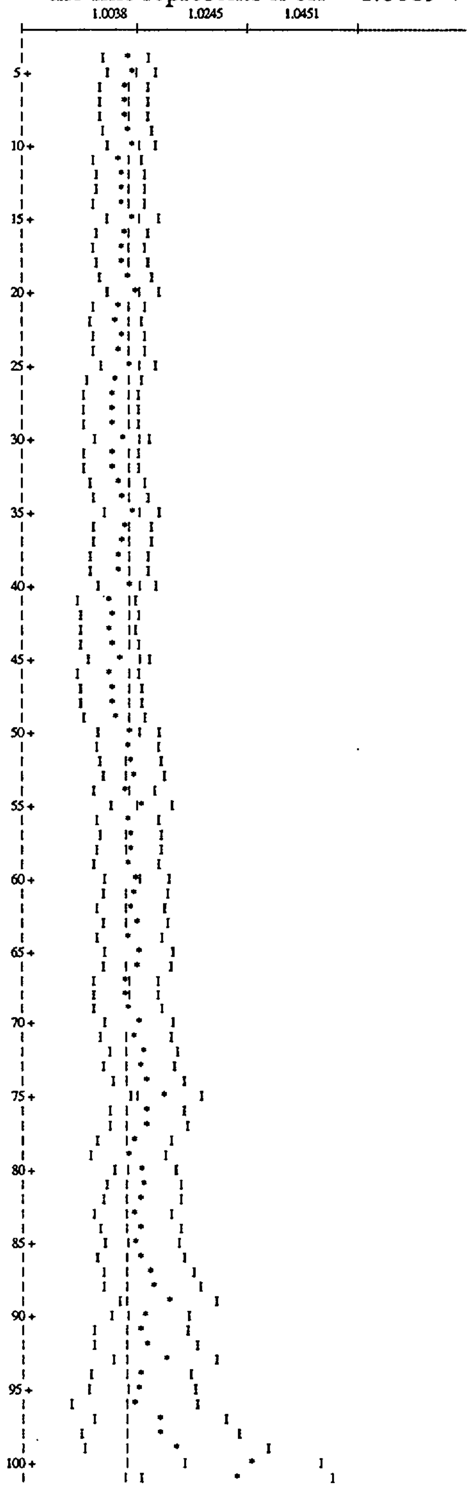
sample problem 4 2c8 15.24 cm paraffin refl automatic refl

plot of average k-effective by generation run.
the line represents k-eff = 1.0089 + or - 0.0056 which occurs for   103 generations run.



sample problem 4 2c8 15.24 cm paraffin refl automatic refl

plot of average k-effective by generation skipped.
the line represents k-eff = 1.0089 + or - 0.0056 which occurs for 3 generations skipped.



