ITS Version 6: The Integrated TIGER Series of Coupled Electron/Photon Monte Carlo Transport Codes

Revision 1

Brian C. Franke, Ronald P. Kensek, and Thomas W. Laub

Prepared by Sandia National Laboratories Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy’s National Nuclear Security Administration under Contract DE-AC04-94-AL85000.

Approved for public release; further dissemination unlimited.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.
ITSVersion 6:
The Integrated TIGER Series of Coupled Electron/Photon Monte CarloTransport Codes
Revision 1

Brian C. Franke, Ronald P. Kensek, and Thomas W. Laub
Radiation Transport Department

Abstract

ITS is a powerful and user-friendly software package permitting state-of-the-art Monte Carlo solution of lineartime-independent coupled electron/photon radiation transport problems, with or without the presence of macroscopic electric and magnetic fields of arbitrary spatial dependence. Our goal has been to simultaneously maximize operational simplicity and physical accuracy. Through a set of preprocessor directives, the user selects one of the many ITS codes. The ease with which the makefile system is applied combines with an input scheme based on order-independent descriptive keywords that makes maximum use of defaults and internal error checking to provide experimentalists and theorists alike with a method for the routine but rigorous solution of sophisticated radiation transport problems. Physical rigor is provided by employing accurate cross sections, sampling distributions, and physical models for describing the production and transport of the electron/photon cascade from 1.0 GeV down to 1.0 keV. The availability of source code permits the more sophisticated user to tailor the codes to specific applications and to extend the capabilities of the codes to more complex applications. Version 6, the latest version of ITS, contains (1) improvements to the ITS 5.0 codes, and (2) conversion to Fortran 90. The general user friendliness of the software has been enhanced through memory allocation to reduce the need for users to modify and recompile the code.

Acknowledgment

There have been many people involved in the development of the ITS codes over the years. We have undoubtedly lost track of the contributions of
some, and even if we had not, there would be too many to list here. It has been assembled by borrowing and linking algorithms from many generous sources. We are grateful to those who have shared their methods and grateful also to the many users who have suggested improvements over the years. While recognizing that this is necessarily an abbreviated list, we would like to acknowledge some of the contributions that we deem to have been most significant or most recent.

The ITS codes owe most of their development to John Halbleib. He wrote the TIGER code and was the primary developer of the ITS codes throughout most of their history. The original TIGER code was built on the ETRAN code developed by Steve Seltzer and Martin Berger, both of NIST (the National Institute of Standards and Technology). Further collaborations with NIST led to improvements of the algorithms and cross sections used by ITS. Tom Mehlhorn played an important role in the first integration of the Integrated TIGER Series.

This release of ITS includes the multigroup capability. The development of MITS (Multigroup ITS) owes much to the contributions of Jim Morel. MITS is also dependent on the CEPXS cross section generating code that was principally developed by Len Lorence.

The parallel capability of the ITS codes was implemented by Greg Valdez. Much testing and many improvements have been contributed by Wesley Fan. Recent improvements in the physical models in the ITS codes were implemented by Veronica Klein. Recent improvements in the regression testing of ITS are due to the coverage analysis work of Lisa Cordova. The Doppler broadening logic was implemented by Tom Quirk. The below 1 keV logic was implemented by Martin Crawford based on suggestions from Ken Adams.

The conversion of the CEPXS code to Fortran 90 was accomplished by Clif Drumm. The conversion of the XGEN and ITS codes to Fortran 90 was accomplished by Martin Crawford, who has also contributed numerous features to ITS version 6.

The CAD geometry capability in ITS was originally implemented with the assistance of Steve Warren, now at Kansas State University. The implementation was further refined with the aid of Perry Gray, Fred Gelbard, Tim Tautges, Paul Wolfenbarger, and Matthew Martin. The current engine geometry interface with CAD capability was written mostly by Matthew Martin. The Cholla code for facet geometry was written by John Fowler of Computational Geometry Consulting, Inc.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy’s National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Contents

1 Introduction to ITS

1.1 History of the TIGER Series
11 TIGER Geometry................................................................. 91
11.1 Problem Geometry .......................................................... 91
11.2 Conventions for Escaping Particles ....................................... 91
11.3 Geometry Syntax............................................................ 91

12 CYLTRAN Geometry........................................................... 93
12.1 Problem Geometry .......................................................... 93
12.2 Conventions for Escaping Particles ....................................... 94
12.3 Geometry Syntax............................................................ 95

13 ACCEPT Geometry............................................................. 96
13.1 Specification of Input Bodies.............................................. 96
    13.1.1 Body Definition..................................................... 96
    13.1.2 Body Data .......................................................... 101
13.2 Specification of Input Zones ............................................. 102
    13.2.1 Input Zone Definition ............................................. 102
    13.2.2 InputZoneData .................................................... 104
13.3 Subzone Specification.................................................... 105
    13.3.1 Subzone Definition............................................... 105
        SingleBody Subzoning .............................................. 105
        Multi-Body Subzoning ............................................. 107
        AutomaticCAD Subzoning ......................................... 109
        Subzone Overlays.................................................... 110
    13.3.2 Subzone Data ..................................................... 110
13.4 Volume Specification .................................................... 111
    13.4.1 Volume Definition ............................................... 111
    13.4.2 Volume Data ..................................................... 112
13.5 Material Specification ................................................. 112
    13.5.1 Material Definition............................................... 112
13.5.2 Material Data ............................................................... 112

13.6 Conventions for Escaping Particles ........................................... 112

13.7 Geometry Syntax .................................................................. 113

14 CAD Geometry ......................................................................... 116

14.1 GEOMETRY Keyword ........................................................... 116

14.1.1 CAD ONLY .................................................................. 116

14.1.2 HYBRID ...................................................................... 117

14.1.3 MIRRORCG .................................................................. 117

14.1.4 CG ONLY ..................................................................... 117

14.2 Conventions for Escaping Particles ........................................... 117

14.3 CAD Models ...................................................................... 117

14.4 Geometry Syntax .................................................................. 117

15 Suggestions for Efficient Operation .................................................... 119

16 Output Files ............................................................................ 121

16.1 Pre-Processing Information ...................................................... 121

16.1.1 Output Header ............................................................... 121

16.1.2 CAD Parameters (CAD Only) .......................................... 121

16.1.3 Reading Input ............................................................... 121

16.1.4 Reading Cross Section Data ............................................. 122

16.1.5 Processing Input ............................................................. 124

16.1.6 Storage Requirements vs. Allocations .............................. 124

16.1.7 Geometry-Dependent Input ............................................. 124
16.1.8 Source Information..................................................... 124
16.1.9 Output Options.......................................................... 124
16.1.10 Physical Options.......................................................... 125
16.2 Monte Carlo Output.......................................................... 125
16.2.1 Parallel Processing (MPI Only) ....................................... 125
16.2.2 Monte Carlo Errors...................................................... 125
16.3 Results............................................................................. 126
16.3.1 Diagnostics ................................................................. 126
16.3.2 Integral Electron Emission (ITS ACCEPT Only, ELECTRON-EMISSION keyword) ............................................. 130
16.3.3 Integral Escape (Forward Only) ....................................... 130
16.3.4 Boundary Currents (TIGER Only) .................................... 131
16.3.5 Energy and Charge Deposition (Forward Only) ................. 131
16.3.6 Particle Flux (Forward Only) .......................................... 132
   Electron (ELECTRON-FLUX keyword) ................................ 133
   Photon (PHOTON-FLUX keyword) ..................................... 133
16.3.7 Spectrum of Absorbed Energy (ITS Only, PULSE-HEIGHT keyword) .......... 134
16.3.8 Electron Emission (ITS ACCEPT Only, ELECTRON-EMISSION keyword) ..... 134
16.3.9 Escape Spectra (Forward Only) ...................................... 135
   Electron (ELECTRON-ESCAPE keyword) ............................ 135
   Photon (PHOTON-ESCAPE keyword) .................................. 135
16.3.10 Sources and Responses (Adjoint Only) ............................. 136
16.3.11 CAD diagnostics (CAD, not CG ONLY mode)
16.3.12 Timing Data .......................................................... 138

17 PCODES ........................................................................ 140

18 Electric and Magnetic Fields .............................................. 141

19 Biasing Options and Variance Reduction .......................... 143

19.1 Zone-Dependent Cutoff Energies .................................. 143
19.2 Forced Photon Collisions .............................................. 144
19.3 Russian Roulette .......................................................... 144
19.4 Next-Event Estimator for Photon Escape ...................... 144
19.5 Photon Only Transport .................................................. 144
19.6 Scaling of Bremsstrahlung Production (ITS Only) .......... 145
19.7 Scaling of Electron-to-Photon Interactions (MITS Only) .. 145
19.8 Scaling the Probability for Electron Impact Ionization (ITS Only) .................................................. 145
19.9 Scaling of Photon-to-Electron Interactions (MITS Only) .. 145
19.10 Trapped Electrons (Forward Only) ............................... 146

20 Statistics ........................................................................ 147

21 Automatic Subzoning ....................................................... 148

21.1 Non-Conformal Subzone Overlays ................................. 150

22 Random Number Generators ......................................... 152

22.1 Portable Random Number Generators ......................... 152

22.2 Range ........................................................................ 152
22.3 Access to the Seed ....................................................... 152
22.4 Reproducibility ........................................................... 153
22.5 Cycle Length .............................................................. 153
Figures

1 Source position and referenced direction .......................................................... 52
2 Surface indices for ACCEPT bodies ............................................................... 77
3 Example of the half section of a problem cylinder ........................................... 94
4 Rectangular Parallelepiped (RPP) ................................................................. 97
5 Sphere (SPH) .................................................................................................. 97
6 Right Circular Cylinder (RCC) ....................................................................... 98
7 Right Elliptical Cylinder (REC) ..................................................................... 98
8 Truncated Right-Angle Cone (TRC) .............................................................. 99
9 Ellipsoid (ELL) ............................................................................................. 99
10 Ellipsoid of Revolution (ELR) ...................................................................... 100
11 Right Angle Wedge (WED) .......................................................................... 100
12 Box (BOX) .................................................................................................... 101
13 Arbitrary Polyhedron (ARB) ........................................................................ 101
14 Torus (TOR) ................................................................................................. 102
15 Illustration of various methods of combining bodies for specification of input zones . 104
   SPH-SPH subzoning .................................................................................. 108
   TOR-TOR subzoning .................................................................................. 108

Tables

1 Chronology of TIGER series development ..................................................... 11
2 ITS version 5 member codes ......................................................................... 12
3 ITS version 6 member codes ......................................................................... 13
4 ITS keywords and default settings ............................................................... 35
5 MITS forward keywords and default settings .............................................. 37
6 MITS adjoint keywords and default settings ............................................... 39
7 Biasing sub-keywords, default settings, and properties .............................. 40
8 ITS-CAD keywords and default settings ...................................................... 86
9 Data required to describe each body type ................................................... 103
10 Flags available with the "tests.pl" ITTestingscript ...................................... 157
   C.1 Summary of XGEN keywords and default settings ............................. 166
   D.2 List of available elements and default properties in XGEN ................. 171
   F.3 Summary of CEPXS keywords (for use with MITS) and default settings . . .
1 Introduction to ITS

The TIGER series of time-independent coupled electron/photon Monte Carlo transport codes is a group of multimaterial and multidimensional codes designed to provide a state-of-the-art description of the production and transport of the electron/photon cascade. The continuous-energy ITS codes are based primarily on the ETRAN model[1], which combines microscopic photon transport with a macroscopic random walk[2] for electron transport. The multigroup ITS codes are based primarily on the MORSE model[3], with a modification by Sloan[4] to model electron elastic scattering, both of which preserve the angular moments of scattering with discrete scattering angle models. Emphasis is on simplicity of application without sacrificing the rigor or sophistication of the physical model.

1.1 History of the TIGER Series
Table 1 chronicles the development of the TIGER series, beginning with the EZTRAN[5] and EZTRAN2[6] codes in the early 1970’s. These codes were basically user oriented versions of the ETRAN codes. They were severely limited in their application to real physical problems because of their restriction to a single homogeneous material. Overcoming this limitation was the original motivation for the development of the TIGER series.

Table 1. Chronology of TIGER series development

<table>
<thead>
<tr>
<th>Code</th>
<th>Date</th>
<th>Released</th>
<th>Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>EZTRAN</td>
<td>Sep 71</td>
<td>Yes</td>
<td>1-D</td>
</tr>
<tr>
<td>EZTRAN2</td>
<td>Oct 73</td>
<td>Yes</td>
<td>2-D/3-Da</td>
</tr>
<tr>
<td>TIGER</td>
<td>Mar 74</td>
<td>Yes</td>
<td>1-D</td>
</tr>
<tr>
<td>CYLTRAN</td>
<td>Mar 75</td>
<td>Yes</td>
<td>2-D/3-Da</td>
</tr>
<tr>
<td>CYLTRANM</td>
<td>Jun 77</td>
<td>No</td>
<td>2-D/3-Da</td>
</tr>
<tr>
<td>TIGERP</td>
<td>May 78</td>
<td>Yes</td>
<td>1-D</td>
</tr>
<tr>
<td>SPHERE</td>
<td>Jun 78</td>
<td>Yes</td>
<td>1-D</td>
</tr>
<tr>
<td>ACCEPT</td>
<td>May 79</td>
<td>Yes</td>
<td>3-D</td>
</tr>
<tr>
<td>SPHEM</td>
<td>Jul 79</td>
<td>No</td>
<td>1-D/3-Da</td>
</tr>
<tr>
<td>CYLTRANP</td>
<td>Late 81</td>
<td>No</td>
<td>2-D/3-Da</td>
</tr>
<tr>
<td>ACCEPTM</td>
<td>Late 81</td>
<td>No</td>
<td>3-D</td>
</tr>
</tbody>
</table>

The first dimension refers to the material geometry, while the second dimension refers to the description of the particle trajectories.

TIGER[7], CYLTRAN[8], and ACCEPT[9] are the base codes of the series and differ primarily in their dimensionality and geometric modeling. TIGER is a one-dimensional multilayer code. CYLTRAN employs a fully three-dimensional description of particle trajectories within an axisymmetric cylindrical material geometry and quite naturally finds application in problems involving electron or photon beam sources. ACCEPT is a general three-dimensional transport code that uses the combinatorial-geometry scheme developed at MAGI[10, 11].