0 k-effective satisfies the chi**2 test for normality at the 95 % level

```
                         sample problem 4  2c8 15.24 cm paraffin refl  automatic refl
                                                                                                  skipping   3 generations
group  fission  unit    region     fissions       percent       absorptions     percent        leakage       percent
       fraction                                   deviation                     deviation                    deviation

  1     0.0201                   2.02600e-02      5.7062       6.05224e-03      5.5136      2.46712e-03     12.4387
  2     0.1000                   1.00877e-01      1.7334       3.48590e-02      1.7333      5.05600e-03     10.7242
  3     0.1241                   1.25203e-01      1.5041       4.80236e-02      1.5040      4.69872e-03     15.6597
  4     0.0793                   8.00374e-02      1.7430       3.23455e-02      1.7429      2.01191e-03     26.2031
  5     0.1065                   1.07465e-01      1.5611       4.57281e-02      1.5610      1.77354e-03     27.8548
  6     0.1451                   1.46420e-01      1.2994       6.67500e-02      1.2993      2.91366e-03     33.3053
  7     0.1053                   1.06196e-01      1.7570       5.27076e-02      1.7568      1.44770e-03     49.7664
  8     0.0320                   3.22408e-02      3.0359       1.80197e-02      3.0332      0.00000e+00      0.0000
  9     0.0110                   1.11419e-02      5.4242       6.42173e-03      5.3942      6.34005e-04    100.0000
 10     0.0115                   1.16177e-02      7.4252       6.83932e-03      7.3374      1.88126e-03    100.0000
 11     0.0128                   1.29595e-02      5.1535       8.15107e-03      5.0258      0.00000e+00      0.0000
 12     0.0080                   8.11131e-03      7.3803       5.26472e-03      7.0011      0.00000e+00      0.0000
 13     0.0064                   6.50624e-03      8.2909       4.47061e-03      7.4871      1.19510e-02     90.4102
 14     0.0059                   5.98307e-03      8.8545       4.52903e-03      7.3522      0.00000e+00      0.0000
 15     0.0032                   3.18029e-03     11.2688       2.67603e-03      8.7972      0.00000e+00      0.0000
 16     0.0022                   2.17128e-03     16.3498       1.55444e-03     12.0012      0.00000e+00      0.0000
 17     0.0011                   1.05956e-03     22.4097       7.90708e-04     16.6574      0.00000e+00      0.0000
 18     0.0008                   8.43787e-04     24.1038       6.05062e-04     16.7541      0.00000e+00      0.0000
 19     0.0025                   2.55658e-03     15.0323       1.59181e-03     11.0931      0.00000e+00      0.0000
 20     0.0066                   6.64838e-03      9.6695       4.82569e-03      6.5379      0.00000e+00      0.0000
 21     0.0021                   2.16581e-03     17.3109       1.81232e-03     10.5336      0.00000e+00      0.0000
 22     0.0043                   4.33107e-03     10.6130       4.41449e-03      5.9860      0.00000e+00      0.0000
 23     0.0311                   3.13841e-02      4.2642       4.52525e-02      3.3283      0.00000e+00      0.0000
 24     0.0691                   6.97149e-02      2.8327       1.40493e-01      3.7570      4.59931e-03    100.0000
 25     0.0495                   4.99745e-02      3.8238       1.33800e-01      3.7401      0.00000e+00      0.0000
 26     0.0484                   4.87827e-02      3.6228       1.83446e-01      3.8525      0.00000e+00      0.0000
 27     0.0110                   1.11056e-02      7.3323       7.65197e-02      5.0193      0.00000e+00      0.0000

 system total =                  1.00894e+00      0.5512       9.37944e-01      2.3802      3.94342e-02     30.7921
 elapsed time   1.10933 minutes
```

```
random number=       2b6f4c116a47
                                            sample problem 4  2c8 15.24 cm paraffin refl automatic refl




                                                   **** fission densities ****

                                         fission      percent          total
                           unit  region  density      deviation        fissions

                             1      1    1.129e-04      0.55           1.009e+00
                                    2    0.000e+00      0.00           0.000e+00

   global unit

                             2      1    0.000e+00      0.00           0.000e+00
                                    2    0.000e+00      0.00           0.000e+00
                                    3    0.000e+00      0.00           0.000e+00
                                    4    0.000e+00      0.00           0.000e+00
                                    5    0.000e+00      0.00           0.000e+00
                                    6    0.000e+00      0.00           0.000e+00
                                    7    0.000e+00      0.00           0.000e+00
```

```
                                        sample problem 4  2c8 15.24 cm paraffin refl automatic refl

fluxes for unit    1
        region   1           region   2

group   flux     percent     flux     percent
                deviation            deviation
  1   8.444e-06    5.14    2.589e-06    5.78
  2   7.029e-05    1.71    2.198e-05    2.02
  3   8.715e-05    1.35    2.965e-05    1.73
  4   6.012e-05    1.60    1.912e-05    2.08
  5   8.281e-05    1.50    2.679e-05    1.68
  6   1.259e-04    1.28    3.957e-05    1.31
  7   8.163e-05    1.59    2.839e-05    1.53
  8   1.678e-05    2.96    1.163e-05    2.89
  9   3.121e-06    5.62    7.981e-06    3.40
 10   1.413e-06    7.91    6.532e-06    3.60
 11   6.814e-07    6.97    6.516e-06    4.00
 12   6.910e-07    9.30    4.067e-06    5.00
 13   6.987e-07    9.38    3.932e-06    5.57
 14   8.732e-07   10.05    4.062e-06    5.65
 15   2.295e-07   13.46    2.154e-06    7.88
 16   1.409e-07   19.49    1.028e-06   11.00
 17   2.967e-08   32.15    5.056e-07   14.17
 18   1.385e-08   30.57    4.174e-07   16.25
 19   4.651e-08   22.22    7.937e-07   11.27
 20   8.542e-08   11.58    2.475e-06    6.79
 21   1.263e-08   19.58    8.219e-07   11.45
 22   2.545e-08   14.16    1.609e-06    7.62
 23   1.637e-07    5.71    1.152e-05    3.64
 24   2.291e-07    4.42    2.541e-05    2.84
 25   1.109e-07    5.50    1.863e-05    3.40
 26   7.266e-08    5.09    1.723e-05    3.34
 27   8.232e-09   10.13    4.113e-06    5.45
```

```
                              sample problem 4  2c8 15.24 cm paraffin refl automatic refl

fluxes for global unit
      region   1            region   2            region   3            region   4            region   5            region   6

group   flux     percent    flux     percent    flux     percent    flux     percent    flux     percent    flux     percent
               deviation           deviation           deviation           deviation           deviation           deviation
  1   0.000e+00    0.00  1.124e-06    5.60  6.040e-07    6.28  3.530e-07    7.85  2.057e-07    9.62  1.253e-07   10.02
  2   0.000e+00    0.00  8.231e-06    2.23  3.280e-06    2.75  1.351e-06    3.39  6.365e-07    5.96  2.919e-07    7.94
  3   0.000e+00    0.00  1.062e-05    1.71  3.946e-06    2.22  1.658e-06    3.45  6.991e-07    6.25  3.421e-07   11.58
  4   0.000e+00    0.00  6.865e-06    1.93  2.453e-06    2.47  8.889e-07    4.18  3.197e-07    8.08  1.191e-07   13.20
  5   0.000e+00    0.00  9.933e-06    1.76  3.424e-06    2.38  1.343e-06    3.57  5.578e-07    6.80  2.283e-07   16.16
  6   0.000e+00    0.00  1.567e-05    1.02  5.170e-06    1.80  1.851e-06    3.71  7.094e-07    7.69  2.582e-07   18.23
  7   0.000e+00    0.00  1.540e-05    1.03  5.536e-06    1.77  2.011e-06    4.23  7.169e-07   11.40  3.135e-07   36.46
  8   0.000e+00    0.00  1.069e-05    1.20  4.387e-06    1.93  1.688e-06    5.89  6.649e-07   11.02  1.856e-07   27.19
  9   0.000e+00    0.00  8.045e-06    1.26  3.577e-06    1.89  1.403e-06    5.34  4.940e-07   13.57  2.394e-07   33.68
 10   0.000e+00    0.00  6.932e-06    1.28  3.286e-06    1.93  1.261e-06    6.31  3.106e-07   19.29  1.836e-07   40.66
 11   0.000e+00    0.00  6.866e-06    1.28  3.548e-06    1.84  1.401e-06    6.64  5.136e-07   16.33  1.581e-07   51.13
 12   0.000e+00    0.00  4.626e-06    1.63  2.412e-06    2.66  9.080e-07    7.12  2.784e-07   20.16  1.613e-07   57.24
 13   0.000e+00    0.00  3.980e-06    1.46  2.272e-06    2.55  8.419e-07    6.88  2.954e-07   21.52  4.255e-08   57.23
 14   0.000e+00    0.00  4.133e-06    1.58  2.506e-06    2.29  9.428e-07    8.73  3.657e-07   19.83  3.459e-08   61.84
 15   0.000e+00    0.00  2.078e-06    2.09  1.310e-06    3.80  4.315e-07   11.19  1.459e-07   58.02  6.207e-08   73.54
 16   0.000e+00    0.00  1.103e-06    2.97  6.867e-07    5.18  2.925e-07   11.44  9.149e-08   31.83  2.700e-08   72.51
 17   0.000e+00    0.00  5.519e-07    3.72  2.891e-07    7.17  9.915e-08   17.17  2.791e-08   54.17  6.394e-09  100.00
 18   0.000e+00    0.00  4.525e-07    4.11  2.876e-07    8.58  1.064e-07   18.55  1.634e-08   56.38  0.000e+00    0.00
 19   0.000e+00    0.00  8.701e-07    2.76  5.371e-07    5.54  1.854e-07   16.27  7.411e-08   33.41  0.000e+00    0.00
 20   0.000e+00    0.00  2.885e-06    1.81  1.771e-06    3.98  6.651e-07   10.23  2.937e-07   21.91  4.071e-08   76.60
 21   0.000e+00    0.00  9.250e-07    3.19  5.914e-07    5.86  2.006e-07   14.88  1.090e-07   40.62  0.000e+00    0.00
 22   0.000e+00    0.00  2.120e-06    2.15  1.620e-06    4.78  6.674e-07   10.68  2.690e-07   31.81  2.143e-09  100.00
 23   0.000e+00    0.00  1.829e-05    1.89  1.593e-05    3.09  8.342e-06    7.62  3.375e-06   19.58  4.498e-07   46.56
 24   0.000e+00    0.00  4.363e-05    1.92  3.984e-05    3.31  2.028e-05    7.99  8.002e-06   18.78  2.051e-06   59.15
 25   0.000e+00    0.00  3.356e-05    1.96  3.128e-05    3.32  1.604e-05    7.79  6.256e-06   16.69  1.284e-06   50.37
 26   0.000e+00    0.00  3.476e-05    1.98  3.211e-05    3.17  1.645e-05    7.32  6.191e-06   18.46  1.163e-06   54.24
 27   0.000e+00    0.00  8.034e-06    2.01  7.315e-06    3.38  3.834e-06    8.00  1.486e-06   20.77  5.067e-07   57.72
```

```
                                            sample problem 4  2c8 15.24 cm paraffin refl automatic refl
fluxes for global unit
           region   7

group    flux      percent
                  deviation
  1    1.019e-07    13.60
  2    2.224e-07    21.01
  3    1.931e-07    16.18
  4    7.805e-08    25.01
  5    7.216e-08    26.51
  6    2.192e-07    37.31
  7    1.063e-07    41.65
  8    0.000e+00     0.00
  9    9.083e-08    57.13
 10    1.430e-07   100.00
 11    1.442e-08    70.39
 12    3.865e-08   100.00
 13    3.595e-07    72.65
 14    0.000e+00     0.00
 15    0.000e+00     0.00
 16    2.964e-08   100.00
 17    0.000e+00     0.00
 18    0.000e+00     0.00
 19    0.000e+00     0.00
 20    0.000e+00     0.00
 21    0.000e+00     0.00
 22    0.000e+00     0.00
 23    0.000e+00     0.00
 24    1.557e-07   100.00
 25    1.480e-08   100.00
 26    0.000e+00     0.00
 27    0.000e+00     0.00
```

```
sample problem 4  2c8 15.24 cm paraffin refl automatic refl
                                    frequency for generations    4 to  103
 0.8588 to 0.8819     *
 0.8819 to 0.9050     *
 0.9050 to 0.9281     *****
 0.9281 to 0.9512     ******
 0.9512 to 0.9743     *************
 0.9743 to 0.9974     ********************
 0.9974 to 1.0205     **************
 1.0205 to 1.0436     *************
 1.0436 to 1.0667     ************
 1.0667 to 1.0898     *******
 1.0898 to 1.1129     *****
 1.1129 to 1.1360     **
 1.1360 to 1.1590
 1.1590 to 1.1821     *
                                    frequency for generations   29 to  103
 0.8588 to 0.8819     *
 0.8819 to 0.9050     *
 0.9050 to 0.9281     ***
 0.9281 to 0.9512     ******
 0.9512 to 0.9743     ***********
 0.9743 to 0.9974     **************
 0.9974 to 1.0205     ***********
 1.0205 to 1.0436     **********
 1.0436 to 1.0667     *********
 1.0667 to 1.0898     ****
 1.0898 to 1.1129     **
 1.1129 to 1.1360     **
 1.1360 to 1.1590
 1.1590 to 1.1821     *
                                    frequency for generations   54 to  103
 0.8588 to 0.8819
 0.8819 to 0.9050
 0.9050 to 0.9281     ***
 0.9281 to 0.9512     ****
 0.9512 to 0.9743     *******
 0.9743 to 0.9974     ********
 0.9974 to 1.0205     ********
 1.0205 to 1.0436     *******
 1.0436 to 1.0667     *******
 1.0667 to 1.0898     ***
 1.0898 to 1.1129     **
 1.1129 to 1.1360     *
 1.1360 to 1.1590
 1.1590 to 1.1821
                                    frequency for generations   79 to  103
 0.8588 to 0.8819
 0.8819 to 0.9050
 0.9050 to 0.9281
 0.9281 to 0.9512     **
 0.9512 to 0.9743     ****
 0.9743 to 0.9974     *****
 0.9974 to 1.0205     ****
 1.0205 to 1.0436     *****
 1.0436 to 1.0667     **
 1.0667 to 1.0898     **
 1.0898 to 1.1129     *
 1.1129 to 1.1360
 1.1360 to 1.1590
 1.1590 to 1.1821

***********************************************************************************************************************************************************
           congratulations!  you have  successfully traversed the perilous path through keno v in   1.10933 minutes
***********************************************************************************************************************************************************
```

```
                kk          kk  eeeeeeeeeeeee   nn          nn   ooooooooooo                    vv          vv
                kk         kk   eeeeeeeeeeeee   nnn         nn  ooooooooooooo                   vv          vv
                kk        kk    ee              nnnn        nn  oo         oo                   vv          vv
                kk       kk     ee              nn nn       nn  oo         oo                   vv          vv
                kk      kk      ee              nn   nn     nn  oo         oo                   vv          vv
                kkkkkkkk        eeeeeeeee       nn    nn    nn  oo         oo   -------------   vv          vv
                kkkkkkkk        eeeeeeeee       nn     nn   nn  oo         oo   -------------    vv        vv
                kk      kk      ee              nn      nn  nn  oo         oo                      vv     vv
                kk       kk     ee              nn       nn nn  oo         oo                       vv   vv
                kk        kk    ee              nn        nnnn  oo         oo                        vv vv
                kk         kk   eeeeeeeeeeeee   nn         nnn  ooooooooooooo                         vvv
                kk          kk  eeeeeeeeeeeee   nn          nn   ooooooooooo                           v

xx          xx           44      sssssssssss
 xx        xx           444     sssssssssssss
   xx     xx           4444     ss          ss
    xx   xx           44 44     ss
     xx xx           44  44     ss
      xxx           44   44     ssssssssssss
      xxx          44    44      ssssssssssss
     xx xx       444444444444               ss
    xx   xx     4444444444444               ss
   xx     xx             44     ss          ss
 xx        xx            44     sssssssssssss
xx           xx          44      sssssssssss

   0000000       99999999999                //     0000000      7777777777777               //   99999999999    5555555555555
  000000000     9999999999999              //     000000000     777777777777               //   9999999999999   5555555555555
 00       00    99         99             //     00       00    77        77              //    99         99   55
00         00   99         99            //     00         00            77              //     99         99   55
00         00   99         99           //      00         00           77              //      99         99   55
00         00   9999999999999          //       00         00          77              //       9999999999999   555555555555
00         00    99999999999          //        00         00          77             //         99999999999    5555555555555
00         00              99        //         00         00         77             //                    99              55
00         00              99       //          00         00         77            //                     99              55
 00       00               99      //            00       00          77            //                      99   55         55
  000000000     9999999999999     //               000000000          77           //             9999999999999   5555555555555
   0000000      999999999999     //                 0000000           77          //              999999999999     55555555555

   0000000               44                       33333333333       0000000                            0000000         0000000
  000000000            444                      3333333333333     000000000                          000000000       000000000
 00       00          4444          :::         33         33    00       00          :::          00       00      00       00
00         00         44 44         :::                    33   00         00         :::         00         00    00         00
00         00        44  44         :::                    33   00         00         :::         00         00    00         00
00         00       44   44                              333   00         00                     00         00    00         00
00         00      44    44                              333   00         00                     00         00    00         00
00         00    444444444444         :::                    33   00         00         :::         00         00    00         00
00         00   4444444444444         :::                    33   00         00         :::         00         00    00         00
 00       00              44          :::         33         33    00       00          :::          00       00      00       00
  000000000              44                     3333333333333     000000000                          000000000       000000000
   0000000               44                       33333333333       0000000                            0000000         0000000
 sssssssssss     ccccccccccc      aaaaaaaaa     ll              eeeeeeeeeeeee
sssssssssssss   ccccccccccccc    aaaaaaaaaaa    ll              eeeeeeeeeeeee
ss         ss   cc         cc   aa         aa   ll              ee
ss              cc              aa         aa   ll              ee
ss              cc              aa         aa   ll              ee
ssssssssssss    cc              aaaaaaaaaaaaa   ll              eeeeeeeee
 ssssssssssss   cc              aaaaaaaaaaaaa   ll              eeeeeeeee
           ss   cc              aa         aa   ll              ee
           ss   cc              aa         aa   ll              ee
ss         ss   cc         cc   aa         aa   ll              ee
sssssssssssss   ccccccccccccc   aa         aa   lllllllllllll   eeeeeeeeeeeee
 sssssssssss     ccccccccccc    aa         aa   lllllllllllll   eeeeeeeeeeeee
```

```
********************************************************************************
********************************************************************************
********************************************************************************
*****                                                                      *****
*****                  program verification information                    *****
*****                                                                      *****
*****                 code system:    scale  version:    4.3               *****
*****                                                                      *****
********************************************************************************
********************************************************************************
*****                                                                      *****
*****                                                                      *****
*****              program:  kenova                                        *****
*****                                                                      *****
*****        creation date:  08/13/96                                      *****
*****                                                                      *****
*****              library:  /scale4.3/bin                                 *****
*****                                                                      *****
*****                                                                      *****
*****      production code:  kenova                                        *****
*****                                                                      *****
*****              version:  3.7                                           *****
*****                                                                      *****
*****              jobname:  x4s                                           *****
*****                                                                      *****
*****    date of execution:  09/07/95                                      *****
*****                                                                      *****
*****    time of execution:  04:30:00                                      *****
*****                                                                      *****
*****                                                                      *****
********************************************************************************
********************************************************************************
********************************************************************************
```

```
************************************************************************************************
***                                                                                          ***
***                 sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)       ***
***                                                                                          ***
************************************************************************************************
***                              ******       numeric parameters       ******                ***
***                                                                                          ***
***                                                                                          ***
***               tme       maximum problem time (min)                        120.00         ***
***                                                                                          ***
***               tba       time per generation (min)                           0.50         ***
***                                                                                          ***
***               gen       number of generations                                103         ***
***                                                                                          ***
***               npg       number per generation                                300         ***
***                                                                                          ***
***               nsk       number of generations to be skipped                    3         ***
***                                                                                          ***
***               beg       beginning generation number                            1         ***
***                                                                                          ***
***               res       generations between checkpoints                        0         ***
***                                                                                          ***
***               x1d       number of extra 1-d cross sections                     1         ***
***                                                                                          ***
***               nbk       neutron bank size                                    325         ***
***                                                                                          ***
***               xnb       extra positions in neutron bank                        0         ***
***                                                                                          ***
***               nfb       fission bank size                                    300         ***
***                                                                                          ***
***               xfb       extra positions in fission bank                        0         ***
***                                                                                          ***
***               wta       default value of weight average                   0.5000         ***
***                                                                                          ***
***               wth       weight high for splitting                         3.0000         ***
***                                                                                          ***
***               wtl       weight low for russian roulette                   0.3333         ***
***                                                                                          ***
***               rnd       starting random number                       bb827100001         ***
***                                                                                          ***
***               nb8       number of d.a. blocks on unit  8                     200         ***
***                                                                                          ***
***               nl8       length of d.a. blocks on unit  8                     512         ***
***                                                                                          ***
***               adj       mode of calculation                              forward         ***
***                                                                                          ***
***                         input data written on restart unit                    no         ***
***                                                                                          ***
***                         binary data interface                                 no         ***
***                                                                                          ***
***                                                                                          ***
************************************************************************************************
************************************************************************************************
```

```
*********************************************************************************************************************
*********************************************************************************************************************
***                                                                                                               ***
***                              sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)               ***
***                                                                                                               ***
*********************************************************************************************************************
***                                 ******      logical parameters        ******                                  ***
***                                                                                                               ***
***  run  execute problem after checking data    yes            plt  plot picture map(s)                      yes ***
***                                                                                                               ***
***  flx  compute flux                           yes            fdn  compute fission densities                yes ***
***                                                                                                               ***
***  smu  compute avg unit self-multiplication   yes            nub  compute nu-bar & avg fission group       yes ***
***                                                                                                               ***
***  mku  compute matrix k-eff by unit number    yes            mkp  compute matrix k-eff by unit location    yes ***
***                                                                                                               ***
***  cku  compute cofactor k-eff by unit number   no            ckp  compute cofactor k-eff by unit location   no ***
***                                                                                                               ***
***  fmu  print fiss prod matrix by unit number  yes            fmp  print fiss prod matrix by unit location yes ***
***                                                                                                               ***
***  mkh  compute matrix k-eff by hole number     no            mka  compute matrix k-eff by array number      no ***
***                                                                                                               ***
***  ckh  compute cofactor k-eff by hole number   no            cka  compute cofactor k-eff by array number    no ***
***                                                                                                               ***
***  fmh  print fiss prod matrix by hole number   no            fma  print fiss prod matrix by array number    no ***
***                                                                                                               ***
***  hhl  collect matrix by highest hole level    no            hal  collect matrix by highest array level     no ***
***                                                                                                               ***
***  amx  print all mixed cross sections          no            far  print fis. and abs. by region             no ***
***                                                                                                               ***
***  xs1  print 1-d mixture x-sections            no            gas  print far by group                        no ***
***                                                                                                               ***
***  xs2  print 2-d mixture x-sections            no            pax  print xsec-albedo correlation tables      no ***
***                                                                                                               ***
***  xap  print mixture angles & probabilities    no            pwt  print weight average array                no ***
***                                                                                                               ***
***  pki  print fission spectrum                  no            pgm  print input geometry                      no ***
***                                                                                                               ***
***  p1d  print extra 1-d cross sections          no            bug  print debug information                   no ***
***                                                                                                               ***
***                                                             trk  print tracking information                no ***
***                                                                                                               ***
***                                                                                                               ***
*********************************************************************************************************************
*********************************************************************************************************************
*********************************************************************************************************************
                                      parameter input completed

                        ........    0 io's were used reading the parameter data      ........

                       *************** data reading completed ***************
```

```
**********************************************************************************************************************
***                                                                                                                ***
***                              sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)                ***
***                                                                                                                ***
**********************************************************************************************************************
**********************************************************************************************************************
***                                                                                                                ***
***          unit                                                 volume                                           ***
***         number               data set name                     name        unit function                       ***
***         ------               -------------                     ----        -------------                       ***
***                                                                                                                ***
***      xsc  14     ft14f001                                                  mixed cross sections                ***
***                                                                                                                ***
***      alb  79     /scale4.3/data/albedos                                    input albedos                       ***
***                                                                                                                ***
***      wts  80     /scale4.3/data/weights                                    input weights                       ***
***                                                                                                                ***
***      skt  16      unknown                                                  write scratch data                  ***
***                                                                                                                ***
***      lib   4     ft04f001                                                  input ampx working library          ***
***                                                                                                                ***
***            8     ft08f001                                                  input data direct access            ***
***                                                                                                                ***
***            9      unknown                                                  super grouped direct access         ***
***                                                                                                                ***
***           10      unknown                                                  xsec mixing direct access           ***
***                                                                                                                ***
**********************************************************************************************************************
                        ........       0 io's were used preparing input data     ........

                         cross sections read from the ampx working library on unit    4
```

```
                                    sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

                                                  mixing table

                                         number of scattering angles =  2
                                     cross section message threshold =3.0E-05


 mixture =      1          density(g/cc) =  18.760
    nuclide   atom-dens.   wgt. frac.     za      awt              nuclide title
    1092234  4.82716e-04  9.99999e-03   92234   234.0405      uranium-234    endf/b-iv mat 1043                         updated 08/12/94
    1092235  4.47971e-02  9.32000e-01   92235   235.0441      uranium-235    endf/b-iv mat 1261                         updated 08/12/94
    1092236  9.57231e-05  2.00000e-03   92236   236.0458     u-236 1163 sigo=5+4 newxlacs p-3 293k f-1/e-m(1.+5)      updated 08/12/94
    1092238  2.65767e-03  5.60000e-02   92238   238.0510      uranium-238    endf/b-iv mat 1262                         updated 08/12/94

 mixture =      2          density(g/cc) =  1.5547
    nuclide   atom-dens.   wgt. frac.     za      awt              nuclide title
    2001001  5.77964e-02  6.22048e-02    1001     1.0077      hydrogen       endf/b-iv mat 1269/thrm1002                updated 08/12/94