The original base codes were primarily designed for transport from a few tens of MeV down to 1.0 and 10.0 keV for electrons and photons, respectively. Furthermore, fluorescence and Auger 1

Introduction
processes in the base codes are only allowed for the K-shell of the highest atomic number element in a given material. For some applications, it is desirable to have a more detailed model of the low energy transport. In the TIGERP[12] and CYLTRANP[13] codes, we added the more elaborate ionization/relaxation model from the SANDYL code[14] to the TIGER and CYLTRAN codes, and we extended photon transport down to 1.0 keV (all member codes of the ITS system allow transport over the range 1.0 GeV to 1.0 keV).

In CYLTRANM[15], we combined the collisional transport of CYLTRAN with transport in macroscopic electric and magnetic fields of arbitrary spatial dependence using a Runge-Kutta-Fehlberg algorithm[16] to integrate the Lorentz force equations. An important modification of this algorithm[17] made possible the development of the ACCEPTM code[18], which combines the collisional transport of the ACCEPT code with macroscopic field transport. SPHERE[19] and SPHEM[20] were two special purpose codes that were restricted to multiple concentric spherical shells without and with macroscopic field transport, respectively. EZTRAN, EZTRAN2, and SPHEM are considered obsolete.

Before ITS, that still left us with eight separate code packages to maintain. Five of these – TIGER, CYLTRAN, ACCEPT, TIGERP and SPHERE – had been publicly released and were disseminated through the Radiation Shielding Information Center at Oak Ridge National Laboratory. CYLTRANM, CYLTRANP and ACCEPTM were not publicly released, but were maintained locally for use throughout Sandia National Laboratories. Maintaining multiple code packages had become quite burdensome for us as well as for users of the codes. As a result, important modifications were not being implemented in a timely fashion. Furthermore, the multiplicity of packages had resulted in uneven development of the various codes such that each code had unique features that had not yet been implemented in the other codes.
In order to remedy this situation we developed ITS (the Integrated TIGER Series), whose full implementation superseded all other versions of the TIGER series codes[21]. The combined program library file was obtained by integrating the eight codes in the first three columns and first three rows of Table 2 in such a way as to minimize the repetition of coding that is common to two or more of these codes. This process led quite naturally to the development of a new code, ACCEPTP. In ACCEPTP, the improved low-energy physics of the SANDYL code was added to the ACCEPT code. Those individual codes appearing in Table 1, but not in Table 2, were of a more specialized nature than the others and were no longer supported since their function was duplicated by at least one of the ITS codes. Additional cross-section data and associated logic allowed transport from 1.0 keV to 1.0 GeV[22] for both electrons and photons. A new free-format, order-independent input procedure based on descriptive keywords and maximum use of defaults and internal error checking resulted in a very simple and user-friendly input scheme. Integration of the various codes resulted in the availability of additional common options for each code.

Table 2. ITS version 5 member codes

<table>
<thead>
<tr>
<th></th>
<th>Standard Codes</th>
<th>Enhanced Ionization/Relaxation (PCODES)</th>
<th>Macroscopic Fields (MCODES)</th>
<th>Multigroup (MITS Codes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIGER</td>
<td>ITS-TIGER</td>
<td>ITS-TIGERP</td>
<td>N/A</td>
<td>MITS-TIGER</td>
</tr>
<tr>
<td>CYLTRAN</td>
<td>ITS-CYLTRAN</td>
<td>ITS-CYLTRANP</td>
<td>ITS-CYLTRANM</td>
<td>N/A</td>
</tr>
<tr>
<td>ACCEPT</td>
<td>ITS-ACCEPT</td>
<td>ITS-ACCEPTP</td>
<td>ITS-ACCEPTM</td>
<td>MITS-ACCEPT</td>
</tr>
<tr>
<td>CAD</td>
<td>ITS-CAD</td>
<td>ITS-CADP</td>
<td>N/A</td>
<td>MITS-CAD</td>
</tr>
</tbody>
</table>

ITS was further enhanced throughout the 1980’s to improve the physical models, enhance user friendliness, and increase the options available to the user[23]. Beginning in 1994, a set 1

Introduction
of multigroup ITS codes (MITS) was developed[24, 25, 26]. The MITS codes inherited the ITS combinatorial-geometry logic, but through alternative physical models, it added a capability for adjoint transport calculations. In 1997, we began the development of a capability of tracking particles on CAD geometries[27]. Throughout the 1990’s the original continuous-energy combinatorial-geometry ITS codes were further enhanced with complicated subzoning capabilities, more source options, and flexible biasing schemes. In 2000, ITS, the MITS codes, and the CAD tracking capability were integrated into a single package of codes. In 2002, ITS 5.0 was internally released at Sandia with a verified capability of adjoint particle transport on CAD geometries[28]. The CAD capability was substantially revised in 2004, to improve the efficiency of geometry interrogation queries. Also in 2004, the Cholla capability for using facetted geometry models was implemented. The geometry interface code, engine1, currently allows for ACIS and Cholla geometry. Advanced users may consider linking with their own CAD or other geometry modelers.

Table 3 shows the code options available in the current version of ITS. The codes grouped by row in Table 3 will be referred to as the TIGER codes, the CYLTRAN codes, the ACCEPT codes, and the CAD codes, respectively. From left to right, the codes grouped by column will be referred to as the standard codes, the PCODES, and the MITS codes, respectively. We acknowledge that some confusion may result from a dual context-dependent use of the term “ITS”. In general, we will use “ITS” to mean the complement of the MITS codes (i.e., the first three columns of Table 3), and we will use “ITS codes” to refer to the entire series of codes including the MITS codes. The MCODES capabilities have been integrated into the standard codes and are now enabled by the keyword EBFIELDS.

Table 3. ITS version 6 member codes
Since the initial release[29], feedback from the user community has been of great benefit to the development of the ITS code system. As a consequence of this feedback, subsequent versions have implemented important improvements in physical accuracy, new capabilities, variance reduction, and user friendliness.

2. Overview of the Documentation

The documentation has been written with the intention that it may be used as a reference manual for the expert user and the beginning user alike. Brief overviews are provided on step-by-step execution of the program (in Running ITS), on preprocessor options (in Code Options), and on program keywords (in Summary of ITS Keywords). These may serve as quick references for the expert and as an initial outline for the novice. Each of these are followed by more detailed explanations that in turn may be associated with sections containing further details and/or theory for the code features. It is intended that all users should begin by referring to the Running ITS section and follow references in the documentation as necessary to gain further understanding of features to be used. Users are encouraged to peruse the entire manual to gain a better understanding of the code package and how various options may be employed. However, some sections apply to restricted subsets of the code options available and may not be relevant to the user’s specific types of problems.

2.1 Document Sections

The following is a brief description of each of the sections contained in this manual:

- Introduction to ITS: discusses the history of the ITS codes.
- Overview of the Documentation: is this section.
Overview of the ITS Code Package: gives a brief description of the cross section generators and Monte Carlo codes that comprise the ITS code package and a discussion of the code capabilities available.

Installation: gives general guidance for installing the ITS software on a platform. It discusses some of the modifications that may be necessary in the configure/make system of files.

Running ITS: contains an overview, as well as step-by-step instructions, for executing a calculation with ITS. Some known platform dependencies of the code are listed.

Code Options: contains the preprocessor definitions available for selecting member codes of the Integrated TIGER Series.

Summary of ITS Keywords: contains tables of input keywords available for multigroup forward, multigroup adjoint, and continuous-energy options and the default settings associated with those keywords.

Keywords for ITS: contains an alphabetical listing of the input keywords for all of the ITS codes and descriptions for using each keyword.

Summary of ITS-CAD Keywords: contains a table of keywords available for the parameter file used with CAD calculations and default settings associated with those keywords.

Keywords for ITS-CAD: contains a listing of the parameter keywords for use in the parameter file with the CAD code option and descriptions for using each keyword.

2. Overview of the Documentation

TIGER Geometry: contains the formatting requirements for input of 1-D geometry.

CYLTRAN Geometry: contains the formatting requirements for input of 2-D geometry.

ACCEPT Geometry: contains the formatting requirements for input of 3-D combinatorial geometry.

CAD Geometry: contains the requirements for input of 3-D CAD geometry models.

Output: contains descriptions of the information in each section of an output file.

Suggestions for Efficient Operation: discusses some of the issues involved in using a Monte Carlo code efficiently, such as the proper selection of energy ranges, biasing parameters, and number of particle histories simulated.

PCODES: contains a description of the ITS options for detailed ionization and relaxation modeling.

Electric and Magnetic Fields: contains a description of the field options.

Biasing Options and Variance Reduction: contains explanations of the biasing settings available in the ITS codes.

Statistics: contains a description of how statistical estimations are performed in ITS.

Automatic Subzoning: contains descriptions of how subzoning is implemented in ITS codes.

Random Number Generators: contains descriptions of the portable
random number generators implemented.

Adjoint Calculations: provides some theoretical
description of the adjoint mode of the MITS code option.

Testing of ITS: contains instructions on how to perform installation,
commit, and regression tests for ITS.

Unit Testing of ITS: contains descriptions of each of the unit tests for
ITS that are contained in the repository and how to perform those tests. This
is available as a separate document outside of the manual.

ACIS Library Build for ITS-CAD: contains information on building the
ACIS libraries and specific modifications that have been necessary to build
the libraries on certain platforms. This is available as a separate document
and is only available to Sandia developers.

3. Overview of
the ITS Code
Package

Last Modified Date: 2008/04/17 22:45:31

3 Overview of the ITS Code Package

ITS consists of three essential code directories:

1. XGEN – the continuous-energy cross-section generation program
2. CEPXS – the multigroup cross-section generation program
3. ITS – the Monte Carlo program

XGEN is used to generate cross sections for the continuous-energy ITS
codes. Relatively few physics and modeling decisions are required when
these cross sections are generated, but for example, one must choose
between the standard codes and the PCODES. CEPXS is used to generate
cross sections for the multigroup ITS codes. A number of physics and
modeling decisions can be (or must be) made at the time the multigroup
cross sections are generated. The heart of ITS is the set of Monte Carlo
program files. Here, numerous decisions must be made about the
simulation, e.g., problem geometry, physics options, forward vs. adjoint,
biasing options, etc.

3.1 NewCapabilities Since 5.0

Array allocation is the foremost feature added to ITS since
version 5.0. The requirement that a user modify array-size
parameters and recompile for different calculations has been greatly reduced.
Instead, ITS now reads the input file to dynamically allocate almost all arrays to the
required sizes. While most arrays are dynamically allocated based on the
contents of the input file, some array sizes are still hardwired. In a few cases,
keywords have been made
available to allow the user to increase these array sizes without code
modification and recompilation. For example, the CODEZONES-PER-PATH,
TRAJECTORY-POINTS, OVERLAP-MAXOUTPUT, and PLOTS-OVAL-
RESOLUTION keywords serve this purpose.
Along with automating the memory allocation functions, we have provided the user with keywords that assist in controlling memory usage. The new keywords NO-DEPOSITIONOUTPUT and NO-SZDEPOSITION-OUTPUT can be used to suppress printing deposition in the output file, while the new keywords NO-DEPOSITION and NO-DETAILED-DEPOSITION can be used to limit the tallies and memory allocated for deposition. To limit the memory required for tallying photon flux and escape, line radiation can be tallied with continuum radiation using the new LINE-TALLY-WITH-CONTINUUM and ANNIHILATION-LINETALLY-WITH-CONTINUUM keywords.

Electrons can now be transported in CAD geometries in the continuous-energy codes.

The user can now run the PCODES without uniform shell biasing enabled, and uniform shell biasing is no longer the default behavior of the PCODES. Instead this can be activated with the BIAS-SHELL sub-keyword under the BIASING keyword. With this change we now recommend that the PCODES be used for almost all calculations.

We have begun to add support for radiation below 1 keV. For now, this is restricted to source photon sampled below 1 keV (i.e., radiation is still not produced below 1 keV) and only absorption is modeled. No scattering of radiation below 1 keV is included. See the BELOW-1KEV keyword for more details. The new capability for attenuation of source photons below 1 keV requires the Livermore evaluated data library for photons, which is not distributed with ITS. It is available for download from the IAEA website at http://wwwnds.iaea.org/epdl97/. ITS uses the EPDL97 in ENDF/B-VI format. This file should be located in the XGEN distribution at .../Code/XSdata/epdl97.all.

The efficiency of large combinatorial geometries can be improved using features to read and write the geometry connectivity array. See the READ-CONNECTIVITY and DUMPCONNECTIVITY keywords for more details.

We have implemented a more efficient parallel algorithm and more robust parallel error handling. The improved efficiency is most significant for calculations performed on thousands of processors. Dynamic load balancing is now accessed as a keyword, DYNAMIC-MPI, rather than as a preprocessor definition.

The field codes have changed considerably as the MCODES have been integrated into the standard codes. There are now more options for selection of fields without code modification. See the EBFIELDS keyword for more details. See the following section on changes to input requirements also.

The PLOTS capability is now an integral part of the CYLTRAN and ACCEPT codes. PLOTS is invoked only with a keyword, without requiring a unique preprocessor definition. Plotting is now amenable to use with spreadsheets and plotting programs like Tecplot.

Material thickness information can be extracted to a separate file while
performing a RAYTRACE calculation. See the AREAL-DENSITY-OUTPUT keyword for more information.

ITS now performs the proper normalization for volumetric sources in adjoint.

Electron escape and emission data can be output to a PFF formatted file. (The user must supply the pfflibraries.) See the PFF-FILE keyword for more details.

Deposition data can be written to a separate file in a simple list format with a variety of formatting options. These are options to the FINITE-ELEMENT-FORMAT keyword.

ITS now allows other files to be “included” in an input deck with the INCLUDE-FILE keyword. This allows a user to eliminate duplicate information between input decks. E.g., one file with the combinatorial geometry data used for multiple runs with differing output data requirements.

Adiffutility capable of suppressing differences in numerical data below specified tolerances is now included with ITS. This can simplify the task of performing installation testing of ITS and XGEN on a new platform.

3.2 Changes to Input Requirements Since 5.0

MCODES was removed as a preprocessor definition and is replaced by

the new EBFIELDS keyword. MCODES input decks required enabling fields on the geometry line for CYLTRAN and on the material line for ACCEPT. This is no longer allowed, and the EBFIELDS keyword allows the user to define the zones where fields are enabled or disabled using OFF (was value ‘0’), MAGNETIC (was value ‘1’), or BOTH (was value ‘2’).

3. Overview of the ITS Code Package

BIAS-GLOBAL and BIAS-ZONE are obsolete keywords. The BIASING keyword provides equivalent functionality.

DETAIL-IONIZE is an obsolete keyword. This functionality is no longer available.
4 Installation

As noted in the Code Overview, there are three components in the ITS code package:

1. XGEN – the continuous-energy cross-section generation program
2. CEPXS – the multigroup cross-section generation program
3. ITS – the Monte Carlo program

These code directories should be included in any ITS release.

4.1 Unix, Linux, and Mac Installation

In this section we describe the anticipated one time only modifications that may be required to install the ITS package on a new platform. These modifications are made in the configure and Makefile systems. No modifications are anticipated for the codes themselves. Note that the following instructions are written for the ITS code directory, but apply similarly to the XGEN and CEPXS code directories. There should be little difference in the config files for ITS, XGEN, and CEPXS.

WARNING: Before performing any modifications to the files provided, we strongly recommend that an unaltered copy be stored! This will allow for future determinations of code modifications that were necessary.

From within the its/Code directory, execute "./configure". The script will respond with a statement of the form “Configuring for a *-*-* host.” Only the last of the three variables (the operating system) is important for configuring. If header and target files exist and are found in the config directory, the configure script will print a statement of the form: 'Created “Makefile” using "config/mh-*" and "config/mt-*"'. If the script prints the statement ‘Created “Makefile” but does not state which header and target files were used, then the necessary files do not exist, the configure failed, and the Makefile will not function.

If the configure failed, the scripts must be modified to use config files customized to the platform. The simplest approach is simply to modify the config/mh-custom and config/mt-custom files to supply the necessary information specific to the platform. The configure command “configure -host=i386-custom” will include the mh-custom and mt-custom files in the Makefile. The “i386” does not affect the Makefile and can be used regardless of the system you are working on. For using the testing and sendn scripts, the nm6CVS file (located in its/Scripts, but also to be copied to $HOME/bin) must be altered to replace the text “PLACEHOLDER” with the system response to the command “uname -n”.

A more complicated approach is to create header and target files specific to the platform and identifiable by the configure script. As an example, the configure script may identify the host as rs6000-ibm-aix4.3.2.0. This indicates that the operating system has been identified as aix4.3.2.0. The configure script will use the config files config/mh-aix and config/mt-aix to create the
Makefile. On the other hand, if the configure scripts were to identify the host as i686-unknownexample3.2, then the user would need to create scripts named "config/mh-exa" and "config/mtexa", or "config/mh-example3.2" and "config/mt-example3.2", or such. It will be easiest to create the new scripts by modifying two of the existing configure scripts, so that the required variables and formats are available. (The mh-custom and mt-custom are well commented for this purpose.)

4.

Installation

When new configure files are created, it is necessary to modify the configure.in file, so that the configure script can find the new files. The use of an asterisk at the end of the name of the configure file will indicate whether the name of the operating system must be identified exactly or if only the given leading characters need to be identified. There are several target files ("config/mt-*") for which the exact name must be identified.

There are times when it is desirable to be able to compile in different ways on a single platform. For example, one may wish at times to compile on a solaris machine to run on that same machine, in which case the host and target machines are the same. Or one may wish to (cross-)compile on the solaris machine to run on another machine, in which case the host and target machines are different. In this case, one can create a configure file for the target machine that can be selected as a flag when configuring. One then only needs to specify the target machine in the configure commands (e.g., "configure -target=i386-tflops"). If one is adding a new unique target option, it is necessary to add the option to the configure.in file and to the config.sub file. The changes required in the config.sub file may be identified by searching for "tflops". Specification of a target machine by name can be useful not only for cross compiling but also for installing the software on an unusual instance of an operating system, while maintaining the target configure file for the usual operating system.

There are a few modifications that might be needed to use the CAD capability in ITS with the Cholla libraries. The Cholla library has its own makefile system that is invoked by the ITS makefile system. The name of the compiler (CCC from the mt-* file) and the optimization/debug flag (CCFLAGS
SCRIPT from the Defines.mk file) are passed into the Cholla makefiles. However, if there are compile flags that are required on your system for compiling C++ code, these must be specified in the Cholla makefiles. These modifications must be included in three files: Code/Cholla/Makefile, Code/Cholla/facetbool/makefile, and Code/Cholla/primitives/makefile.

When the Makefile has been successfully created, one must attempt to make an executable. To do this, it is necessary to specify a valid set of preprocessor definitions. This can be done in Defines.mk. Changes to this file will not be affected by the configure command. The Template.mk file should not be modified, as it serves as a backup for the Defines.mk. Valid options are listed in this file. Examples of valid options are “RNG = RNG1”, “OPT1 = MITS”, and “OPT2 = “.

An executable can then be produced by executing “gmake”. The ITS makefile system requires a version of make that supports conditionals. The makefile system is currently hardwired to use gmake. If you have an appropriate make utility (e.g., one that handles conditionals), you will need to alias gmake to invoke your make utility. If gmake is not successful, it will be necessary to identify the settings in the Defines.mk and/or config files that need to be modified. Our general experience is that the problem is (1) most likely to be in the specification of preprocessor definitions in the Defines.mk file or the compilers in the target config file, (2) possibly in other settings in the target config file, and (3) least likely to be in the host config file or Makefile.in.

We then recommend that the installation tests be run to determine that the software is functioning properly on the new platform. To run the tests it is necessary to copy the sendn and nm6CVS files from the Scripts directory to a $HOME/bindirectory. It is also necessary to copy the Subscripts directory to $HOME/bin/Subscripts. If creating a Makefile requires a command other than simply “./configure” without flags, it may be necessary to modify the $HOME/bin/nm6CVS script to recognize the platform dependency. A guide for running the installation tests is provided in the section on Testing of ITS.

When the installation tests have been successfully executed, the software has been successfully installed.

Finally, we recommend that the user consider the advantages of importing the software into a
CVS version control system[31]. The original files provided should be imported first. Then any modifications required to install the software on the platform can be committed to the repository. This provides a convenient mechanism for storing and tracking future code modifications and extracting changes that have been made to the code over time.

4.2 PC Installation

These installation instructions are for the CG version of ITS using Visual Studio. The instructions were tested using Visual Studio 2005 with the Intel Fortran compiler 10.1.011 [IA-32]. These instructions can easily be adapted to XGEN or CEPXS.

With the code directory installed on your PC, create a new Visual Studio Project:

- File->New->Project
- Select Intel(R) Fortran
- Select Visual Studio Installed Template: Empty Project.
- Name: its

Location: (base directory)
Uncheck the Create directory for solution.
Click [OK]. Add the project files into the ITS Visual Studio Project:
- For the files in the Code/Source directory:
  - Click on the ITS project.
  - Right-click on Source Files folder in the project tree.
  - Select menu item Add->Existing Item.
  - Select Source item in Visual Studio installed templates.
  - Browse to the Code/Source directory.
  - Select all the FORTRAN files (*.F).
  - Click on the [Add] button.
- Repeat for the files in the Code/Interface directory.
- Repeat for the files in the Code/Modules directory (but OMIT the test-*F files). Set the type of compilation for the project:
  - Select menu item Project->its Properties.
  - Use Configuration drop-list box to select Release, Debug, or All Configurations. Change the compile definition for *.F files to use 132 character lines, free form:
    - Select menu item Project->its Properties.
Configuration Properties->Fortran->Language->Source File Format: "Use Free Format". Set the Data configuration:

- Configuration Properties->Fortran->Data->Default Integer KIND:4
- Configuration Properties->Fortran->Data->Default Real KIND:8
- Configuration Properties->Fortran->Data->Default Double Precision KIND:8

Configuration Properties->Fortran->Data->LocalVariable Storage: AllVariablesSAVE

Configuration Properties->Fortran->Data->Initialize stack var...:Yes

Optionally, set run-time checking. The following settings will cause the code to run slower but may help detect coding errors:

- Configuration Properties->Fortran->Run-time->GenerateTraceback Information:Yes
- Configuration Properties->Fortran->Run-time->Check Array and String Bounds:Yes
- Configuration Properties->Fortran->Run-time->Check UninitializedVariables:Yes Add Include Directory: Additional Include Directories:

.../Code/Source Define the CodeTypePreprocessor definitions:

Select menu item Project->Its Properties. Turn on the preprocessor:

Configuration Properties->Fortran->Preprocessor->Preprocess Source File-Yes

Add the code-specific compiler flags: Configuration Properties-

>Fortran->Preprocessor Select from the following and enter as comma separated (e.g., "ITS RNG1,ITS CYLTRAN"):

- Required: ITS RNG1, ITS RNG2, or ITS RNG3
- Required: ITS TIGER, ITS CYLTRAN, or ITS ACCEPT
- Optional: ITS MULTIGROUP (not allowed with ITS CYLTRAN)
- Optional: ITS PCODES (if using a PCODES version of XGEN cross sections)

- Optional: ITS USE PFF
(requires PFF libraries) Optionally, change the directory and name for the executable:

- Create the directory, e.g., C:/tmp/its
- Change the executable name:

  Configuration Properties->Linker->Output File:

C:/tmp/its/its.exe Build your executable:
  Build->Build Solution (or Build its)

To test your build:

- Copy the cross section file to the executable directory
  E.g., copy ../../Tests/RegTests/cross3/taala to C:/tmp/its/fort.11

- Copy the input file
  E.g., copy ../../Tests/RegTests/Input/itscyl1.inp to C:/tmp/its/

- Open a command window.
  Change to the C:/tmp/its directory
  run "its.exe itscyl1.inp output" from a command prompt.

Optionally, run using the Visual Studio as the debugger:
  Modify Configuration Properties->Debugging->Command Arguments:
  itscyl1.inp output

  Modify Configuration Properties->Debugging->Working Directory:
  C:/tmp/its

Verifying the output:
  E.g., diff(... full well-defined path ...)/Tests/RegTests/Output/itscyl1

1.out output

If you want to perform more tests you can match input, cross sections, and output by examining the installation.pm, commit.pm, and regression.pm files in the.../Tests/RegTests directory. Expect some differences because:

  Some lines have been removed from the stored output files.
Some leading 0’s have been added to the stored output files. There could be some round-off differences between different platforms. We frequently see very small numbers being identically zero on different platforms.

5

Running ITS

This document contains outlines for running the ITS code. Instructions are provided for running the code by using commands and running the code by using scripts either with or without access to the CVS repository. Known platform dependencies of the code are discussed in the last section.

5.1 Running ITS without Scripts

CROSS SECTIONS: Cross sections must be generated either with XGEN (for the continuous-energy codes) or with CEPXS (for the multigroup codes). The fort.11 file is needed by ITS.

CHECKOUT: Do a “cvs checkout -P its” to acquire a copy of the code on the platform where the repository is located. If necessary, tar the directory and transfer it to the desired platform.

CODE MODIFICATIONS: Code modifications can be made before building an executable.

MAKEFILE SETTINGS: In the directory its/Code, you must alter the Defines.mk settings to specify the necessary definitions for your build of the ITS executable. There are no valid defaults! An unaltered copy should remain stored as Template.mk.

CONFIGURE: Execute “./configure”. If the code has been previously
installed correctly, then the platform and operating system will be identified, and the proper config/mh-* file and con-fig/mt-* file will be included in the Makefile. It may be necessary to specify a target platform when configuring.

MAKE: Execute the Makefile with the “gmake” command to produce the ITS executable, its.x.

INPUT FILE: An ITS input file must be constructed. Examples are available for each code option in its/Tests/RegTests/Input. See the section on Keywords for ITS for additional information.

CAD FILES: If performing a CAD calculation, a prm file must be constructed and either sat files or facet files must be provided to specify the geometry.

EXECUTION: The command “its.x mdat output” executes the ITS code, where mdat is the ITS input file. (The files its.x, fort.11, and mdat are required.) If no input file name is specified, the input file defaults to “its.inp”. If no output file name is specified, the output file defaults to “its.out”. If performing a CAD calculation, the command “its.x prmfile” executes the ITS code. (The files its.x, fort.11, prmfile, mdat, and sat files or facet files are required.)

EVALUATE RESULTS: The output file should be evaluated to determine if the run was successful and if the results are satisfactory. For an unsuccessful calculation, an abort call will usually be indicated at the end of the output file. It may be necessary to look for errors in the output file (search for “>>>>>”). The abort may have been postponed until all input had been read, and one error may be the cause of additional errors.

5.2 Running ITS without Scripts - Details

CROSS SECTIONS: Cross sections must be generated either with XGEN (for the continuous-energy codes) or with CEPXS (for the multigroup codes). Unless the FILE-NAMES keyword is used in the ITS input, the file must be
named “fort.11”. More detailed discussion of generating cross section files can be found with the documentation for the XGEN and CEPXS codes.

CHECKOUT: Doa “cvs checkout -P” to acquire a copy of the most recent version of the code on the platform where the repository is located. It is recommended that a -P flag be used when checking out a copy of the code to avoid acquiring empty directories that correspond to outdated directory structure. A variety of date and release tags can be used to request older versions of the code. See CVS documentation at http://ximbiot.com for additional information. If necessary, tar the directory and transfer it to the desired platform.

CODE MODIFICATIONS: If desired, code modifications can be made before building an executable. If you do not have direct access to the CVS repository, it is strongly recommended that an unaltered version of the code be stored for tracking any intentional or unintentional alterations that you make.

MAKEFILE SETTINGS: In the directory its/Code, you must alter the settings in the Defines.mk file to specify the necessary definitions for your build of the ITS executable. There are no valid defaults! An unaltered copy should remain stored in Template.mk. Instructions for setting the preprocessor definitions are included in the Defines.mk file and in the Code Options section of this manual. It is important that options be set under the proper variables (e.g., MITS should be selected under OPT1, MPI should be selected under OPT2, etc.). There are some options for which it is valid to leave the definition blank, and they are illustrated in Defines.mk (e.g., OPT1 can be MITS, PCODES, or blank).