    2007014  2.13092e-03  3.18719e-02    7014    14.0033      nitrogen-14    endf/b-iv mat 1275                         updated 08/12/94
    2008016  3.74130e-02  6.38986e-01    8016    15.9904      oxygen-16      endf/b-iv mat 1276                         updated 08/12/94
    2092234  1.06784e-05  2.66936e-03   92234   234.0405      uranium-234    endf/b-iv mat 1043                         updated 08/12/94
    2092235  9.84599e-04  2.47184e-01   92235   235.0441      uranium-235    endf/b-iv mat 1261                         updated 08/12/94
    2092236  5.29385e-06  1.33468e-03   92236   236.0458     u-236 1163 sigo=5+4 newxlacs p-3 293k f-1/e-m(1.+5)      updated 08/12/94
    2092238  6.19413e-05  1.57493e-02   92238   238.0510      uranium-238    endf/b-iv mat 1262                         updated 08/12/94

 mixture =      3          density(g/cc) =  1.1799
    nuclide   atom-dens.   wgt. frac.     za      awt              nuclide title
   11001001  5.68187e-02  8.05783e-02    1001     1.0077      hydrogen       endf/b-iv mat 1269/thrm1002                updated 08/12/94
   11006012  3.55117e-02  5.99750e-01    6000    12.0001      carbon-12      endf/b-iv mat 1274/thrm1065                updated 08/12/94
   11008016  1.42047e-02  3.19672e-01    8016    15.9904      oxygen-16      endf/b-iv mat 1276                         updated 08/12/94

                             2001001     hydrogen       endf/b-iv mat 1269/thrm1002           updated 08/12/94
                            11001001     hydrogen       endf/b-iv mat 1269/thrm1002           updated 08/12/94
                            11006012     carbon-12      endf/b-iv mat 1274/thrm1065           updated 08/12/94
                             2007014     nitrogen-14    endf/b-iv mat 1275                    updated 08/12/94
                             2008016     oxygen-16      endf/b-iv mat 1276                    updated 08/12/94
                            11008016     oxygen-16      endf/b-iv mat 1276                    updated 08/12/94
                             1092234     uranium-234    endf/b-iv mat 1043                    updated 08/12/94
                             2092234     uranium-234    endf/b-iv mat 1043                    updated 08/12/94
                             1092235     uranium-235    endf/b-iv mat 1261                    updated 08/12/94
                             2092235     uranium-235    endf/b-iv mat 1261                    updated 08/12/94
                             1092236    u-236 1163 sigo=5+4 newxlacs p-3 293k f-1/e-m(1.+5)   updated 08/12/94
                             2092236    u-236 1163 sigo=5+4 newxlacs p-3 293k f-1/e-m(1.+5)   updated 08/12/94
                             1092238     uranium-238    endf/b-iv mat 1262                    updated 08/12/94
                             2092238     uranium-238    endf/b-iv mat 1262                    updated 08/12/94

 keno message number k5-222       2 transfers for mixture     2 were corrected for bad moments.

 keno message number k5-222       3 transfers for mixture     3 were corrected for bad moments.
                              ........    0 io's were used mixing cross-sections       ........
                               1-d cross section array id numbers
                                  1  2002  1452    27    18  1018
                              ........    0 io's were used preparing the cross sections    ........
```

```
********************************************************************************************************
***                                                                                                  ***
***            sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)                    ***
***                                                                                                  ***
********************************************************************************************************
********************************************************************************************************
***                                                                                                  ***
***                             ****** additional information ******                                 ***
***                                                                                                  ***
***  number of energy groups              27       use lattice geometry                       yes   ***
***                                                                                                  ***
***  no. of fission spectrum source group  1       global array number                          3   ***
***                                                                                                  ***
***  no. of scattering angles in xsecs     2       number of units in the global x dir.         2   ***
***                                                                                                  ***
***  entries/neutron in the neutron bank  24       number of units in the global y dir.         1   ***
***                                                                                                  ***
***  entries/neutron in the fission bank  14       number of units in the global z dir.         1   ***
***                                                                                                  ***
***  number of mixtures used               3       use a global reflector                      no   ***
***                                                                                                  ***
***  number of bias id's used              1       use nested holes                            no   ***
***                                                                                                  ***
***  number of differential albedos used   0       number of holes                              0   ***
***                                                                                                  ***
***  total input geometry regions          8       maximum hole nesting level                   0   ***
***                                                                                                  ***
***  number of geometry regions used       8       use nested arrays                          yes   ***
***                                                                                                  ***
***  largest geometry unit number          4       number of arrays used                        3   ***
***                                                                                                  ***
***  largest array number                  3       maximum array nesting level                  2   ***
***                                                                                                  ***
***                                                                                                  ***
***  +x boundary condition            vacuum       -x boundary condition                   vacuum   ***
***                                                                                                  ***
***  +y boundary condition            vacuum       -y boundary condition                   vacuum   ***
***                                                                                                  ***
***  +z boundary condition            vacuum       -z boundary condition                   vacuum   ***
***                                                                                                  ***
********************************************************************************************************
```

```
***********************************************************************************************
***                                                                                         ***
***          sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)             ***
***                                                                                         ***
***********************************************************************************************
***********************************************************************************************
***                                                                                         ***
***                    ****** space and supergroup information ******                       ***
***                                                                                         ***
***        40000 words is the total space available.                                        ***
***                                                                                         ***
***        13247 words were used for non-supergroup storage.                                ***
***                                                                                         ***
***        26753 words of storage are available for supergrouped data.                      ***
***                                                                                         ***
***        39870 words of storage are available for constructing the supergroups.           ***
***                                                                                         ***
***        26693 words of storage are available to each supergroup.                         ***
***                                                                                         ***
***          474 words are needed for the largest group.                                    ***
***                                                                                         ***
***        13937 words of storage is sufficient to run this problem.                        ***
***                                                                                         ***
***        19647 words of storage will allow the problem to run with one supergroup.        ***
***                                                                                         ***
***        20032 words of storage will be used to run this problem.                         ***
***                                                                                         ***
***********************************************************************************************
***********************************************************************************************
***                                                                                         ***
***                   starting      ending       xsec       albedo      total               ***
***   supergroup        group        group      length      length      length              ***
***                                                                                         ***
***                                                                                         ***
***          1             1           27         1099           0         6340             ***
***                                                                                         ***
***********************************************************************************************
             ........    0 io’s were used in supergrouping     ........

            ************************************************************
            **                                                        **
            **   array      units in   units in   units in   nesting  **
            **   number      x dir.     y dir.     z dir.     level   **
            **                                                        **
            **     1            1          2          2          2    **
            **                                                        **
            **     2            1          2          2          2    **
            **                                                        **
            **     3 global     2          1          1          1    **
            **                                                        **
            ************************************************************

             ........    0 io’s were used loading the data     ........
```

```
                                          sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

                           media bias      geometry description for those units utilized in this problem
region                      num   id

                                                                    -----   unit     1   -----

uranyl nitrate solution in a plexiglas container

  1 cylinder             2  1  radius =  9.5250     +z =  8.8900     -z = -8.8900     centerline is at  x =  0.0000      y =  0.0000

  2 cylinder             3  1  radius =  10.160     +z =  9.5250     -z = -9.5250     centerline is at  x =  0.0000      y =  0.0000

  3 cuboid               0  1      +x =  10.875     -x = -10.875     +y =  10.875     -y = -10.875     +z =  10.240     -z = -10.240

                                                                    -----   unit     2   -----

uranium metal cylinder

  1 cylinder             1  1  radius =  5.7480     +z =  5.3825     -z = -5.3825     centerline is at  x =  0.0000      y =  0.0000

  2 cuboid               0  1      +x =  6.5900     -x = -6.5900     +y =  6.5900     -y = -6.5900     +z =  6.2250     -z = -6.2250

                                                        -----   unit     3  external to lattice  1   -----

1x2x2 array of solution units

  1 array number         1         +x =  21.750     -x =  0.0000     +y =  43.500     -y =  0.0000     +z =  40.960     -z =  0.0000

                                                        -----   unit     4  external to lattice  2   -----

1x2x2 array of metal units padded to match solution array

  1 array number         2         +x =  13.180     -x =  0.0000     +y =  26.360     -y =  0.0000     +z =  24.900     -z =  0.0000

  2 cuboid               0  1      +x =  13.180     -x =  0.0000     +y =  34.930     -y = -8.5700     +z =  32.930     -z = -8.0300
```

```
                       sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)
                   -------  unit orientation description for array   1         -------
  z layer   1, x column   1 to   1 left to right   y row   1 to   2  bottom to top

   1

   1
 -z layer   2, x column   1 to   1 left to right   y row   1 to   2  bottom to top

   1

   1
                   -------  unit orientation description for array   2         -------
  z layer   1, x column   1 to   1 left to right   y row   1 to   2  bottom to top

   2

   2
 -z layer   2, x column   1 to   1 left to right   y row   1 to   2  bottom to top

   2

   2
                   -------  unit orientation description for array   3         -------
  composite array of solution and metal units
  z layer   1, x column   1 to   2 left to right   y row   1 to   1  bottom to top

   4 3
```

```
                     sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)
                    volumes for those units utilized in  this problem
                      geometry                                 cumulative
  unit     region      region          volume                   volume

   1          1          1          5.06771e+03 cm**3         5.06771e+03 cm**3
              2          2          1.11007e+03 cm**3         6.17778e+03 cm**3
              3          3          3.51054e+03 cm**3         9.68832e+03 cm**3

   2          1          4          1.11737e+03 cm**3         1.11737e+03 cm**3
              2          5          1.04535e+03 cm**3         2.16272e+03 cm**3

  surrounding geometry volumes -   geometry region      6 is an array placement boundary region

   3          1          6          3.87533e+04 cm**3         3.87533e+04 cm**3

  surrounding geometry volumes -   geometry region      7 is an array placement boundary region

   4          1          7          8.65088e+03 cm**3         8.65088e+03 cm**3
              2          8          1.48327e+04 cm**3         2.34836e+04 cm**3

                    unit      uses     region     mixture        total volume

                      1         4         1          2         2.02708e+04 cm**3
                                          2          3         4.44028e+03 cm**3
                                          3          0         1.40422e+04 cm**3

                      2         4         1          1         4.46948e+03 cm**3
                                          2          0         4.18139e+03 cm**3

                      3         1         1                    3.87533e+04 cm**3

                      4         1         1                    8.65088e+03 cm**3
                                          2          0         1.48327e+04 cm**3

                                   total mixture volumes
                               mixture        total volume              mass(g)
                                  0         3.30563e+04 cm**3
                                  1         4.46948e+03 cm**3         8.38475e+04
                                  2         2.02708e+04 cm**3         3.15145e+04
                                  3         4.44028e+03 cm**3         5.23896e+03
      ***********************************************************************************************************************
      ***                                      biasing information                                                          ***
      ***                                                                                                                   ***
      ***     a default weight of    0.500 will be used for all bias id's.                                                  ***
      ***********************************************************************************************************************
                            ........    0 io's were used in keno-v before tracking       ........
                            ........  0.00000 minutes were used processing data.       ........
volume fraction of fissile material in the core= 3.97519e-01
start type 0 was used.
the neutrons were started with a flat distribution in a cuboid defined by:
                       +x= 3.49300e+01  -x= 0.00000e+00  +y= 4.35000e+01  -y= 0.00000e+00  +z= 4.09600e+01  -z= 0.00000e+00
the flag to start neutrons in the reflector was turned off
0.00000 minutes were required for starting.   total elapsed time is 0.00000 minutes.
```