CONFIGURE: Execute “./configure”. If the code has been previously installed correctly, then the platform and operating system will be identified, and the proper config/mh-* and config/mt files will be included in the Makefile. It may be necessary to specify a target platform when configuring. See Section 5.5 for information on known platform dependencies.

MAKE: Execute the Makefile with the “gmake” command to produce the ITS executable, its.x. If the code has been installed on the platform correctly, the correct commands will be used. The ITS makefile system requires a version of make that supports conditionals (it is currently hardwired to use gmake).

INPUT FILE: An ITS input file must be constructed. Examples are available for each code option in its/Tests/RegTests/Input. The keywords relevant to specific code options and their defaults are given in the Summary of ITS Keywords section. Specific instructions for formatting each keyword in the ITS input are given in the Keywords for ITS section.

CAD FILES: If performing a CAD calculation, a prmf file must be
constructed and satfiles or facetfiles must be provided. Instructions for setting the CAD parameters in the prm file are provided in the Keywords for ITS-CAD section. One or more satfiles should contain any desired CAD geometry in ACIS SAT format, and facetfiles should contain any desired CAD geometry in CUBIT facet format.

RUNNING ITS

EXECUTION: The command “its.x mdat output” executes the ITS code, where mdat is the ITS input file. (The files its.x, fort.11, and mdat are required to perform a calculation.)

If no input file entered as a parameter to the command, the input file defaults to “its.inp”. If no output file entered as a parameter to the command, the output file defaults to “its.out”.

If performing a CAD calculation, the command “its.x prmfile” executes the ITS code. (The files its.x, fort.11, prmfile, and mdat are required to perform a calculation, as well as any geometry files. The names of the mdat file, satfiles, and facetfiles are specified in the prmfile.)

EVALUATE RESULTS: The output file should be evaluated to determine if the run was successful and if the results are satisfactory. For an unsuccessful calculation, an abort call will usually be indicated at the end of the output file. It may be necessary to look for errors in the output file (search for “>>>>>”). The abort may have been postponed until all input had been read, and one error may be the cause of additional errors. If an error statement is generated, then an anticipated problem has been found in the input and/or cross sections, and diagnostic information should be available. If the code generates an execution error, the user may have introduced a bug via an code modification. If a bug is found in ITS that was not introduced by the user, please notify the ITS developers providing enough details to reproduce and understand the error. Preferably, this would include a description of the error observed, the version of the code being used, diffs showing any code modifications made, and the input and output files demonstrating the bug.
5.3 Running ITS with Scripts

CROSS SECTIONS: Cross sections must be generated either with XGEN (for the continuous-energy codes) or with CEPXS (for the multigroup codes).

CHECKOUT: Do a “cvs checkout -P its” to acquire a copy of the code on the platform where the repository is located. If necessary, tar the directory and transfer it to the desired platform.

BACKUP: Making an unaltered backup copy can be very important for tracking code changes when working on a platform that does not have access to the repository.

SCRIPTS: The scripts sendn and nm6CVS that are located in its/Scripts must be copied to $HOME/bin. The scripts in its/Scripts/Subscripts must be copied to $HOME/bin/Subscripts.

CUI FILE: Copy the its/Scripts/its.cui file to your working directory, and edit it for your specific problem. The sections of a cui file are:

1. driver script - the default nm6CVS is usually acceptable.
2. defs - selects code options. See Code Options for more information.
3. prmfile - only required for CAD calculations. See Keywords for ITS-CAD for information.
4. satfile - only required for CAD calculations with ACIS geometry (repeated for each file).
5. facetfile - only required for CAD calculations with facet geometry (repeated for each file).
6. diffs - this section must be present. Code modification patches may be specified.
7. mdat - contains the ITS input. See Keywords for ITS for more information.

SENDN: Submit the job using the command “sendn its.cui <jobname>”. This script will prompt you for 3 or 4 pieces of information. The information consists of the cross section file to be used, whether the code will be compiled in an existing directory or from a CVS checkout, and how to document code modifications.
EXECUTION: If calculations are performed in “interactive” mode (requiring the user to execute the code), the files will be located in $HOME/tmp/<jobname>. The code will be executed as “its.x mdat output” for non-CAD calculations and “its.x prmfie” for CAD calculations.

POSTPROC: If calculations are performed in “interactive” mode, the postproc script must be executed in the $HOME/tmp/<jobname> directory to complete the script process.

EVALUATERESULTS: The output file should be evaluated to determine if the run was successful and if the results are satisfactory. For an unsuccessful calculation, an “ohoh<jobname>.job” file will be returned. Information will be included in the ohoh file that may indicate the source of the error. Additional information may be found in the $HOME/tmp/<jobname> directory. The mlog2 file contains most of the information generated while the scripts were run. If the program compiled successfully but an error was generated during the execution of ITS, an abort call will usually be indicated at the end of the output file. It may be necessary to look for errors in the output file (search for “>>>>>>”). The abort may have been postponed until all input had been read, and one error may be the cause of additional errors.

5.

Running ITS

5.4 Running ITS with Scripts -Details

CROSS SECTIONS: The sendn script will request the name of a cross section file. For the continuous-energy codes, the cross section file must be located in $HOME/cross3. For the multi-group codes, the cross section file must be located in $HOME/crossm and must be given a name with the extension “.11”. More detailed documentation on generating cross section files can be found with the XGEN and CEPXS codes.

CHECKOUT: Do a “cvs checkout-Pits” to acquire a copy of the most recent version of the code on the platform where the repository is located. It is recommended that a -P flag be used when checking out the code to avoid acquiring empty directories that correspond to outdated directory structure. A variety of date and release tags can be used to request older versions of the code. See CVS documentation at http://ximbiot.com for
additional information.

BACKUP: For working on platforms that do not have direct access to the repository, it is recommended that two checkout copies be set up: one copy to be used as a working directory in which code modifications and builds can be performed, and one unaltered copy that can be compared with to maintain a record of changes made to the working version. The unaltered copy should be given a name uniquely indicating the CVS version. For example, if the checkout was performed as (cvs checkout -D "February 1, 2002" its), then the unaltered copy might be named "its01Feb2002". The name of the directory will be the indication of the version, which is a very important key to repeating a calculation at some later time.

SCRIPTS: The scripts sendn and nm6CVS that are located in its/Scripts must be copied to $HOME/bin. The scripts in its/Scripts/Subscripts must be copied to $HOME/bin/Subscripts. Sendn is a script for launching jobs, nm6CVS is a driver script for performing jobs, and the Subscripts are utilities used by sendn and nm6CVS. Sendn will position the cui file, including the sections of the driver script. The driver script contains the commands necessary to build and execute the program, clean up after itself, and produce relevant result information in a job file. (If necessary, it will also include in the job file information useful in determining the cause of a job failure). $HOME/bin and "." should be included in your $PATH environment variable.

CUIFILE: The its/Scripts/its.cui file is available as an example. You can copy it to your working directory, and edit it for your specific problem. Other examples are available in Tests/Reg-Tests/CUI. The portions of a cui file are:

1 DRIVER SCRIPT: The script nm6CVS is available as a driver script. You may substitute a customized script by including it in the first portion of the cui file.

2 DEFS: Instructions for setting the preprocessor definitions are included in the "defs" portion of the its.cui file and in the Code Options section. It is important that options be set under the proper variables (e.g., MITS should be selected under opt1, and MPI should be selected under opt2), because the script will look (i.e., grep) for these specific settings. There should be no spaces in the settings.

There are several options for running the scripts that may be set in the defs portion of the cui file. Compiler flags may be specified. The user may also request an interactive script. The interactive script should be used on systems that require job queuing or cross compiling. The first half of the script will build the executable. The files for running the job will be located in $HOME/tmp/<jobname>. The executable is named "its.x", the input file is named "mdat", the parameter file for ITS-CAD is named "prmfile", and the program output should be directed to a file named "output".

5
When the calculation is complete, the user may execute the “postproc” file in the $HOME/tmp/<jobname> directory. This will produce a job file in the directory from which the job was originally submitted.

1 PRMFILE: Instructions for setting the CAD parameters in the “prmfile” portion of the its.cui file are provided in the Keywords for ITS-CAD section. This portion is only required for CAD calculations. If the full path for any satfile or facetfile is included in the listing in the prmfile, the satfile and facetfile sections are not necessary; they are only required when those files must be located in the same directory as the executable.

2 SATFILE: The “satfile” portion is used to transfer the desired ACIS CAD geometry to the location in the tmp directory where the calculation will be performed. Only two lines are required. The first line should include the desired *.sat file containing the CAD geometry. The second line may contain anything but must contain a return. This portion is only necessary for including ACIS geometry in CAD calculations. The name given to this section (typically “satfile.sat”) should be the name referenced in the prmfile. It is possible to have multiple satfiles as long as each section has a unique name and is referred to appropriately in the prmfile.

3 FACETFILE: The “facetfile” portion is used to transfer the desired Cubit facet CAD geometry to the location in the tmp directory where the calculation will be performed. Only one line is required specifying the path and name of the file containing the facet geometry. This section is only necessary for including facet geometry in CAD calculations. The name given to this section (typically “facetfile.fac”) should be the name referenced in the prmfile. Since one may only include a single zone in each facet file, it is often desirable to have multiple facet files. This is easily accomplished as long as each section has a unique name and is referred to appropriately in the prmfile.

4 DIFFS: Patches to be applied to the code should be inserted in the “diffs” portion of the its.cui file. These patches can be formatted as a cvs diff or as a directory diff. Diffs can be lifted out of job files from previous calculations. Alternatively, one can checkout a copy of the code, make modifications, and then generate a diffs file. On a platform with access to the CVS repository, one can use “cvs diff > diffs” from within the its/Code directory. On a platform without access to the CVS repository, one can perform a directory diff, but this must be performed from within the its/Code directory of the unaltered copy of the code, and it must use the -r and -b flags (in that order), such as “diff -r -b $HOME/itsaltered/Code> diffs”. Then, the diffs file can be used in the its.cui file. Multiple diff files (diffs from two different executions of the diff command) cannot be patched to a single cvs file.
New files can be added to a build. Within the diffs portion, a line of the following format denotes the start of a new file:

New file: <Directory/Filename> To be included in the compiling and linking of the code, the new file must be referenced in the Makefile.in list of sources. This change in the Makefile.in can also be included in the diffs portion by making the desired change in a copy of Makefile.in and using the cvs diff procedure described above.

7.MDAT: The ITS input should be included in the last portion of the cui file. The keywords relevant to specific code options and their defaults are given in the Summary of ITS Keywords.

Running ITS section. Specific instructions for formatting each keyword in the ITS input are given in the Keywords for ITS section.

SENDN WITHOUT CVS: Submit the job using the command “sendn its.cui <jobname>”. This script prompts you for the following three pieces of information:

1. CROSS SECTIONS: First, it requests the name of the cross section file <cross> to be used. For the continuous-energy codes, the script looks for the file at “$HOME/cross3/<cross>”. For the multigroup codes, the script looks for the file at “$HOME/crossm/<cross>.11”.

2. LOCAL COMPILE: Second, sendn requests the location of a copy of ITS that can be used for making the executable. Thus, you may use a version of the code that you have checked out and modified. You must give a complete pathname for the base ITS directory (e.g., /scratch/temporary/its). No files will be deleted from the make directory as a result of the calculation, but some files may be modified if requested in the diffs section of the cui file. Any new files that have been included in the directory (other than through the diffs section of the cui file) will not appear in the job file, but they may be used if the Makefile.in has been so modified (in which case, the jobfile will show the reference to the new file as a difference in the Makefile.in). Attempts to apply diffs in a directory where the diffs have already been applied for a previous calculation will result in an error.

3. DIRECTORYDIFF: Next, the script requests a local directory with
which to perform a diff. The diff command will be used to compare the two directories and all subdirectories. The job file will not record the version number of either directory, therefore the user may not have enough information in the job file to duplicate a calculation unless the diff directory has a name corresponding to the tag used in the cvs checkout of the code. The directory name appears in the job file.

SENDN WITH CVS: Submit the job using the command “sendn its.cui <jobname>”. This script prompts you for the following 3 pieces of information (and possibly the 4th depending upon your responses to the first 3):

1. CROSS SECTIONS: First, it requests the name of the cross section file <cross> to be used. For the continuous-energy codes, the script looks for the file at “$HOME/cross3/<cross>”. For the multigroup codes, the script looks for the file at “$HOME/crossm/<cross>.11”.
2. LOCAL COMPILE: Second, sendn requests the location of a checked-out copy of ITS that can be used for making the executable. Thus, you may use a version of the code that you have checked out and modified. You must give a complete pathname for the base ITS directory (e.g., /scratch/temporary/its). No files will be deleted from the make directory as a result of the calculation, but some files may be modified if requested in the diffs section of the cui file. Any new files that have been included in the directory (other than through the diffs section of the cui file) will not appear in the job file, but they may be used if the Makefile.in has been so modified (in which case, the job file will show the reference to the new file as a difference in the Makefile.in). Attempts to apply diffs in a directory where the diffs have already been applied for a previous calculation will result in an error.

CVS COMPILE: To request a cvs checkout of the code, you may respond to this request with “none” (or press “Enter”). The cvs checkout will be performed in the $HOME/tmp/<jobname> directory and should not affect any other versions of the code.

3. CVS DIFF for LOCAL COMPILE: Enter “none” or simply press “Enter” with no input for this third request. VERSION for CVS

    COMPILe: If your response to the second request was “none”, the script
requests the version for a cvs checkout. The command will be issued as
“cvs checkout <options> its/Code”. The response to this request will be
used for the <options> and any valid cvs flags may be used that do not
contain a slash, “/”. Examples of valid syntax for responses are: -D now,
-D “March 28, 2001”, -D “3 hours ago”, -D “2 fortnights ago”, -r
ITSversion5.0, etc. Another valid response is to simply press “Enter”
with no input, which will result in the checkout of the most recent
version of the code.