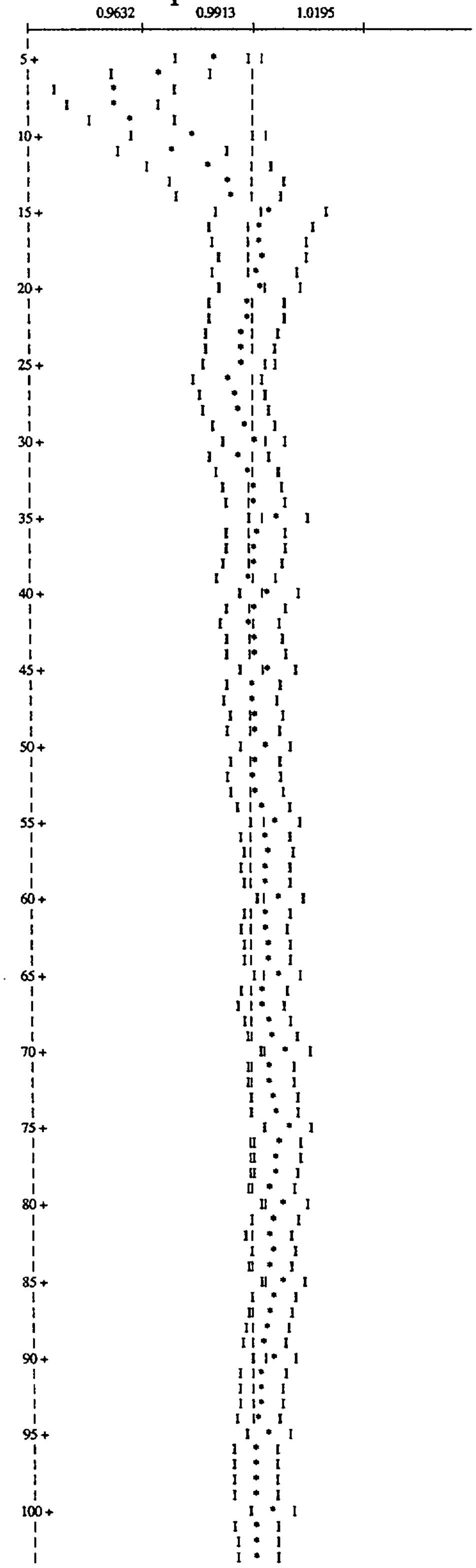
```
sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

                  generation      elapsed time        average         avg k-eff           matrix        matrix k-eff
    generation   k-effective         minutes        k-effective       deviation        k-effective       deviation
keno message number k5-132      warning....only  284 independent fission points were generated
          1      7.96943e-01       2.13333e-02      1.00000e+00      0.00000e+00       8.88898e-01      0.00000e+00
          2      1.05306e+00       2.13333e-02      1.00000e+00      0.00000e+00       9.80205e-01      6.65921e-02
          3      9.66360e-01       2.13333e-02      9.66360e-01      0.00000e+00       9.80786e-01      4.97743e-02
          4      9.82821e-01       2.13333e-02      9.74590e-01      8.23039e-03       9.70505e-01      4.41498e-02
          5      1.00681e+00       2.13333e-02      9.85329e-01      1.17436e-02       9.76325e-01      3.72217e-02
          6      9.49767e-01       4.26667e-02      9.76439e-01      1.21656e-02       9.75191e-01      4.00267e-02
          7      9.14772e-01       4.26667e-02      9.64105e-01      1.55213e-02       9.67834e-01      3.82051e-02
          8      9.61786e-01       4.26667e-02      9.63719e-01      1.26790e-02       9.73327e-01      4.08133e-02
          9      1.00263e+00       4.26667e-02      9.69277e-01      1.20715e-02       9.77179e-01      4.15454e-02
         10      1.08623e+00       4.26667e-02      9.83896e-01      1.79724e-02       9.87432e-01      3.88200e-02
         11      9.67810e-01       4.26667e-02      9.82109e-01      1.59506e-02       9.87425e-01      3.71531e-02
         12      1.09931e+00       4.26667e-02      9.93829e-01      1.84636e-02       9.95126e-01      3.43877e-02
         13      1.05620e+00       4.26667e-02      9.99499e-01      1.76371e-02       1.00098e+00      3.22140e-02
         14      1.01521e+00       4.26667e-02      1.00081e+00      1.61536e-02       1.00314e+00      3.14800e-02
         15      1.11939e+00       4.26667e-02      1.00993e+00      1.74355e-02       1.01081e+00      3.14877e-02
         16      1.02942e+00       6.40000e-02      1.01132e+00      1.62020e-02       1.01008e+00      3.00795e-02
         17      1.00884e+00       6.40000e-02      1.01116e+00      1.50842e-02       1.00862e+00      2.83707e-02
         18      1.03041e+00       6.40000e-02      1.01236e+00      1.41611e-02       1.01275e+00      2.78280e-02
         19      9.75326e-01       6.40000e-02      1.01018e+00      1.34793e-02       1.01197e+00      2.72243e-02
         20      9.56316e-01       6.40000e-02      1.00719e+00      1.30559e-02       1.00787e+00      2.68409e-02
         21      1.01472e+00       6.40000e-02      1.00758e+00      1.23560e-02       1.00745e+00      2.59711e-02
         22      1.00633e+00       8.53333e-02      1.00752e+00      1.17221e-02       1.00778e+00      2.51483e-02
         23      9.75637e-01       8.53333e-02      1.00600e+00      1.12529e-02       1.00632e+00      2.46280e-02
         24      1.00407e+00       8.53333e-02      1.00592e+00      1.07296e-02       1.00474e+00      2.35726e-02
         25      8.95448e-01       8.53333e-02      1.00111e+00      1.13217e-02       1.00032e+00      2.27365e-02
         26      1.00285e+00       8.53333e-02      1.00119e+00      1.08400e-02       1.00050e+00      2.20429e-02
         27      1.05487e+00       1.06667e-01      1.00333e+00      1.06168e-02       1.00264e+00      2.14294e-02
         28      1.03669e+00       1.06667e-01      1.00462e+00      1.02806e-02       1.00422e+00      2.09508e-02
         29      1.06917e+00       1.06667e-01      1.00701e+00      1.01773e-02       1.00635e+00      2.07288e-02
         30      9.95230e-01       1.06667e-01      1.00659e+00      9.81613e-03       1.00641e+00      2.02680e-02
         31      9.79844e-01       1.06667e-01      1.00566e+00      9.51638e-03       1.00539e+00      1.97305e-02
         32      1.09738e+00       1.06667e-01      1.00872e+00      9.68872e-03       1.00798e+00      1.91765e-02
         33      1.05896e+00       1.06667e-01      1.01034e+00      9.51005e-03       1.00916e+00      1.91381e-02
         34      1.03390e+00       1.06667e-01      1.01108e+00      9.23744e-03       1.01084e+00      1.88450e-02
         35      1.11889e+00       1.06667e-01      1.01434e+00      9.53059e-03       1.01385e+00      1.84499e-02
         36      9.17474e-01       1.06667e-01      1.01150e+00      9.67506e-03       1.01188e+00      1.81234e-02
         37      1.00515e+00       1.06667e-01      1.01131e+00      9.39631e-03       1.01141e+00      1.76680e-02
         38      9.82617e-01       1.28000e-01      1.01052e+00      9.16630e-03       1.01059e+00      1.72682e-02
         39      9.30596e-01       1.28000e-01      1.00836e+00      9.17307e-03       1.00888e+00      1.69875e-02
         40      1.12493e+00       1.28000e-01      1.01142e+00      9.44068e-03       1.01189e+00      1.69169e-02
         41      1.00211e+00       1.28000e-01      1.01119e+00      9.19853e-03       1.01160e+00      1.66184e-02
         42      9.24284e-01       1.28000e-01      1.00901e+00      9.22509e-03       1.00954e+00      1.64050e-02
         43      1.09279e+00       1.28000e-01      1.01106e+00      9.22638e-03       1.01153e+00      1.60644e-02
         44      1.03087e+00       1.49333e-01      1.01153e+00      9.01637e-03       1.01181e+00      1.56956e-02
         45      9.98612e-01       1.49333e-01      1.01123e+00      8.80931e-03       1.01153e+00      1.53682e-02
         46      9.76368e-01       1.49333e-01      1.01044e+00      8.64316e-03       1.01042e+00      1.50986e-02
         47      9.79931e-01       1.49333e-01      1.00976e+00      8.47606e-03       1.00972e+00      1.50440e-02
         48      1.08248e+00       1.49333e-01      1.01134e+00      8.43913e-03       1.01079e+00      1.53048e-02
         49      9.84573e-01       1.70667e-01      1.01077e+00      8.27723e-03       1.01089e+00      1.52886e-02
         50      9.81265e-01       1.70667e-01      1.01015e+00      8.12624e-03       1.01050e+00      1.51324e-02
         51      1.06397e+00       1.70667e-01      1.01125e+00      8.03408e-03       1.01217e+00      1.49015e-02
         52      9.78029e-01       1.70667e-01      1.01059e+00      7.89976e-03       1.01161e+00      1.47339e-02
         53      1.07156e+00       1.70667e-01      1.01178e+00      7.83505e-03       1.01290e+00      1.45308e-02
         54      1.12695e+00       1.92000e-01      1.01400e+00      7.99577e-03       1.01502e+00      1.44571e-02
         55      9.95216e-01       1.92000e-01      1.01364e+00      7.85146e-03       1.01534e+00      1.44684e-02
         56      1.04803e+00       1.92000e-01      1.01428e+00      7.73096e-03       1.01559e+00      1.43923e-02
         57      1.11127e+00       1.92000e-01      1.01604e+00      7.79126e-03       1.01690e+00      1.41701e-02
         58        9.58752e-01      1.92000e-01      1.01502e+00      7.71897e-03      1.01602e+00      1.39779e-02
         59        1.02188e+00      1.92000e-01      1.01514e+00      7.58330e-03      1.01637e+00      1.38653e-02
         60        1.01163e+00      1.92000e-01      1.01508e+00      7.45165e-03      1.01641e+00      1.38970e-02
         61        1.00643e+00      1.92000e-01      1.01493e+00      7.32573e-03      1.01632e+00      1.39727e-02
         62        9.95579e-01      1.92000e-01      1.01461e+00      7.20982e-03      1.01628e+00      1.41202e-02
         63        1.05864e+00      1.92000e-01      1.01533e+00      7.12727e-03      1.01639e+00      1.41501e-02
         64        1.02762e+00      1.92000e-01      1.01553e+00      7.01418e-03      1.01639e+00      1.41861e-02
         65        9.61611e-01      1.92000e-01      1.01468e+00      6.95481e-03      1.01542e+00      1.41957e-02
         66        9.66308e-01      1.92000e-01      1.01392e+00      6.88687e-03      1.01480e+00      1.43344e-02
         67        9.83432e-01      2.13333e-01      1.01345e+00      6.79629e-03      1.01418e+00      1.44063e-02
         68        1.14918e+00      2.13333e-01      1.01551e+00      7.00135e-03      1.01621e+00      1.45296e-02
         69        1.09561e+00      2.13333e-01      1.01670e+00      6.99892e-03      1.01699e+00      1.44911e-02
         70        1.04543e+00      2.13333e-01      1.01713e+00      6.90816e-03      1.01731e+00      1.43321e-02
         71        9.68881e-01      2.13333e-01      1.01643e+00      6.84312e-03      1.01643e+00      1.42186e-02
         72        1.00639e+00      2.34667e-01      1.01628e+00      6.74617e-03      1.01665e+00      1.40810e-02
         73        1.09803e+00      2.34667e-01      1.01743e+00      6.74940e-03      1.01766e+00      1.39654e-02
         74        1.05952e+00      2.34667e-01      1.01802e+00      6.68061e-03      1.01806e+00      1.38287e-02
         75        1.01480e+00      2.34667e-01      1.01797e+00      6.58861e-03      1.01770e+00      1.36430e-02
         76        1.07554e+00      2.34667e-01      1.01875e+00      6.54536e-03      1.01867e+00      1.35147e-02
         77        1.01366e+00      2.56000e-01      1.01868e+00      6.45785e-03      1.01856e+00      1.33425e-02
         78        9.70921e-01      2.56000e-01      1.01806e+00      6.40323e-03      1.01805e+00      1.32009e-02
         79        8.75749e-01      2.56000e-01      1.01621e+00      6.58422e-03      1.01625e+00      1.30928e-02
         80        1.02836e+00      2.56000e-01      1.01636e+00      6.50113e-03      1.01624e+00      1.29282e-02
         81        1.10038e+00      2.56000e-01      1.01743e+00      6.50583e-03      1.01716e+00      1.28081e-02
 keno message number k5-132     warning....only  297 independent fission points were generated
         82        8.97266e-01      2.56000e-01      1.01593e+00      6.59725e-03      1.01571e+00      1.26929e-02
         83        1.10648e+00      2.56000e-01      1.01704e+00      6.61051e-03      1.01676e+00      1.25572e-02
         84        9.45938e-01      2.56000e-01      1.01618e+00      6.58673e-03      1.01607e+00      1.24185e-02
         85        1.04135e+00      2.56000e-01      1.01648e+00      6.51395e-03      1.01627e+00      1.22769e-02
         86        1.09585e+00      2.56000e-01      1.01742e+00      6.50494e-03      1.01708e+00      1.21886e-02
         87        9.39632e-01      2.56000e-01      1.01651e+00      6.49278e-03      1.01620e+00      1.20638e-02
         88        8.79969e-01      2.56000e-01      1.01492e+00      6.61033e-03      1.01498e+00      1.19835e-02
         89        9.37776e-01      2.77333e-01      1.01403e+00      6.59380e-03      1.01405e+00      1.18893e-02
         90        9.13217e-01      2.77333e-01      1.01289e+00      6.61836e-03      1.01289e+00      1.17993e-02
         91        1.04553e+00      2.77333e-01      1.01326e+00      6.55384e-03      1.01331e+00      1.17761e-02
         92        1.00236e+00      2.77333e-01      1.01313e+00      6.48174e-03      1.01310e+00      1.17507e-02
         93        9.56570e-01      2.77333e-01      1.01251e+00      6.44019e-03      1.01241e+00      1.16337e-02
         94        9.18702e-01      2.77333e-01      1.01149e+00      6.45090e-03      1.01134e+00      1.15374e-02
         95        9.54892e-01      2.98667e-01      1.01088e+00      6.41012e-03      1.01041e+00      1.14595e-02
         96        1.02399e+00      2.98667e-01      1.01102e+00      6.34309e-03      1.01068e+00      1.13534e-02
         97        9.86815e-01      2.98667e-01      1.01077e+00      6.28114e-03      1.01057e+00      1.13175e-02
         98        9.85306e-01      2.98667e-01      1.01050e+00      6.22102e-03      1.01011e+00      1.12725e-02
         99        9.97817e-01      2.98667e-01      1.01037e+00      6.15794e-03      1.00974e+00      1.12006e-02
        100        1.05886e+00      3.20000e-01      1.01087e+00      6.11483e-03      1.01016e+00      1.10930e-02
        101        9.83692e-01      3.20000e-01      1.01059e+00      6.05897e-03      1.00983e+00      1.09966e-02
        102        1.06340e+00      3.20000e-01      1.01112e+00      6.02128e-03      1.01049e+00      1.09003e-02
        103        1.01257e+00      3.20000e-01      1.01114e+00      5.96138e-03      1.01044e+00      1.07989e-02

keno message number k5-123          execution terminated due to completion of the specified number of generations.

  the matrix k-effective is the largest eigenvalue of the fission production by position index matrix.
 there are nbxmax * nbymax * nbzmax positions in an array.
```

```
                              sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

 lifetime   =  3.57030e-06 + or -  9.05739e-08            generation time =  5.20918e-06 + or -  1.14556e-07
  nu bar    =  2.50739e+00 + or -  1.51263e-03       average fission group =  1.39679e+01 + or -  1.05840e-01
                          energy(ev) of the average lethargy causing fission =  2.18921e+02 + or -  2.09930e+01
                                                    self multiplication =  6.51139e-01 + or -  4.91823e-03

  no. of initial
   generations       average                          67 per cent          95 per cent          99 per cent       number of
     skipped       k-effective      deviation      confidence interval  confidence interval  confidence interval   histories
        3          1.01158    + or - 0.00630     1.00529 to 1.01788   0.99899 to 1.02418   0.99269 to 1.03048      30000
        4          1.01187    + or - 0.00606     1.00582 to 1.01793   0.99976 to 1.02399   0.99370 to 1.03005      29700
        5          1.01193    + or - 0.00653     1.00540 to 1.01845   0.99887 to 1.02498   0.99234 to 1.03151      29400
        6          1.01257    + or - 0.00619     1.00638 to 1.01876   1.00019 to 1.02495   0.99400 to 1.03114      29100
        7          1.01359    + or - 0.00633     1.00725 to 1.01992   1.00092 to 1.02625   0.99459 to 1.03258      28800
        8          1.01413    + or - 0.00634     1.00779 to 1.02047   1.00145 to 1.02681   0.99511 to 1.03315      28500
        9          1.01425    + or - 0.00643     1.00782 to 1.02068   1.00139 to 1.02711   0.99496 to 1.03355      28200
       10          1.01348    + or - 0.00625     1.00723 to 1.01973   1.00098 to 1.02598   0.99472 to 1.03223      27900
       11          1.01398    + or - 0.00657     1.00741 to 1.02054   1.00084 to 1.02711   0.99428 to 1.03367      27600
       12          1.01304    + or - 0.00630     1.00674 to 1.01934   1.00044 to 1.02564   0.99414 to 1.03193      27300
       17          1.01113    + or - 0.00674     1.00440 to 1.01787   0.99766 to 1.02461   0.99092 to 1.03134      25800
       22          1.01203    + or - 0.00688     1.00515 to 1.01890   0.99828 to 1.02578   0.99140 to 1.03266      24300
       27          1.01370    + or - 0.00738     1.00633 to 1.02108   0.99895 to 1.02846   0.99158 to 1.03583      22800
       32          1.01216    + or - 0.00758     1.00457 to 1.01974   0.99699 to 1.02732   0.98941 to 1.03490      21300
       37          1.01104    + or - 0.00774     1.00331 to 1.01878   0.99557 to 1.02651   0.98783 to 1.03425      19800
       42          1.01253    + or - 0.00825     1.00427 to 1.02078   0.99602 to 1.02904   0.98777 to 1.03729      18300
       47          1.01224    + or - 0.00866     1.00359 to 1.02090   0.99493 to 1.02956   0.98628 to 1.03821      16800
       52          1.01167    + or - 0.00963     1.00204 to 1.02131   0.99241 to 1.03094   0.98278 to 1.04057      15300
       57          1.00527    + or - 0.00984     0.99542 to 1.01511   0.98558 to 1.02496   0.97574 to 1.03480      13800
       62          1.00605    + or - 0.01105     0.99500 to 1.01710   0.98395 to 1.02815   0.97290 to 1.03920      12300
       67          1.00696    + or - 0.01223     0.99473 to 1.01919   0.98249 to 1.03142   0.97026 to 1.04365      10800
       72          0.99951    + or - 0.01264     0.98688 to 1.01215   0.97424 to 1.02479   0.96161 to 1.03742       9300
       77          0.98936    + or - 0.01440     0.97496 to 1.00377   0.96055 to 1.01817   0.94615 to 1.03257       7800
       82          0.99289    + or - 0.01575     0.97714 to 1.00865   0.96139 to 1.02440   0.94563 to 1.04015       6300
       87          0.98259    + or - 0.01973     0.96286 to 1.00232   0.94314 to 1.02205   0.92341 to 1.04178       4800
       92          0.99478    + or - 0.02085     0.97394 to 1.01563   0.95309 to 1.03648   0.93224 to 1.05733       3300
       97          1.01694    + or - 0.01461     1.00233 to 1.03156   0.98771 to 1.04617   0.97310 to 1.06078       1800
```

sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

plot of average k-effective by generation run.

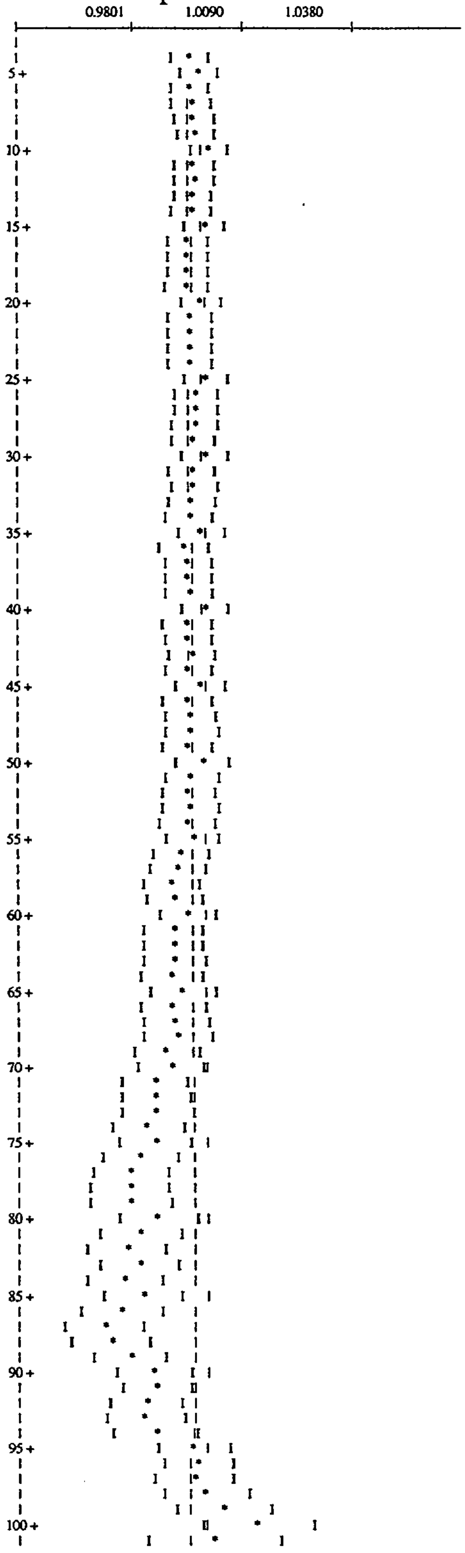
the line represents k-eff = 1.0116 + or - 0.0063 which occurs for 103 generations run.



sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

plot of average k-effective by generation skipped.

the line represents k-eff = 1.0119 + or - 0.0061 which occurs for 4 generations skipped.