4.CVSDIFF for LOCAL COMPILE: If you gave a pathname for making the
executable but not for a directory diff, the script requests the version for a
cvs diff. The syntax for responses to this request are the same as for
specifying the cvs version for a checkout. The directory in which the
executable is made will be compared with the repository using the cvs
diff command. This version (that will appear in the job file) and the
results of the cvs diff (that will also appear in the job file) can be
used to reproduce a calculation.

CVS UPDATE/DIFF for CVS COMPILE: If you did not give a pathname
for making the executable, the script requests a version for a cvs update
and diff. In this case, the script will
checkout a version of the code using the options in the third response, apply the diff
from the cui file, attempt to update to the version of the code
specified in this response, and then compare the resulting code to the
repository using the cvs diff command with the options specified in this
response. The version requested here (that will appear in the job file) and the results of the cvs diff (that will also appear in the job file) can be
used to reproduce a calculation.

For either of these, if no option is specified for the cvs diff, “-D now” will
be used. The date and time of the submission of the calculation, which
are recorded in the job file, and the results of the cvs diff can be used to
reproduce the calculation at a later date.

EXECUTION: On platforms where the interactive script is used because
cross compiling is required, it may be necessary to move the files to perform
the calculation. However, it will be necessary to move the files back to their
original location after execution.

If execution fails or if after the run the output file is found to contain errors, it may be
desirable to use the tmp directory as a working directory to debug the code or
calculation. However, after the calculation has been performed successfully, it is
likely that one must resubmit the job with corrected input so that the job file will
accurately reflect the code modifications and the input used to perform the
calculation.

POSTPROC: When the interactive script is used, postproc must be
executed to produce a job file. The job file will be placed in the directory from
which the job was originally launched, just as with the non-interactive script.

EVALUATE RESULTS: Errors may occur at a number of stages in the
calculation. The stage at which the error occurred is usually indicated near the
start of the ohoh file. Some common causes of errors are:
1 Configure - If the platform has not been used before, the necessary config files may not be present.

2 Make - If the definitions have not been properly specified or `Template.mk` has been modified, the definitions may not have been properly set by the scripts. The `Template.mk` must contain certain words to be replaced by the scripts.

5 Running

ITS

3. Execution - If an error statement is generated (search in the output for “<<<<”), then an anticipated problem has been found in the input and/or cross sections, and diagnostic information should be available. Such an error might also result if the `Template.mk` was modified, and an executable with the wrong code options is being used. If the code generates an execution error, the user may have introduced a bug via code modification. If `abug` is found in ITS that was not introduced by the user, please notify the ITS developers providing enough details to reproduce and understand the error. Preferably, this would include a description of the error observed, the version of the code being used, diffs showing any code modifications made, and the input and output files demonstrating the bug.

5.5 Platform Dependencies

The following is a list of platforms on which ITS has been successfully built and tested. Following the name of the platform is the system type and operating system.

Red Storm (Catamount)

The driver script is functional for building the executable on a compile node in interactive mode. Jobs are submitted to the queue using the `qsub` command. If not using the scripts, the `configure` command must specify Red Storm as the host platform:
configure -host=i386-redstorm
ACIS12 is available for CAD calculations.

Thunderbird (Linux)

The driver script is functional for building the executable on a compile node in interactive mode. Jobs are submitted to the queue using the qsub command. If not using the scripts, the configure command must specify Thunderbird as the host plat

```bash
configure -host=i386-redstorm
```
Purple or Up (AIX)

The driver script is functional for building the executable on a compile node in interactive mode.
Jobs are submitted to the queue using the psub command.

```
psub -ln <nodes> -g <processors> -tM <max time> its.com
```
If not using the scripts, the configure command must specify Purple as the host platform:
```
configure -host=i386-purple
```

ACIS15 is available for CAD calculations. The ACISR15 libraries were provided by Spatial (rather than built compiling the source code on purple).

QA,QB(HP Alpha,Tru64 OSF1V5)

The driver script is functional for building the executable in interactive mode. ACIS12 libraries are available for CAD. The ITS C++ CAD interface code should be compiled with the

5
Running
debug flag. Jobs are submitted to the queuing system using the bsub command. The ACIS library path must be set in the config/mt-osf5 file for running on QSC or QT.

- Crater (AMD Athlon, Linux) The driver script is functional in all modes. With MPI, the script is set to use 2 processors. The mpirun command should be used to specify the number of processors for parallel calculations.
- Pegasus (64-bit Intel, Linux) The driver script is functional in all modes. With MPI, the script is set to use 2 processors.

The mpirun command should be used to specify the number of processors for parallel calculations.

PC

- See the Installation section for guidance on installing ITS on a PC.

6 Code Options

Numerous code options can be selected for compiling the ITS codes. These options have been implemented as preprocessor definitions. It is necessary to choose between the multigroup (MITS) codes and the continuous-energy codes. (The selection between forward and adjoint mode in the MITS codes is made as an input option.) Other options include choosing between the 1-, 2-, and 3-dimensional geometry representations and choosing
a random number generator.

The code options must be specified in either the CUI file (if using the scripts to execute the code) or in the Defines.mk file (if building an executable). Refer to the section on Running ITS for more details on applying definitions. Abbreviated versions of these definitions are used in the CUI and Defines.mk.

6.1 Preprocessor Definitions

**ITS MULTIGROUP** (to request the multigroup codes; omitted for continuous energy codes)

Select the code as one of:

- ITS -TIGER (1-D)
- ITS CYLTRAN (2-D cylindrical geometry; 3-D transport)
- ITS ACCEPT (3-D)

Select the random number generator as one of:

- ITS RNG1 (generator in 3.0, only available in serial)
- ITS RNG2 (RANMAR)
- ITS RNG3 (MersenneTwister, only available in development version)

To select to run on a parallel platform:

- ITS USE MPI

Other available options are:

- **ITS PCODES** (more ionization and relaxation for continuous energy codes)
- **ITS USE PFF** (pffformatted output)
- **ITS CAD** (for linking with Cholla or ACIS CAD geometry)
- **ITS USE CHOLLA** (for facet geometry)
- **ITS USE ACIS12** (the path for ACIS libraries is specified in the config/mt-* file)

6.2 Definition Requirements

- **ITS RNG1**, **ITS RNG2**, or **ITS RNG3** must be selected as the random number generator.
- **ITS TIGER**, **ITS CYLTRAN**, **ITS ACCEPT**, or **ITS CAD** must be selected as the code.
- **ITS CAD** must be accompanied by **ITS ACCEPT** (automatic in the makefile).
- **ITS CYLTRAN** is not currently functional for **ITS MULTIGROUP**.
- **ITS RNG1** cannot be used with **ITS MPI**.
- **ITS USE PFF** cannot be used with **ITS MULTIGROUP**.
- **ITS CAD** must be used with **ITS USE ACIS12** or **ITS USE CHOLLA**.
7 Summary of ITS Keywords

This section contains listings of keywords relevant to the ITS (continuous-energy) codes, the MITS codes in forward mode, and the MITS codes in adjoint mode in Tables 4, 5, and 6, respectively. Only those keywords applicable to the code and mode are listed in each table. Many keywords in forward mode do not apply to adjoint mode and vice versa. The keywords are listed approximately in order of importance. In addition, for each keyword the default code behavior is listed. The default behavior will be employed by the code if the keyword is not found in the input deck.

The secondary keywords used for biasing are listed in Table 7. These are secondary keywords for the BIASING keyword. The sub-keyword, limitations on the code and mode with which the sub-keyword can be used, and the default code behavior are listed in the table. In addition, the global and zone-dependent features of the sub-keywords are listed.

More detailed descriptions of the syntax, sub-keywords, and use of these keywords are contained in the Keywords for ITS section. In some cases, these keywords or their defaults depend upon the code option (preprocessor definition) beyond the choice of MITS or ITS. A listing of the available preprocessor definitions is contained in the section on Code Options.

Table 4. ITS keywords and default settings

<table>
<thead>
<tr>
<th>KEYWORD</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>**** GEOMETRY ****</td>
<td>required</td>
</tr>
<tr>
<td>GEOMETRY</td>
<td>required</td>
</tr>
<tr>
<td>**** SOURCE ****</td>
<td></td>
</tr>
<tr>
<td>ELECTRONS or PHOTONS</td>
<td>electron source</td>
</tr>
<tr>
<td>ENERGY or SPECTRUM</td>
<td>1.0 MeV monoenergetic</td>
</tr>
<tr>
<td>POSITION</td>
<td>point source at origin (TIGER and ACCEPT) on axis at minimum-z (CYLTRAN)</td>
</tr>
<tr>
<td>DIRECTION</td>
<td>monodirectional source in positive-z direction</td>
</tr>
<tr>
<td>CUTOFFS</td>
<td>electrons: 5% of maximum; photons: 0.01 MeV</td>
</tr>
<tr>
<td>**** OUTPUT OPTIONS ****</td>
<td></td>
</tr>
<tr>
<td>ELECTRON-EMISSION</td>
<td>off</td>
</tr>
<tr>
<td>ELECTRON-ESCAPE, PHOTON-ESCAPE</td>
<td>off</td>
</tr>
<tr>
<td>ELECTRON-FLUX, PHOTON-FLUX</td>
<td>off</td>
</tr>
<tr>
<td>Keyword</td>
<td>Value</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>PULSE-HEIGHT</td>
<td>off</td>
</tr>
<tr>
<td>ESCAPE-SURFACES</td>
<td>all escape surfaces</td>
</tr>
<tr>
<td>PLOTS (CYLTRAN and ACCEPT Only)</td>
<td>off</td>
</tr>
<tr>
<td>**** COMMONLY USED OPTIONS ****</td>
<td></td>
</tr>
<tr>
<td>TITLE</td>
<td>blank title</td>
</tr>
<tr>
<td>HISTORIES or HISTORIES-PER-BATCH</td>
<td>1000 histories</td>
</tr>
<tr>
<td>BATCHES</td>
<td>20 batches</td>
</tr>
<tr>
<td>BIASING</td>
<td>no biasing parameters are activated</td>
</tr>
<tr>
<td>EBFIELDS</td>
<td>no fields</td>
</tr>
</tbody>
</table>

7. Summary of ITS Keywords

Table 4 (continued).

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>**** RARELY USED OPTIONS ****</td>
<td></td>
</tr>
<tr>
<td>DYNAMIC-MPI (MPI Only)</td>
<td>processor load balancing</td>
</tr>
<tr>
<td>TASKS (MPI Only)</td>
<td>number of processors available</td>
</tr>
<tr>
<td>RESTART and/or DUMP</td>
<td>no restart and no restart dump</td>
</tr>
<tr>
<td>PRINT-ALL or NO-INTERMEDIATE-OUTPUT or NO-DEPOSITION-OUTPUT or NO-OVERLAP-OUTPUT or OVERLAP-OUTPUT-MAX or NO-SZDEPOSITION-OUTPUT or NO-GEOMETRY-TABLE</td>
<td>final batch in output; intermediate in fort.12 all results data written to output</td>
</tr>
<tr>
<td>NO-DEPOSITION or NO-DETAILED-DEPOSITION</td>
<td>detailed dose and charge deposition</td>
</tr>
<tr>
<td>RANDOM-NUMBER</td>
<td>0 (converted to 519)</td>
</tr>
<tr>
<td>NEW-DATA-SET</td>
<td>1 run</td>
</tr>
<tr>
<td>INCLUDE-FILE</td>
<td>all data in single input file</td>
</tr>
<tr>
<td>FILE-NAMES</td>
<td>default names (fort.3, fort.11, etc.)</td>
</tr>
<tr>
<td>FINITE-ELEMENT-FORMAT (ACCEPT Only)</td>
<td>no fort.3 output</td>
</tr>
<tr>
<td>PFF-FORMAT (ITS PFF Only)</td>
<td>no pfoutput</td>
</tr>
<tr>
<td>REFLECTION-ZONE (ACCEPT Only)</td>
<td>no reflection zone</td>
</tr>
<tr>
<td>DEPOSITION-UNITS (ACCEPT Only)</td>
<td>dose in MeV per source particle charge deposition in electrons per source particle</td>
</tr>
<tr>
<td>ECHO</td>
<td>on</td>
</tr>
<tr>
<td>BELOW-1KEV</td>
<td>off</td>
</tr>
</tbody>
</table>
LINE-TALLY-WITH-CONTINUUM or ANNIHILATION-LINE-...