k-effective satisfies the chi**2 test for normality at the 95 % level

```
                        sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)
                                                                                    skipping  3 generations

 group  fission   unit    region     fissions      percent      absorptions    percent        leakage       percent
        fraction                                  deviation                   deviation                    deviation

   1     0.0122                   1.23622e-02      7.4935      3.92297e-03      6.9050      8.67153e-03      5.9725
   2     0.0634                   6.41315e-02      2.3700      2.35268e-02      2.2212      6.06877e-02      2.3630
   3     0.0769                   7.77707e-02      2.1070      3.00449e-02      2.0886      6.88189e-02      2.0430
   4     0.0502                   5.08048e-02      2.7608      2.06237e-02      2.7459      4.15151e-02      2.8795
   5     0.0728                   7.36105e-02      2.1071      3.13569e-02      2.1044      6.11337e-02      2.2217
   6     0.0978                   9.89699e-02      1.8232      4.51986e-02      1.8192      8.93912e-02      2.0403
   7     0.0758                   7.66726e-02      2.1024      3.80573e-02      2.1019      6.58386e-02      1.8587
   8     0.0226                   2.28807e-02      4.1007      1.27902e-02      4.0965      2.49189e-02      3.5134
   9     0.0097                   9.82067e-03      5.6824      5.65481e-03      5.6573      1.64170e-02      4.1361
  10     0.0143                   1.44321e-02      3.8462      8.47419e-03      3.8152      1.27335e-02      4.6349
  11     0.0213                   2.15464e-02      2.3102      1.35707e-02      2.2696      1.20764e-02      5.0950
  12     0.0232                   2.34263e-02      2.0685      1.54414e-02      2.0091      7.63928e-03      6.4666
  13     0.0214                   2.16238e-02      2.2664      1.60679e-02      2.1283      6.66834e-03      5.6328
  14     0.0177                   1.79218e-02      2.2315      1.52000e-02      1.9068      6.31946e-03      7.1374
  15     0.0044                   4.42994e-03      5.3005      3.13526e-03      4.9170      2.41451e-03     10.5608
  16     0.0026                   2.61669e-03      6.3035      1.55246e-03      5.6759      1.18403e-03     17.9407
  17     0.0037                   3.69474e-03      4.5650      2.13203e-03      4.4767      7.44697e-04     17.6343
  18     0.0047                   4.75234e-03      4.2690      2.43910e-03      4.2011      5.04459e-04     20.4663
  19     0.0058                   5.81979e-03      3.8694      2.84695e-03      3.7318      1.02139e-03     16.2079
  20     0.0235                   2.37287e-02      2.5137      1.16065e-02      2.4664      3.11033e-03      8.7438
  21     0.0122                   1.23851e-02      2.7884      6.34937e-03      2.7656      9.43159e-04     15.9374
  22     0.0263                   2.66053e-02      2.0580      1.41340e-02      2.0471      1.42038e-03     13.7368
  23     0.0705                   7.12888e-02      1.5150      3.68100e-02      1.5052      3.86403e-03      8.5388
  24     0.0957                   9.67836e-02      1.6632      4.89143e-02      1.6573      3.89012e-03      7.3503
  25     0.0691                   6.98929e-02      1.9245      3.52126e-02      1.9161      1.78273e-03     10.7594
  26     0.0762                   7.70916e-02      1.7259      3.88872e-02      1.7184      1.34664e-03     10.7559
  27     0.0262                   2.65203e-02      2.9634      1.34464e-02      2.9392      4.09003e-04     23.3217

 system total =                   1.01158e+00      0.5936      4.97397e-01      0.6116      5.05465e-01      0.5723

elapsed time   0.32000 minutes
 random number=       65bf12f974f8
 sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)
***********************************************************************************************************************************

  position k-effective=  1.01058e+00  + or -  1.07997e-02
  the position k-effective is the largest eigenvalue of the fission production by position index matrix.

***********************************************************************************************************************************

 elapsed time   0.32000 minutes

 fission production by position index matrix
   (  i,  j) p is the number of next generation neutrons produced at position index j by a neutron born at position index i.

   (  0,  0) 0.00e+00  (  0,  1) 0.00e+00  (  0,  2) 0.00e+00

   (  1,  0) 0.00e+00  (  1,  1) 9.02e-01  (  1,  2) 1.80e-01

   (  2,  0) 0.00e+00  (  2,  1) 1.05e-01  (  2,  2) 8.36e-01
```

```
sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

  source vector by position index


   index     vector
     0     0.00000e+00
     1     4.91594e-01
     2     5.08406e-01

                     average self multiplication by array position

the number of next generation neutrons produced in a unit located at a given position in the array by
a neutron born in that same unit is   8.68506e-01 + or -   1.03680e-02
```

```
 sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)
**********************************************************************************************************************************

  unit k-effective=  1.01058e+00  + or -  1.07997e-02
  the unit k-effective is the largest eigenvalue of the fission production by unit number matrix.

**********************************************************************************************************************************

 elapsed time   0.32000 minutes

  fission production by unit number matrix
   (  i,  j) p is the number of next generation neutrons produced in unit j by a neutron born in unit i.

   (  1,  1) 8.36e-01  (  1,  2) 1.05e-01  (  1,  3) 0.00e+00  (  1,  4) 0.00e+00

   (  2,  1) 1.80e-01  (  2,  2) 9.02e-01  (  2,  3) 0.00e+00  (  2,  4) 0.00e+00

   (  3,  1) 0.00e+00  (  3,  2) 0.00e+00  (  3,  3) 0.00e+00  (  3,  4) 0.00e+00

   (  4,  1) 0.00e+00  (  4,  2) 0.00e+00  (  4,  3) 0.00e+00  (  4,  4) 0.00e+00
```

```
sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

  source vector by unit


    unit     vector
     1    5.08406e-01
     2    4.91594e-01
     3    0.00000e+00
     4    0.00000e+00

                    average self multiplication by unit

the number of next generation neutrons produced in a unit by
a neutron born in that same unit is   8.68506e-01 + or -  5.98594e-03
```

```
          sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

              **** fission densities ****

                        fission      percent           total
   unit   region        density      deviation         fissions

     1       1         2.541e-05       1.27            5.150e-01
             2         0.000e+00       0.00            0.000e+00
             3         0.000e+00       0.00            0.000e+00

     2       1         1.111e-04       1.25            4.966e-01
             2         0.000e+00       0.00            0.000e+00

     3       1         0.000e+00       0.00            0.000e+00

     4       1         0.000e+00       0.00            0.000e+00
             2         0.000e+00       0.00            0.000e+00
```

```
                                    sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

fluxes for unit     1
          region    1            region    2            region    3

group    flux        percent     flux        percent     flux        percent
                   deviation               deviation               deviation
  1    4.562e-06     6.61      3.049e-06     8.75      2.708e-06     8.46
  2    3.111e-05     2.40      2.122e-05     3.15      1.994e-05     3.24
  3    3.678e-05     1.93      2.555e-05     2.80      2.336e-05     3.00
  4    2.191e-05     2.51      1.507e-05     2.96      1.426e-05     3.76
  5    2.976e-05     1.88      2.096e-05     2.79      1.963e-05     3.57
  6    4.533e-05     1.37      3.218e-05     1.84      3.095e-05     2.17
  7    4.186e-05     1.64      2.932e-05     2.37      2.533e-05     2.68
  8    3.116e-05     1.51      1.987e-05     2.92      1.276e-05     3.85
  9    2.304e-05     1.57      1.411e-05     3.23      9.343e-06     4.14
 10    2.048e-05     1.57      1.174e-05     3.45      7.731e-06     4.72
 11    1.867e-05     1.69      1.126e-05     3.36      6.976e-06     4.90
 12    1.177e-05     1.76      7.146e-06     3.52      4.336e-06     5.63
 13    9.776e-06     1.94      6.179e-06     3.69      3.764e-06     5.12
 14    1.002e-05     1.61      6.216e-06     3.86      3.722e-06     6.44
 15    4.710e-06     2.59      2.845e-06     5.33      1.570e-06     8.43
 16    2.609e-06     3.22      1.404e-06     7.95      8.302e-07    15.67
 17    1.270e-06     4.18      6.948e-07     8.96      4.440e-07    18.29
 18    1.022e-06     4.39      5.173e-07    10.56      2.996e-07    23.34
 19    1.809e-06     3.65      1.113e-06     7.11      6.580e-07    13.42
 20    5.871e-06     2.41      3.204e-06     5.45      1.799e-06     8.79
 21    1.774e-06     3.37      9.667e-07     8.59      4.870e-07    14.49
 22    2.870e-06     2.66      1.945e-06     6.66      1.020e-06    12.62
 23    6.949e-06     1.63      5.058e-06     3.31      2.878e-06     6.29
 24    6.035e-06     1.74      5.991e-06     3.68      3.408e-06     6.30
 25    3.107e-06     2.15      3.687e-06     4.50      1.868e-06     7.69
 26    2.316e-06     1.91      3.479e-06     4.79      1.794e-06     6.98
 27    4.002e-07     3.30      8.335e-07     9.00      4.124e-07    18.64
```

```
                                  sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

 fluxes for unit    2
          region   1          region   2

 group   flux     percent     flux     percent
                 deviation            deviation
   1    9.495e-06    7.71    5.848e-06    8.19
   2    8.730e-05    2.61    4.709e-05    3.11
   3    1.064e-04    2.22    5.755e-05    2.81
   4    7.086e-05    2.86    3.760e-05    3.42
   5    1.111e-04    2.13    6.086e-05    2.90
   6    1.674e-04    1.92    8.780e-05    2.30
   7    1.146e-04    2.09    5.904e-05    2.47
   8    2.097e-05    4.71    1.396e-05    4.27
   9    3.223e-06    9.65    6.664e-06    7.80
  10    1.639e-06    9.46    5.963e-06    8.55
  11    5.517e-07   12.69    4.152e-06    9.01
  12    4.167e-07   18.19    2.544e-06   16.03
  13    6.270e-07   14.47    2.762e-06   12.48
  14    5.787e-07   15.96    3.013e-06   12.24
  15    1.849e-07   19.89    9.377e-07   17.82
  16    5.271e-08   39.63    5.224e-07   31.58
  17    1.328e-08   48.03    4.389e-07   36.59
  18    4.294e-09   43.39    1.073e-07   37.76
  19    1.640e-08   34.37    4.109e-07   35.82
  20    5.782e-08   21.46    1.092e-06   18.88
  21    7.645e-09   50.42    2.423e-07   41.35
  22    4.211e-09   34.69    2.944e-07   31.30
  23    2.283e-08   17.80    1.913e-06   13.87
  24    1.104e-08   19.52    1.133e-06   20.72
  25    5.778e-09   24.54    5.870e-07   17.31
  26    3.104e-09   25.37    5.550e-07   18.24
  27    7.786e-10   52.20    1.553e-07   44.00
```

```
                                   sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

 fluxes for unit    3
           region   1

 group   flux      percent
                  deviation
   1   0.000e+00    0.00
   2   0.000e+00    0.00
   3   0.000e+00    0.00
   4   0.000e+00    0.00
   5   0.000e+00    0.00
   6   0.000e+00    0.00
   7   0.000e+00    0.00
   8   0.000e+00    0.00
   9   0.000e+00    0.00
  10   0.000e+00    0.00
  11   0.000e+00    0.00
  12   0.000e+00    0.00
  13   0.000e+00    0.00
  14   0.000e+00    0.00
  15   0.000e+00    0.00
  16   0.000e+00    0.00
  17   0.000e+00    0.00
  18   0.000e+00    0.00
  19   0.000e+00    0.00
  20   0.000e+00    0.00
  21   0.000e+00    0.00
  22   0.000e+00    0.00
  23   0.000e+00    0.00
  24   0.000e+00    0.00
  25   0.000e+00    0.00
  26   0.000e+00    0.00
  27   0.000e+00    0.00
```

```
                                sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)

fluxes for unit    4
         region    1           region    2

group   flux        percent      flux      percent
                   deviation              deviation
  1   0.000e+00      0.00     2.262e-06      8.72
  2   0.000e+00      0.00     1.591e-05      3.61
  3   0.000e+00      0.00     1.779e-05      3.47
  4   0.000e+00      0.00     1.076e-05      4.12
  5   0.000e+00      0.00     1.823e-05      3.41
  6   0.000e+00      0.00     2.469e-05      3.06
  7   0.000e+00      0.00     1.771e-05      3.29
  8   0.000e+00      0.00     6.144e-06      5.43
  9   0.000e+00      0.00     3.629e-06      9.32
 10   0.000e+00      0.00     2.593e-06      9.84
 11   0.000e+00      0.00     2.895e-06      9.36
 12   0.000e+00      0.00     1.458e-06     12.25
 13   0.000e+00      0.00     1.428e-06     11.05
 14   0.000e+00      0.00     1.273e-06     11.25
 15   0.000e+00      0.00     5.214e-07     21.15
 16   0.000e+00      0.00     1.970e-07     33.82
 17   0.000e+00      0.00     2.780e-07     30.25
 18   0.000e+00      0.00     1.002e-07     45.58
 19   0.000e+00      0.00     1.495e-07     36.68
 20   0.000e+00      0.00     5.985e-07     17.22
 21   0.000e+00      0.00     1.719e-07     31.13
 22   0.000e+00      0.00     3.513e-07     23.14
 23   0.000e+00      0.00     9.205e-07     16.50
 24   0.000e+00      0.00     8.075e-07     14.26
 25   0.000e+00      0.00     3.834e-07     21.15
 26   0.000e+00      0.00     3.359e-07     20.09
 27   0.000e+00      0.00     5.354e-08     61.40
```

```
sample problem 19 4 aqueous 4 metal array of arrays (samp prob 12)
                              frequency for generations    4 to  103
0.8618 to 0.8849     **
0.8849 to 0.9080     **
0.9080 to 0.9310     ******
0.9310 to 0.9541     ****
0.9541 to 0.9772     *************
0.9772 to 1.0003     ****************
1.0003 to 1.0234     ******************
1.0234 to 1.0465     ***********
1.0465 to 1.0696     **********
1.0696 to 1.0927     ****
1.0927 to 1.1158     *********
1.1158 to 1.1389     ****
1.1389 to 1.1620     *
                              frequency for generations   29 to  103
0.8618 to 0.8849     **
0.8849 to 0.9080     *
0.9080 to 0.9310     *****
0.9310 to 0.9541     ***
0.9541 to 0.9772     ********
0.9772 to 1.0003     ***************
1.0003 to 1.0234     **********
1.0234 to 1.0465     ********
1.0465 to 1.0696     ********
1.0696 to 1.0927     ***
1.0927 to 1.1158     ********
1.1158 to 1.1389     ***
1.1389 to 1.1620     *
                              frequency for generations   54 to  103
0.8618 to 0.8849     **
0.8849 to 0.9080     *
0.9080 to 0.9310     **
0.9310 to 0.9541     ***
0.9541 to 0.9772     *******
0.9772 to 1.0003     *******
1.0003 to 1.0234     ********
1.0234 to 1.0465     ******
1.0465 to 1.0696     *****
1.0696 to 1.0927     *
1.0927 to 1.1158     ******
1.1158 to 1.1389     *
1.1389 to 1.1620     *
                              frequency for generations   79 to  103
0.8618 to 0.8849     **
0.8849 to 0.9080     *
0.9080 to 0.9310     **
0.9310 to 0.9541     ***
0.9541 to 0.9772     **
0.9772 to 1.0003     ****
1.0003 to 1.0234     **
1.0234 to 1.0465     ****
1.0465 to 1.0696     **
1.0696 to 1.0927
1.0927 to 1.1158     ***
1.1158 to 1.1389
1.1389 to 1.1620
********************************************************************************************************************************
         congratulations!  you have  successfully traversed the perilous path through keno v in   0.32000 minutes
*********************************************************************************************************************************
```

F11.F LIST OF KENO V.a VALIDATION REPORTS

This section contains a list of KENO validation reports in reverse chronological order.

Nuc. Technol. **107** (3), 285 (September 1994) — "Criticality Data and Validation Studies of Arrays of Mixed-Oxide Fuel Pins in Aqueous and Organic Solutions," Gary R. Smolen, Sidney R. Bierman, and Nobuo Fukumura.

Nucl. Technol. **107** (3), 304 (September 1994) — "Criticality Data and Validation Studies of Plutonium-Uranium Nitrate Solutions in Cylindrical and Slab Geometry," Gary R. Smolen, Raymond C. Lloyd, and Hideyuki Funabashi.

Nucl. Technol. **107** (3), 326 (September 1994) — "Criticality Data and Validation Studies of Plutonium-Uranium Nitrate Solutions in Cylindrical and Slab Geometry," Gary R. Smolen, Raymond C. Lloyd, and Tomozo Koyama.

Nucl. Technol. **107** (3), 340 (September 1994) — "Criticality Data and Validation Studies of Mixed-Oxide Fuel Pin Arrays in Pu+\U + Gd Nitrate," Gary R. Smolen, Raymond C. Lloyd, and Tadakuni Matsumoto.

ORNL-6510 — *Validation Studies Based on Data From Plutonium-Uranium Nitrate Critical Experiments Conducted in Slab and Cylindrical Geometries*, G. R. Smolen, Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., December 1989.

NEACRP-L-306 — *Standard Problem Exercise on Criticality Codes for Dissolving Fissile Oxides in Acids*, OECD, Paris, France, 1989.

Y/DD-419 — *Validation Check Cases of SCALE77 on the ORGDP IBM-3083*, W. C. Jordan, H. R. Dyer, J. C. Turner, Martin Marietta Energy Systems, Inc., Oak Ridge Y-12 Plant, January 1989.

ORNL-6449 — *Validation Studies Based on Data From Low Concentration Mixed Pu + U Aqueous Critical Experiments*," G. R. Smolen, Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., August 1988.
Notice: APPLIED TECHNOLOGY INFORMATION
Receipt of this report requires approval from:

D. E. Bailey
Director, Division of Fuels and Reprocessing
NE-551
U.S. Department of Energy
Washington, DC 20545

ORNL-6443

Validation Studies Performed With Water- and Organic-Moderated and Reflected Mixed Oxide Fuel Pin Critical Experiments, G. R. Smolen, Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., June 1988.
Notice: APPLIED TECHNOLOGY INFORMATION
Receipt of this report requires approval from:
D. E. Bailey
Director, Division of Fuels and Reprocessing
NE-551
U.S. Department of Energy
Washington, DC 20545

ORNL/CSD/TM-242

Recalculation of a Few Bare Plutonium Critical Arrays, H. R. Dyer, Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., April 1987.

ORNL/CSD/TM-238

Validation of KENO-V.a Comparison with Critical Experiments, W. C. Jordan, N. F. Landers, L. M. Petrie, Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., December 1986.

ORNL/TM-9668

Validation Studies for KENO-IV with Mixed Plutonium-Uranium Critical Experiments, R. T. Primm III, Martin Marietta Energy Systems Inc., Oak Ridge Natl. Lab., November 1985.
Notice: APPLIED TECHNOLOGY INFORMATION
Receipt of this report requires approval from
D. E. Bailey
Director, Division of Fuels and Reprocessing
NE-551
U.S. Department of Energy
Washington, DC 20545

ORNL/CFRP-84/20

Validation of the SCALE Code System and One Cross-Section Library for Plutonium and Gadolinium Solutions, R. L. Sanders, University of Tennessee, July 1985.
Notice: APPLIED TECHNOLOGY INFORMATION
Receipt of this report requires approval from
D. E. Bailey
Director, Division of Fuels and Reprocessing
NE-551
U.S. Department of Energy
Washington, DC 20545

ORNL/CSD/TM-223

Validation of KENO-V.a and Two Cross-Section Libraries for Criticality Calculations of Low-Enriched Uranium Systems, M. E. Easter, Martin Marietta Energy Systems Inc., Oak Ridge Natl. Lab., July 1985.

ORNL/TM-9402
Validation of the SCALE Code System and Two Cross-Section Libraries for Plutonium Benchmark Experiments, M.E. Easter, University of Tennessee, R. T. Primm III, Martin Marietta Energy Systems Inc., Oak Ridge Natl. Lab., January 1985.
Notice: APPLIED TECHNOLOGY INFORMATION
Receipt of this report requires approval from
D. E. Bailey
Director, Division of Fuels and Reprocessing
NE-551
U.S. Department of Energy
Washington, DC 20545

ORNL/CSD/TM-224
Assessment of Computational Performance in Nuclear Criticality, L. M. Petrie, J. T. Thomas, Martin Marietta Energy Systems Inc., Oak Ridge Natl. Lab., January 1985.

ORNL/CSD/TM-221
Validation of the Monte Carlo Criticality Program KENO-V.a for Highly Enriched Uranium Systems, J. R. Knight, Martin Marietta Energy Systems, Inc., Oak Ridge Natl. Lab., November 1984.

CSNI Report No. 78
Standard Problem Exercise on Criticality Codes for Large Arrays of Packages on Fissile Materials, CSNI Working Group, OECD, Paris, France, August 1984.

Trans. Am. Nucl. Soc.
V44, p. 291–293
(US) (1983)
"Validation of the NITAWL-KENO Methodology in Modeling New-Fuel Storage Criticality," D. G. Napolitano, Dr. Harris, P. F. Rose, E. Schmidt, E. Schmidt, M. Divadeennam, Yankee Atomic Electric Co., Framingham, Ma. 01701 (1983).

CSNI Report No. 71
Standard Problem Exercise on Criticality Codes for Spent LWR Fuel Transport Containers, CSNI Group of Experts on Nuclear Criticality Safety Computations, OECD, Paris, France, May 1982.

NUREG/CR-1917
"*Validation of Three Cross-Section Libraries Used with the SCALE System for Criticality Safety Analysis,*" ORNL/NUREG/CSD/TM-19, A. M. Hathout et al., U.S. Nuclear Regulatory Commission, June 1981.

Y-2234
Validation of the Monte Carlo Criticality Program KENO-IV and the Hansen-Roach "Sixteen-Energy Group Cross-Sections for High Assay Uranium Systems," G. R. Handley, L. C. Masters, R. V. Stachowiak, Union Carbide Corp., Nuclear Div., Oak Ridge Y-12 Plant, April 1981.

Trans. Am. Nucl. Soc.
V27, p. 406–407
US (1977)
"Validation of Criticality Safety Broad-Group Library Using Uranium Systems," N. F. Cross, R. M. Westfall, K. R. Turnbull, P. B. Fox, Union Carbide Corp., Nuclear Div., Oak Ridge Natl. Lab. (1977).

Y-1948 "*Validation of the KENO code for Nuclear Criticality Safety Calculations of Moderated, Low-Enriched Uranium Systems*," G. R. Handley and C. M. Hopper, Union Carbide Corp., Nuclear Div., Oak Ridge Y-12 Plant, 1974.

Y-1858 "*Validation Checks of the ANISN and KENO Codes by Correlation with Experimental Data*," G. R. Handley and C. M. Hopper, Union Carbide Corp., Oak Ridge Y-12 Plant, 1972.

NRC FORM 335
(2-89)
NRCM 1102
3201, 3202

U.S. NUCLEAR REGULATORY COMMISSION

BIBLIOGRAPHIC DATA SHEET

(See instructions on the reverse)

1. REPORT NUMBER
(Assigned by NRC, Add Vol., Supp., Rev., and Addendum Numbers, if any.)

NUREG/CR-0200, Rev. 6
ORNL/NUREG/CSD-2R6
Vol. 2, Part 2

2. TITLE AND SUBTITLE

SCALE: A MODULAR CODE SYSTEM FOR PERFORMING STANDARDIZED COMPUTER ANALYSES FOR LICENSING EVALUATION

Functional Modules
F9 – F11

3. DATE REPORT PUBLISHED

MONTH	YEAR
May	2000

4. FIN OR GRANT NUMBER
B0009

5. AUTHOR(S)

*Formerly with ORNL

S.M. Bowman, B.L. Broadhead, C.B. Bryan,* J.A. Bucholz, K.W. Childs, A.L. Edwards,* M.B. Emmett, P.B. Fox, S.K. Fraley,* G.E. Giles, N.M. Greene, O.W. Hermann, T.J. Hoffman,* D.F. Hollenbach, W.C. Jordan, J.R. Knight,* N.F. Landers,* L.C. Leal, K. Lucius,* C.V. Parks, L.M. Petrie, J.C. Ryman, J.S. Tang,* J.C. Turner,* J.T. West,* R.M. Westfall, P.T. Williams

6. TYPE OF REPORT
Technical

7. PERIOD COVERED *(Inclusive Dates)*

8. PERFORMING ORGANIZATION — NAME AND ADDRESS *(If NRC, provide Division, Office or Region, U.S. Nuclear Regulatory Commission, and mailing address; if contractor, provide name and mailing address.)*

Oak Ridge National Laboratory
Post Office Box 2008
Oak Ridge, Tennessee 37831-6370

9. SPONSORING ORGANIZATION — NAME AND ADDRESS *(If NRC, type "Same as above"; if contractor, provide NRC Division, Office or Region, U.S. Regulatory Commission, and mailing address.)*

Spent Fuel Project Office
Office of Nuclear Material Safety and Safeguards
U.S. Nuclear Regulatory Commission
Washington, DC 20555-0001

10. SUPPLEMENTARY NOTES

11. ABSTRACT *(200 words or less)*

SCALE, a modular code system for Standardized Computer Analyses Licensing Evaluation, has been developed by Oak Ridge National Laboratory at the request of the U.S. Nuclear Regulatory Commission. The SCALE system utilizes well-established computer codes and methods within standard analysis sequences that (1) allow an input format designed for the occasional user and/or novice, (2) automate the data processing and coupling between modules, and (3) provide accurate and reliable results. System development has been directed at problem-dependent cross-section processing and analysis of criticality safety, shielding, heat transfer, and depletion/decay problems. Since the initial release of SCALE in 1980, the code system has been heavily used for evaluation of nuclear fuel facility and package designs. This revision documents Version 4.4 of the system.

12. KEY WORDS/DESCRIPTORS *(List words or phrases that will assist researchers in locating the report.)*

SCALE, cross sections, criticality safety, shielding, heat transfer, depletion, decay, spent fuel, KENO, ORIGEN, XSDRNPM, MORSE, NITAWL, XSDRN, SAS1, SAS2, SAS3, SAS4, CSAS, ORIGEN-ARP, QADS, HEATING, HTAS1

13. AVAILABILITY STATEMENT
Unlimited

14. SECURITY CLASSIFICATION
(This Page) Unclassified
(This Report) Unclassified

15. NUMBER OF PAGES

16. PRICE