DUMP-CONNECTIVITY (ACCEPT Only) off
READ-CONNECTIVITY (ACCEPT Only) off
CODEZONES-PER-PATH (ACCEPT Only) 20
TRAJECTORY-POINTS (CYLTRAN and ACCEPT Only) 500
PLOTS-OVAL-RESOLUTION(ACCEPT Only) 360

**** DEVELOPMENT USE ONLY ****
CUTOFF-PHOTONS-ESCAPE energy of cutoffphotons is deposited locally
DOPPLER No Doppler broadening
NO-COHERENT photon coherent scattering is simulated
NO-INCOH-BINDING binding effects in incoherent scattering are included
NO-KICKING terminal processing includes kicking
NO-KNOCKONS secondary knock-on electrons
NO-STRAGGLING energy-loss straggling
RESTART-HISTORY no restart
SIMPLE-BREMS more accurate bremsstrahlung distributions

7.
Summary of ITS Keywords

Table 5. MITS forward keywords and default settings

<table>
<thead>
<tr>
<th>KEYWORD</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>**** GEOMETRY ****</td>
<td></td>
</tr>
<tr>
<td>GEOMETRY</td>
<td>required</td>
</tr>
<tr>
<td>**** SOURCE ****</td>
<td></td>
</tr>
<tr>
<td>ELECTRONS or PHOTONS</td>
<td>electron source</td>
</tr>
<tr>
<td>ENERGY or SPECTRUM</td>
<td>mono-group source in the highest-energy group</td>
</tr>
</tbody>
</table>
POSITION | point source at origin
DIRECTION | monodirectional source in positive-z direction
CUTOFFS | bottom of the lowest-energy group for each species

**** OUTPUT OPTIONS ****
ELECTRON-ESCAPE | off
ELECTRON-FLUX | off
PHOTON-ESCAPE | off
PHOTON-FLUX | off
ESCAPE-SURFACES | all escape surfaces
PLOTS (CYLTRAN and ACCEPT Only) | off

**** COMMONLY USED OPTIONS ****
TITLE | blank title
HISTORIES or HISTORIES-PER-BATCH | 1000 histories
BATCHES | 20 batches
BIASING | no biasing parameters are activated

7. Summary of ITS Keywords Table 5 (continued).

<table>
<thead>
<tr>
<th>KEYWORD</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>**** RARELY USED OPTIONS ****</td>
<td></td>
</tr>
<tr>
<td>TASKS (MPI Only)</td>
<td>number of processors available</td>
</tr>
<tr>
<td>DYNAMIC-MPI (MPI Only)</td>
<td>processor load balancing</td>
</tr>
<tr>
<td>RESTART and/or DUMP</td>
<td>no restart and no restart dump</td>
</tr>
<tr>
<td>PRINT-ALL or NO-INTERMEDIATE-OUTPUT or NO-DEPOSITION-OUTPUT or NO-OVERLAP-OUTPUT or OVERLAP-OUTPUT-MAX or NO-SZDEPOSITION-OUTPUT or NO-GEOMETRY-TABLE</td>
<td>final batch in output; intermediate in fort.12 all results data written to output</td>
</tr>
</tbody>
</table>
NO-DEPOSITION or NO-DETAILED-DEPOSITION: detailed dose and charge deposition

RANDOM-NUMBER: 0 (converted to 519)
NEW-DATA-SET: 1 run
MICRO: deposition calculated via flux-folding
INCLUDE-FILE: all data in single input file
FILE-NAMEs: default names (fort.3, fort.11, etc.)
FINITE-ELEMENT-FORMAT (ACCEPT Only): no fort.3 output
REFLECTION-ZONE (ACCEPT Only): no reflection zone
DEPOSITION-UNITS (ACCEPT Only): dose in MeV per source particle charge deposition in electrons per source particle

ECHO: on
LINE-TALLY-WITH-CONTINUUM or ANNIHILATION-LINE-... line radiation is tallied separately for photon escape and flux
DUMP-CONNECTIVITY (ACCEPT Only): off
READ-CONNECTIVITY (ACCEPT Only): off
CODEZONES-PER-PATH (ACCEPT Only): 20
PLOTS-OVAL-RESOLUTION (ACCEPT Only): 360

**** DEVELOPMENT USE ONLY ****
CUTOFF-PHOTONS-ESCAPE: energy of cutoff photons is deposited locally
RESTART-HISTORY: no restart

7. Summary of ITS Keywor ds

**Table 6.** MITS adjoint keywords and default settings

<table>
<thead>
<tr>
<th>KEYWORD</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADJOINT</td>
<td>forward</td>
</tr>
<tr>
<td>**** GEOMETRY ****</td>
<td></td>
</tr>
<tr>
<td>GEOMETRY</td>
<td>required</td>
</tr>
<tr>
<td>**** DETECTOR ****</td>
<td></td>
</tr>
<tr>
<td>DETECTOR-RESPONSE</td>
<td>required</td>
</tr>
<tr>
<td>**** SOURCE OUTPUT OPTIONS ****</td>
<td></td>
</tr>
<tr>
<td>SOURCE-SURFACES</td>
<td>all escape surfaces</td>
</tr>
<tr>
<td>SPECTRUM</td>
<td>only a flat forwardspectrum is used</td>
</tr>
<tr>
<td>ELECTRON-SURFACE-SOURCE</td>
<td>off</td>
</tr>
<tr>
<td>ELECTRON-VOLUME-SOURCE</td>
<td>off</td>
</tr>
<tr>
<td>PHOTON-SURFACE-SOURCE</td>
<td>off</td>
</tr>
</tbody>
</table>
PHOTON-VOLUME-SOURCE off

**** COMMONLY USED OPTIONS ****

<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE</td>
<td>blank title</td>
</tr>
<tr>
<td>HISTORIES or HISTORIES-PER-BATCH</td>
<td>1000 histories</td>
</tr>
<tr>
<td>BATCHES</td>
<td>20 batches</td>
</tr>
<tr>
<td>BIASING</td>
<td>no biasing parameters are activated</td>
</tr>
<tr>
<td>PLOTS (CYLTRAN and ACCEPT Only)</td>
<td>off</td>
</tr>
</tbody>
</table>

**** RARELY USED OPTIONS ****

<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TASKS (MPI Only)</td>
<td>number of processors available</td>
</tr>
<tr>
<td>DYNAMIC-MPI (MPI Only)</td>
<td>processor load balancing</td>
</tr>
<tr>
<td>CUTOFFS</td>
<td>top of the highest-energy group for each species</td>
</tr>
<tr>
<td>RESTART and/or DUMP</td>
<td>no restart and no restart dump</td>
</tr>
<tr>
<td>PRINT-ALL or NO-INTERMEDIATE-OUTPUT or NO-OVERLAP-OUTPUT or OVERLAP-OUTPUT-MAX or NO-GEOMETRY-TABLE</td>
<td>final batch in output, intermediate in fort.12 all results data written to output</td>
</tr>
<tr>
<td>RANDOM-NUMBER</td>
<td>0 (converted to 519)</td>
</tr>
<tr>
<td>NEW-DATA-SET</td>
<td>1 run</td>
</tr>
<tr>
<td>INCLUDE-FILE</td>
<td>all data in single input file</td>
</tr>
<tr>
<td>FILE-NAMES</td>
<td>default names (fort.11, fort.12, etc.)</td>
</tr>
<tr>
<td>REFLECTION-ZONE (ACCEPT Only)</td>
<td>no reflection zone</td>
</tr>
<tr>
<td>AREAL-DENSITY-OUTPUT (ACCEPT Only)</td>
<td>no areal density file</td>
</tr>
<tr>
<td>ECHO</td>
<td>on</td>
</tr>
<tr>
<td>CODEZONES-PER-PATH (ACCEPT Only)</td>
<td>20</td>
</tr>
<tr>
<td>PLOTS-OVAL-RESOLUTION (ACCEPT Only)</td>
<td>360</td>
</tr>
</tbody>
</table>

**** DEVELOPMENT USE ONLY ****

<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESTART-HISTORY</td>
<td>no restart</td>
</tr>
</tbody>
</table>
### Table 7. Biasing sub-keywords, default settings, and properties

<table>
<thead>
<tr>
<th>KEYWORD</th>
<th>DEFAULT</th>
<th>Global/Local</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIAS-SHELL (PCODES Only)</td>
<td>natural line radiation</td>
<td>global</td>
</tr>
<tr>
<td>COLLISION-FORCING</td>
<td>natural photon interactions</td>
<td>no global settings; specified by zone</td>
</tr>
<tr>
<td>ELECTRON-RR (ForwardOnly)</td>
<td>natural number of photon-produced secondary electrons are followed</td>
<td>global Russian Roulette probability; activated by zone</td>
</tr>
<tr>
<td>NEXT-EVENT-ESCAPE</td>
<td>off, unless using PHOTON-ESCAPE or PHOTON-SURFACE-SOURCE</td>
<td>global</td>
</tr>
<tr>
<td>PHOTTRAN (ForwardOnly)</td>
<td>secondary electrons are tracked</td>
<td>no secondary electrons set globally; exceptions by zone</td>
</tr>
<tr>
<td>SCALE-BREMS (ITS Only)</td>
<td>natural bremsstrahlung production</td>
<td>global scaling factor; activated by zone (activates SCALE-IMPACT)</td>
</tr>
<tr>
<td>SCALE-EP (MITS Only)</td>
<td>natural electron-to-photon production</td>
<td>global scaling factor; activated by zone</td>
</tr>
<tr>
<td>SCALE-IMPACT (ITS Only)</td>
<td>20% of brems scaling if used; otherwise, natural impact ionization</td>
<td>global scaling factor; activated in zones with SCALE-BREMS</td>
</tr>
<tr>
<td>SCALE-PE (MITS Only)</td>
<td>natural photon-to-electron production</td>
<td>global scaling factor; activated by zone</td>
</tr>
<tr>
<td>TRAP-ELECTRONS (ForwardOnly)</td>
<td>no trapping above cutoff energy</td>
<td>both global and local; the more stringent applies</td>
</tr>
</tbody>
</table>

**** DEVELOPMENT USE ONLY ****

| ELECTRAN (ForwardOnly)      | secondary and scattered photons are tracked                             | no secondary/scattered photons set globally; exceptions by zone |
| NO-BANK (MITS Only)         | secondary particles are banked                                          | global |

---

Keywords for ITS

8

1.

Tables
8 Keywords for ITS

The input keywords must be specified in either the CUI file (if using the scripts to execute the code) or in the input file (if executing the code manually). Refer to the documentation on Running ITS for more details on specifying the input file in the CUI file or otherwise.

8.1 Input Notation

The keywords that are appropriate to use depend upon the code options that have been selected in building the executable (and whether the MITS code is being run in forward or adjoint mode). An overview of the preprocessor definitions is available in the Code Options section. An overview of the keywords that apply to MITS forward, MITS adjoint, and the forwardcontinuousenergy ITS codes is available in the Summary of ITS Keywords section. In this section, following each keyword are any restrictions on the code options or mode. The designation "ITSOnly" refers to the continuous-energy codes (i.e., not MITS). The designation "ForwardOnly" refers to both the MITS and ITS codes, unless otherwise noted. All other designations refer to the preprocessor definitions used in building the executable.

Most primary keywords are order-independent. The two exceptions to this are the NEWDATA-SET keyword and the SUBZONE-ONLY usage of the GEOMETRY keyword.

Most keywords must be used once and not repeated in an input file. Exceptions are BIASING, ECHO, and NEW-DATA-SET. Most sub-keywords should be used only once per use of their primary keyword. Some exceptions are the sub-keywords of ESCAPE-SURFACES, SOURCESURFACES, and GEOMETRY.

Parameters are associated with the preceding keyword appearing on the same line. If parameters are omitted, default values may apply. Consideration should be given to the fact that in some situations this default value is invalid and will trigger an error. Values expected on lines following a keyword are not optional, unless otherwise stated.

Comments may be inserted in the input deck. Anything appearing to the right of an asterisk anywhere in the input deck will be treated as a comment and ignored by the code.

Input is not case sensitive, with only one significant exception. File names entered with the FILE-NAMES keyword will be used exactly as provided in the input deck. Character input provided with the TITLE and DEPOSITION-UNITS keywords will appear in the output file exactly as provided, but this input does not affect the performance of the code.

8.2 Keywords

1. ADJOINT (MITS Only)
Syntax: ADJOINT
Default: Forward

This keyword triggers adjoint transport. It can be used at any point in the input deck. Forward and adjoint runs can be mixed, but for adjoint calculations the adjoint specification must be made for each new-data-set.

2. KEYWORDS
for I T S

Adjoint mode requires three mandatory keywords: GEOMETRY, DETECTOR-RESPONSE, and a surface or volume source. The quantity of interest must be specified using the DETECTORRESPONSE keyword with one of the secondary keywords CHARGE, DOSE, ESCAPE or KERMA. The forward source(s) must be described using one or more of the source keywords with prefix PHOTON-or ELECTRON-and suffix SURFACE-SOURCE or VOLUMESOURCE. If a surface-source is requested, the user may further select surfaces using the SOURCE-SURFACES primary keyword. For ACCEPT, the default is to use, in a combined tally, all surfaces through which a particle enters the escape zone. For TIGER, the default is that both surfaces are specified.

2. ANNIHILATION-LINE-TALLY-WITH-CONTINUUM (Forward Only)

Syntax: ANNIHILATION-LINE-TALLY-WITH-CONTINUUM Default: Annihilation line radiation is tallied separate from the continuum for photon escape and flux.

This keyword triggers annihilation line radiation to be tallied in the
corresponding energy bin of the continuum radiation for photon escape and flux.

3. **AREAL-DENSITY-OUTPUT** *(Adjoint Raytrace Only)*

   **Syntax:** AREAL-DENSITY-OUTPUT
   
   **Default:** No areal density data file

   This keyword triggers the creation of a file containing areal density data for the rays in a raytrace calculation. It can be used at any point in the input deck. This keyword requires adjoint mode and that a raytrace calculation (see the DIRECTION-SPACE secondary keyword of the PHOTON-SURFACE-SOURCE keyword) is being performed. By default the file name is “fort.4” but the file name can be changed using the FILE-NAMES keyword.

   The format of the areal density data file is as follows. The first record contains the text “location:” followed by the x, y, and z coordinates of the point where the rays originate. Each record after the first contains the angle number, the polar angle (degrees) measured from the positive z axis, the azimuthal angle measured from the positive x axis, and the total areal density *(g/cm²)* of all material encountered by the ray until escape from the geometry.

4. **BATCHES**

   **Syntax:** BATCHES [parameter(1)]
   
   **Example:** BATCHES 100
   
   **Default:** 20 batches

   Number of batches of primary particles to be run. [parameter(1)] batches are performed in order to obtain estimates of statistical uncertainties. Each batch contains an equal number of source particles selected either by the HISTORIES keyword or by the HISTORIESPER-BATCH keyword. Accuracy of the estimates degrades substantially for fewer than 10 batches. Although increasing the number of batches improves this accuracy, it also increases the overhead (run time). We recommend that at least 20 batches be used.

5. **BELOW-1KEV** *(PCODES Only)*

8. Keywords for ITS

   **Syntax:** BELOW-1KEV PHOTONS [parameter(1)]
   
   **Example:** BELOW-1KEV PHOTONS 1.0E-3
   
   **Default:** No cross sections below 1 keV. No LLNL cross sections used.
**WARNING:** This is a limited capability! Read some of the limitations below.

This keyword indicates that photon energies below 1 keV may be included in the calculations. \[\text{parameter(1)}\] is the transition energy (in MeV) from the XGEN cross-section data to the LLNL EPDL97 cross-section data set. To use this feature, cross sections must be generated in XGEN using the BELOW-1KEV keyword. The new capability for attenuation of source photons below 1 keV requires the Livermore evaluated data library for photons, which is not distributed with ITS. It is available for download from the IAEA website at http://www-nds.iaea.org/epdl97/. ITS uses the EPDL97 in ENDF/B-VI format. This file should be located in the XGEN distribution at \text{.../Code/XSdata/epdl97.all}.

Only photon absorption is modeled using the LLNL data. Thus, absorption is the only photon physics that will be modeled below the transition energy.

Only source particles will be generated below 1 keV (i.e., there is no scattering to or photon production below 1 keV).

Cross sections below 1 keV contain high levels of uncertainty. Users are encouraged to consider how these cross section uncertainties may be amplified to even greater uncertainty in their Monte Carlo results.

6. **BIASING**

Syntax: BIASING Default: No biasing parameters are activated.

Photons, electrons, and positrons are followed globally (if data is included in the cross sections).

Electron trapping is performed at the electron cutoff energy.

Selectively turns on the input-zone-dependent bias parameters specified (Photon Collision Forcing, Photran, Russian Roulette, Scale Electron-to-Photon Interactions, Scale Photon-to-Electron Interactions, and/or Electron Trapping). This primary
keyword may be repeated,
but some secondary keywords should not be repeated.

Restrictions: ELECTRON-RR is disallowed in adjoint. The user cannot simultaneously spec
ify SCALE-EP and SCALE-PE in the same input zone.
Note that SCALE-EP and SCALE-PE always refer to scaling the forward
cross section, even
in adjoint. Thus, when using SCALE-PE in adjoint, an electron-adjuncton
would be more
likely to turn into a photon-adjuncton.

(a) BIAS-SHELL *(PCODES Only)*
Syntax: BIAS-SHELL

Example: BIAS-SHELL
Default: Samples line radiation based on the actual probability.

This keyword activates a biasing option that samples line radiation
uniformly among all possible lines. Photon weights are adjusted based on
the actual probability of sampling the given line radiation.
Using this biasing option can make PCODES more efficient for the calculation of flux or escape of line radiation. However, this option can make PCODES less efficient for the calculation of quantities like energy
deposition, where the importance of line radiation is likely to be proportional to the actual probability.

(b) **COLLISION-FORCING** Syntax:

```
COLLISION-FORCING
[parameter(1)]
Example: COLLISION-FORCING5
14-68
```

0.3 0.2 0.1 0.5 0.1 15 Default:
Photon cross sections determine interaction probabilities. This keyword specifies the photon forced interaction probabilities. This keyword must be followed by two sets of \([\text{parameter}(1)]\) numbers. The first set contains the input zones for which forced interaction probabilities are to be specified. The second set specifies the forced probabilities for the corresponding input zones. This keyword may be repeated, but prior settings may be overwritten. For each zone the last setting in the input deck will be used.

(c) **ELECTRAN** *(Forward Only)* Syntax:

```
ELECTRAN Example: ELECTRAN 1, 4-6, 8Default: Secondary and scattered photons are followed. If photon cross sections are not provided for MITS, then this keyword is unnecessary. This keyword indicates that electron-produced secondary photons are not to be tracked and scattered photons are not to be tracked. (Coherent scattering of photons is allowed; the NO-COHERENT keyword can be used to deactivate this physics.) Additional parameters are optional and specify exceptions. Zones in which secondary photons are to be tracked are listed beginning on the following line. A dash indicates that all zones between two numbers are included. Beginning the
list with a dash includes all zones from 1 to the indicated zone. Ending the list with a dash includes all zones from the last number given to the last zone number. Using ELECTRAN with a photon source provides the equivalent of a first-collision electron source. Using ELECTRAN with an electron source provides electron-only transport.

The ELECTRAN secondary keyword can be repeated.

(d) **ELECTRON-RR**

*(Forward Only)*

Syntax:

```
ELECTRON-RR [parameter(1)] [keyword]
```

Example:

```
ELECTRON-RR 0.1 CUSTOM -RR 14- 689
```

Default: Russian Roulette is not used. Natural photon-to-electron cross sections are used.
[parameter(1)] is the Russian Roulette survival probability used in determining the number of photon produced secondary electrons followed. If [parameter(1)] is omitted or 0.0, Russian Roulette will be used such that the natural number of electrons (the number produced if SCALE-BREMS or SCALE-EP had not been used) would be followed, if Russian Roulette and SCALE-BREMS or SCALE-EP were used throughout the problem. The keyword CUSTOM-RR indicates that customized Russian Roulette logic has been included by the user in function FLRRK. The additional parameters set the zones for which Russian Roulette is to be turned on. The list of input zones beginning on the following line specify the regions of the problem where Russian Roulette is to be used. The fraction of photon produced secondary electrons to be followed is the inverse of the scaled bremsstrahlung production. The ELECTRON-RR secondary keyword may be repeated, but [parameter(1)] will only be set to the value in the last occurrence of the keyword.

(e) NEXT-EVENT-ESCAPE

Syntax: NEXT-EVENT-ESCAPE
[keyword] Example:
NEXT-EVENT-ESCAPE OFF
Default: Feature is off, unless photon-escape or photon-surface-source is specified. With this keyword, a more efficient calculation of photon escape can be made. For differential escape scoring, this feature is automatically activated. The use of this keyword with the OFF sub-keyword can be used to deactivate next-event-escape.
logic when differential photon escape is requested.

(f) **NO-BANK** *(MITS Only)*

Syntax: NO-

BANK Default:

Secondary

particles are

banked and

relative

weights are

kept at unity.

With this

secondary

keyword,

secondary

particles

arenot

banked.

Rather, their

weights

are changed to account for multiplicity and absorption.WARN**ING:** Using this feature is strongly discouraged.

(g) **PHOTRAN** *(Forward Only)* Syntax:

PHOTRAN Example: PHOTRAN 1,

4-6, 8Default: Secondary electrons

are followed. If electron cross

sections are not provided for MITS,

then this keyword is unnecessary.

The keyword PHOTRAN indicates

that photon-produced secondary

electrons are not to be tracked.

Additional parameters are optional

and specify exceptions. The zones

in which secondary electrons are to

be tracked are listed beginning on

the following line. A dash indicates

that all zones between two

numbers are included. Beginning

the list with a dash includes all

zones from 1 to the indicated zone.

Ending the list with a dash includes all

zones from the last number given to

the last zone number. Using

PHOTRAN with a electron source is
allowed, since it only excludes tracking of secondary electrons. Using PHOTTRAN with a photon source can provide photon-only transport, if no exception zones are specified. The PHOTTRAN secondary keyword may be repeated.

8.

Keywords for ITS

(f) SCALE-BREMS (ITS Only) Syntax: SCALE-BREMS [parameter(1)]
[parameter(2)] Example: SCALE-BREMS 500.02 4-6,9 Default: Natural bremsstrahlung cross sections are used. [parameter(1)] is a scale factor used to modify bremsstrahlung production so as to increase the photon population without increasing the number of primary histories. For example, if [parameter(1)] is set equal to two, then there will be twice as much bremsstrahlung photon production. The ELECTRON-RR secondary keyword can be used to control the number of secondary electrons generated by this increased population of photons. [parameter(2)]
is the index of the material, according to the order in which the materials are read from the cross section file, on which the impact ionization scaling is based if SCALE-IMPACT is not used. The default is material number 1. The additional parameters beginning on the following line specify those zones in which bremsstrahlung biasing is activated.

The SCALE-BREMS secondary keyword may be repeated, but [parameter(1)] and [parameter(2)] will only be set to the value in the last occurrence of the keyword.

(i) **SCALE-EP** *(MIT Only)* Syntax: SCALE-EP [parameter(1)]
Example: SCALE-EP 2.0 4,5,6, 8-9 Default: Natural electron-to-photon cross sections are used. This keyword specifies the factor by which the electron-to-photon cross sections are to be scaled and in which input zones the scaled cross sections are to be used. [parameter(1)] is the scaling factor. The additional parameters beginning on the following line are the input zones in which the scaled cross sections are applied.

The SCALE-EP secondary keyword may be repeated, but [parameter(1)] will only be set to the value in the last occurrence of the keyword.

(j) **SCALE-IMPACT** *(ITS Only)* Syntax: SCALE-IMPACT [parameter(1)]
Example: SCALE-IMPACT 20.0 Default: Natural probability of electron impact ionization, except that if the SCALE-BREMS keyword has been used, then the scale factors for electron impact ionization will be based on the bremsstrahlung scaling (such that for an electron slowing from the maximum energy to the global electron cutoff energy for every five bremsstrahlung interactions there will be one electron impact ionization event in the material specified by [parameter(2)] of SCALE-BREMS). [parameter(1)] is used to scale electron impact ionization so as to increase the photon population (line radiation) without increasing the number of primary histories. The cross sections are scaled such that an electron slowing from the maximum energy to the global electron cutoff energy will, on the average, undergo a number of ionization events equal to [parameter(1)] in each material. The values by which the cross sections may be scaled in every material are written to output. The cross sections will never
be scaled down (such that fewer ionization events are simulated than with the natural cross sections).

WARNING: Impact ionization scaling is only activated in zones in which SCALE-BREMS is activated.

(k) **SCALE-PE (MITS Only)**

Syntax: SCALE-PE [parameter(1)]

Example: SCALE-PE 0.5 3-7

Default: Natural photon-to-electron cross sections are used. This keyword specifies the factor by which the photon-to-electron cross sections are to be scaled and in which input zones the scaled cross sections are to be used. [parameter(1)] is the scaling factor. The additional parameters are the input zones in which the scaled
cross sections are applied.
The SCALE-PE secondary keyword may be repeated, but [parameter(1)] will only be set to the value in the last occurrence of the keyword.

(i) TRAP-ELECTRONS *(Forward Only)*
Syntax: TRAP-ELECTRONS [parameter(1)] [parameter(2)]

ITS Example: TRAP-ELECTRONS 0.25 1-5
0.25 0.25 0.25 0.35 0.35
MITS Example: TRAP-ELECTRONS 425 14-68
40 35 40 35 35

Default: The trap-electron energy is the electron cutoff energy. For ITS, the global electron trapping cutoff specified by [parameter(1)] is energy in MeV. For MITS, the global electron trapping cutoff is the lower energy bound of the group specified by [parameter(1)]. This keyword must be followed by two sets of [parameter(2)] numbers. The first set contains the input zones for which electron trapping energies are to be specified. The second set specifies the energy below which trapping is tested for the corresponding input zones. In ITS the electron trapping energy is specified directly in MeV. In MITS the corresponding energy group is specified. (The lower energy group bound is used in forward, and the upper energy group bound is used in adjoint.) The TRAP-ELECTRONS secondary keyword may be repeated, but [parameter(1)] and [parameter(2)] will only be set to the value in the last occurrence of the keyword. Electron trapping is either ineffective or not implemented correctly yet for adjoint calculations. Electron trapping does not function with Cholla facet geometry. The code will always assume that an electron is not trapped if it is in a part that contains faceted geometry using the Cholla geometry library. The keyword CUSTOM-TRAP indicates that customized trapping logic has been included by the user in subroutine SAVE.

7. CODEZONES-PER-PATH *(ACCEPT Only)*
With this keyword, the user can increase the number of zones that a particle may pass through. A change is required only when specified by ITS output.

8. **CUTOFF-PHOTONS-ESCAPE** *(Forward Only)*

   Syntax: CUTOFF-PHOTONS-ESCAPE
   
   Defaults: Energy of photons below the cutoff energy is deposited locally.

   This keyword specifies that photons falling below the cutoff energy are assumed to escape from the problem. A diagnostic in the output states the average energy of photons per history that is assumed to have escaped from the problem.

   Use of the ELECTRAN biasing feature causes all electron-produced photons and all scattered photons (except for coherent scattering; see the NO-COHERENT keyword) to be considered below the cutoff energy.

9. **CUTOFFS**

   Syntax: CUTOFFS
   
   [parameter(1)] - [parameter(6)]
   
   Example: CUTOFFS
   
   0.01 0.12 * For ITS
   
   13
0.5 0.2

Example: CUTOFFS484950230 * For MITS

13
45 45
134
45 45 40

Defaults: In ITS the global electron cutoff energy equals 5% of the maximum source energy, and the global photon cutoff energy equals 0.01 MeV. In MITS forward mode, the cutoff group is the last group for each species. In MITS adjoint mode, the cutoff group is the first group for each species.

This keyword specifies the global cutoff energy for electrons [parameter(1)] and photons [parameter(2)] (and only in MITS, [parameter(3)] is reserved for another particle type). In MITS forward mode, the cutoff energy is the lower energy bound of the specified group. In MITS adjoint mode, the cutoff energy is the upper energy bound of the specified group.

This keyword can also be used to specify local cutoff groups. The numbers of input zones for which local cutoff groups are to be specified are given for electrons [parameter(3)] (or in MITS, electrons [parameter(4)], photons [parameter(5)], and a reserved data position). For each of these parameters that are non-zero, two sets of data must follow. The first set contains the input zones for which local cutoff groups are to be specified. The second set
specifies the local cutoff energy for the corresponding input zone. The more stringent of the local and global cutoffs will be used by the code. In MITSthe indices refer to the local group numbers as they were generated by CEPXS before they were reversed (for adjoint calculations) in the Monte Carlo. In forward mode, electrons which slow down below the lowest energy in the cutoff group are no longer transported and their energy and charge are locally deposited. Photons which downscatter below the cutoff group or which are absorbed (only in the default cutoff group) have their energy locally deposited and transport terminated. In adjoint mode, particles which speed up beyond the highest energy in the cutoff group will no longer be transported.

**WARNING:** The CUTOFFS keyword should be used with caution in adjoint mode.

10. **DEPOSITION-UNITS** *(ACCEPT Forward Only)*

    Syntax: DEPOSITION-UNITS [keyword] [keyword] [parameter(1)]
    Example:
    DEPOSITION-UNITS MASS SCALE 620.5 'MeV-cm²/g-ph' 'el-cm²/g-ph'
    Default: Dose in units of MeV per source particle and charge deposition in units of electrons per source particle.

This keyword modifies the default units of charge and energy deposition outputs for the ACCEPT codes. Three secondary keywords may be used with DEPOSITION-UNITS: **MASS**, **VOLUME**, and **SCALE**. MASS and VOLUME are mutually exclusive. If the MASS keyword is used, deposition values for each input zone will be divided by the mass of material in the zone in units of grams. If the VOLUME keyword is used, deposition values for each input zone will be divided by the volume of the zone in units of cm³. If the SCALE keyword is used (either separately or in addition to the MASS or VOLUME keyword), deposition values for each input zone will be multiplied by the scaling factor [parameter(1)]. The two character strings on the line following the keyword are the new units for energy and charge deposition that will be used only for labels in the output file. The character strings can be up to 15 characters long and should be enclosed in single quotation marks.

**WARNING:** Accurate MASS or VOLUME scaling depend on the accuracy of the volume data used. Internal calculation of zone volumes is not available for all subzoned bodies or body combinations. Refer to the GEOMETRY keyword for further information.

11. **DETECTOR-RESPONSE** *(Adjjoint Only)*
Syntax: DETECTOR-RESPONSE
Default: No default. User must specify a detector response.

This keyword is the means by which the user specifies what single quantity of interest (known as the detector response in the forward mode) is desired for the adjoint calculation.

(a) CHARGE Syntax: CHARGE
Default: There is no default quantity of interest in the adjoint calculation.

This keyword specifies charge deposition for the quantity of interest to be determined in an adjoint calculation. The following MATERIAL sub-keyword must be present.

i. MATERIAL
Syntax: MATERIAL [parameter(1)]
Example: MATERIAL5
Default: No default, this sub-keyword must be present.

This specifies the material in which charge deposition is calculated.

ii. LOCATION
Syntax: LOCATION
Default: Point charge deposition calculated at the origin.

See the POSITION keyword for secondary keywords. Normalization logic is only included for POINT and VOLUME distributions.

(b) DOSE Syntax: DOSE
Default: There is no default quantity of interest in the adjoint calculation.
This keywords specifies energy deposition for the quantity of interest to be
determined in an adjoint calculation. The following MATERIAL sub-
keyword must be present.

i. MATERIAL
   Syntax: MATERIAL [parameter(1)]
   Example: MATERIAL5
   Default: No default, this sub-keyword must be present.

   This specifies the material in which energy deposition is calculated.

ii. LOCATION
   Syntax: LOCATION
   Default: Point dose deposition calculated at the origin.

See the POSITION keyword for secondary keywords. Normalization
logic is only included for POINT and VOLUME distributions.

(c) ESCAPE Syntax: ESCAPE
   [keyword] Example:
   ESCAPE PHOTONS
   Default: There is no
default quantity of
interest in the adjoint
calculation. There is no
default source particle
type. This keyword
specifies particle escape
(or leakage) for the
quantity of interest to be
determined in an adjoint
calculation. The tertiary
keyword must be
present and be either
ELECTRONS or
PHOTONS for electron-
escape or photon-
escape, respectively.

The user can further specify the type of escaping quantity through the
following sub-keywords:

i. GROUP Syntax: GROUP [parameter(1)] Example: GROUP8 Default:
The particle escape is integrated over all energy groups.

This sub-keyword specifies the group index of the quantity of interest.
The order of the group structure is that produced by CEPXS before
the inversion which occurs in the Monte Carlo in adjoint mode.
ii. LOCATION
   Syntax: LOCATION
   Default: No default surface for particle escape.

   See the POSITION keyword for secondary keywords. Only the
   SURFACE keywords are functional. The sub-keyword
   SURFACE specifies the surface through which forward escape
   detector-response will be calculated.

iii. BINT Syntax: BINT [parameter(1)] [parameter(2)]
    Example: BINT 30.0 45.0 Default: The particle escape is
    integrated over lab angles from 0-90 degrees.

    The reference direction is LOCAL-NORMAL to the specified
    escape surface. The escape distribution is between angles
    given by [parameter(1)] and [parameter(2)] with
    defaults of 0 and 90 degrees, respectively. The escape direction
    distribution bin is based on a cosine-law to yield particle current.

(d) KERMA Syntax: KERMA
    Default: There is no
    default quantity of
    interest in the adjoint
    calculation. This
    keyword specifies
    KERMA (Kinetic
    Energy Released in
    MAterial) for the
    quantity of interest to
    be determined in an
adjoint calculation.

Operationally, the kerma dose is calculated from the photon flux. Photon-generated electrons are assumed locally deposited with a small correction for escaping bremsstrahlung. The following MATERIAL sub-keyword must be present.

i. **MATERIAL**
   Syntax: MATERIAL [parameter(1)]
   Example: MATERIAL5
   Default: No default, this sub-keyword must be present.

   This specifies the material in which KERMA is calculated.

ii. **LOCATION**
   Syntax: LOCATION
   Default: Point KERMA calculated at the origin.

   See the POSITION keyword for secondary keywords. Normalization logic is only included for POINT and VOLUME distributions.

12. **DIRECTION** (*Forward Only*)

   Syntax: DIRECTION [parameter(1)] [parameter(2)]
   Example: DIRECTION 90.0 90.0
   Default: Reference direction is positive-z direction.

   This keyword is used to define the source referenced direction and the distribution of particles

   in angle relative to the referenced direction (either isotropic or cosine-law). [parameter(1)] is the spherical polar angle \( \theta \), in degrees, and [parameter(2)] (CYLTRAN or ACCEPT Only) is the azimuthal angle \( \phi \) that define the reference direction.

   The meaning of source parameters specified with the DIRECTION and POSITION keywords is illustrated in Figure 1.
(a) **ISOTROPIC** Syntax: ISOTROPIC [parameter(1)] [parameter(2)]
Example: ISOTROPIC 10.0 45.0 Default: Monodirectional in the reference direction. Defines the distribution of source particles as isotropic between angles given by [parameter(1)] and [parameter(2)], with respect to the reference direction. The default values for [parameter(1)] and [parameter(2)] are 0 and 90 degrees, respectively.

(b) **COSINE-LAW** Syntax: COSINE-LAW[parameter(1)] [parameter(2)]
Example: COSINE-LAW Default: Monodirectional in the reference direction. Defines the distribution of source particles as proportional to the cosine of the angle.
with respect to the reference direction. The source is distributed between angles given by \([\text{parameter}(1)]\) and \([\text{parameter}(2)]\) with defaults of 0 and 90 degrees, respectively.

13. **DOPPLER** (*ITS Only*)

   Syntax: DOPPLER
   Default: Incoherent photon scattering will exclude Doppler broadening.

   This keyword causes incoherent photon scattering to be simulated with both binding effects and Doppler broadening. The use of this keyword is restricted to problems containing only single element materials. The effects of Doppler broadening are most profound at low energies; at photon energies above 2 MeV the keyword shows no significant effect when compared to the default incoherent photon scattering settings.

14. **DUMP**

   Syntax: DUMP
   Default: no dump

   If the DUMP keywords are present, a dump file will be written after each batch to “fort.10”. When a dump file is written, any existing file with the specified name will be overwritten. If the dump file is to be used for a subsequent restart (see primary keyword RESTART), it must be saved. See the FILE-NAMES keyword for specifying alternative names.
In serial, a dump file will be written after each batch. In parallel with static load-balancing, a dump file will be written after each cycle. In parallel with dynamic load-balancing enabled (see primary keyword DYNAMIC-MPI), a dump file will be written after each batch has been accumulated, but the frequency is controlled with the intermediate output setting on the TASKS keyword. A dump file is always written at the end of a calculation.

The code will always attempt to write a dump file before aborting a calculation, regardless of whether a dump file has been requested.

15. DUMP-CONNECTIVITY

Syntax: DUMP-CONNECTIVITY

Default: no connectivity dump

If the DUMP-CONNECTIVITY keyword is present, a connectivity dump file will be written after each batch to “fort.9”. When a dump file is written, any existing file with the specified name will be overwritten. If the dump file is to be used for a subsequent restart (see primary keyword READ-CONNECTIVITY), it must be saved. See the FILE-NAMES keyword for specifying alternative names.

In serial, a connectivity dump file will be written after each batch. In parallel with static load-balancing, a connectivity dump file will be written after each cycle, if the master process is performing batches of Monte Carlo work. In parallel with dynamic load-balancing, a connectivity dump file cannot be written. Since each process accumulates connectivity information in isolation and that information is not gathered to the master process, only the connectivity data learned by the master process can be written to the dump file.

16. DYNAMIC-MPI (MPI Only)

Syntax: DYNAMIC-MPI

Example: DYNAMIC-MPI

Default: Static load balancing is used.

This keyword applies only for parallel processing. With dynamic load balancing, the master process does not perform Monte Carlo calculations; instead, it reads input, coordinates the Monte Carlo work performed on other processes, and writes output. Some capabilities, such as RAYTRACE, DUMP-CONNECTIVITY, and particle trajectory PLOTS, are not allowed with dynamic load balancing.

17. EBFIELDS (ITS, CYLTRAN or ACCEPT, and non-PCODES Only)
Keywords for ITS

Syntax: EBFIELDS [keyword(1)] [parameter(1)] [INPUT-MODEL]
[parameter(2)] [parameter(3)] [keyword(2)] [parameter(4)]
Example1: EBFIELDS 1-TESLA * constant magnetic field of 1 Tesla along z axis
  MAGNETIC * activate magnetic field 1-7 * in input zones 1 through 7
Example 2: EBFIELDS CURRENT-EB * uniform axial electric field BOTH *
  activate magnetic and electric fields 2, 5 * in input zones 2 and 5
Example 3: EBFIELDS CUSTOM-B * custom magnetic field only, 3 (Z, B) pairs
  INPUT-MODEL
  0.0000 .5  * at 0 cm, field strength = 0.5 Tesla
  20.000 .6  * at 20 cm, field strength = 0.6 Tesla
  40.000 .7  * at 40 cm, field strength = 0.7 Tesla
  MAGNETIC * activate magnetic field
  1-7  * in input zones 1 through 7
  OFF  * turn off field
  6   * in zone 6
Example 4: EBFIELDS B-3D-MESH * magnetic field on uniform mesh
  30 10 100 * 30 bins in x, 10 bins in y, 100 bins in z
  -1.0, -1.0, -1.0 1.0, 1.0, 1.0 * min and max x, y, z values
  1.24E-01, -1.24E-01, 5.20E+00 * values of the magnetic field vectors
  ... *(3 values on each of 31 x 11 x 101 lines) MAGNETIC * activate
magnetic field only 1-* in all input zones
Example5: EBFIELDS USER6 * user defined/coded field
description Default: Fieldsoffin all zones; constant magnetic fieldof1Teslaalongthe z-axis; no electric field.

This keyword offers the user choices for specifying electric and magnetic fields effects and allows the user to set which input zones shall have these effects computed.
Fields are no longer activated in each zone through the GEOMETRY input on the zone-specification or material-specification, for CYLTRAN and ACCEPT, respectively.

The value of [keyword(1)] determines the field model for the electric/magnetic fields. It can be one of 1-TESLA, CURRENT-EB, CUSTOM-B, B-3D-MESH, or USER. 1-TESLA is the default.

The value of [keyword(2)] can be one of MAGNETIC, BOTH, or OFF. The effect of these is to activate the magnetic fields, activate
both electric and magnetic fields, or deactivate fields, respectively, in the zones specified by the list \[[\text{parameter}(4)]\] beginning on the following line.

For **1-TESLA**, there is no electric field and a constant magnetic field of 1 Tesla along the z-axis. Using 'BOTH' for input zones still only enables a magnetic field.

For **CURRENT-EB**, there is a uniform axial electric field of 1 MV/cm and an external magnetic field as if it had been generated by an infinitely long, uniform-current-density beam with radius of 0.15 cm and total current (which would be along the z-axis) of 150 kAmps.

For **CUSTOM-B**, there is no electric field and a magnetic field specified in the following manner. **INPUT-MODEL** must appear on a subsequent line, followed by \(Z, B\) pairs. On each line \[[\text{parameter}(2)\] and \[[\text{parameter}(3)]\] contain distance along the z-axis and the total magnetic field (in Tesla) at that \(z\)-value, respectively. The \(z\) values must be strictly increasing. For particles outside the range of \(z\) values, the magnetic field closest to that position will be used. For particles within the \(z\)-grid, an interpolation based on linear field lines is used (i.e., values of radii such that the product of the enclosed area times the total magnetic field is constant). The azimuthal magnetic field is zero, and the radial and axial components are determined such that the mathematical divergence of the magnetic field is zero. The user should NOT specify a value of zero for the total magnetic field while using **CUSTOM-B**.

For **B-3D-MESH**, the values of the magnetic field are specified on a regular, uniform rectangular mesh. With this option, the next line contains 3 integers, which represent the number of uniform bins in the \(x\)-, \(y\)-, and \(z\)-directions, respectively (to be referred to as \(\text{max}I\), \(\text{max}J\) and \(\text{max}K\), respectively). The next line contains 6 values corresponding to the minimum and maximum extents of the mesh in the Cartesian coordinate axes (i.e., minimum-\(x\), minimum-\(y\), minimum-\(z\), maximum-\(x\), maximum-\(y\), and maximum-\(z\)). This should encompass all space in which the field may be turned on. Following this are \((\text{max}I+1)\times(\text{max}J+1)\times(\text{max}K+1)\) lines. On each line, the first three values are the \(x\)-, \(y\)-, and \(z\)-components of the magnetic field (in Tesla). Any other values on the line are ignored. The order of the values should advance from minimal to maximum Cartesian coordinates, advancing first in \(x\), then in \(y\), and then in \(z\). This is illustrated by the following do-loop structure, which might produce the required input by evaluating a magnetic field as a function \(B\):

\[
\text{do } K=0,\text{max}K \\
\text{do } J=0,\text{max}J \\
\text{do } I=0,\text{max}I
\]
Thus, the first \( maxI+1 \) values are along the minimum-y and minimum-z edge of the rectangular mesh. The first \( (maxI+1)*(maxJ+1) \) values are on the minimum-z face of the mesh.
For **USER**, the value of [parameter(1)] must correspond to the user modifying subroutine BFLD to specify their own field description. The value for [parameter(1)] must be larger than 5, as values 1 - 5 are reserved. **INPUT-MODEL** and the associated real numbers, and also OFF, MAGNETIC, and BOTH are available for user-defined application.

18. **ECHO**

Syntax: ECHO [parameter(1)]

Example: ECHO0

Default: echoing is active

If “ECHO 0” is inserted in the input stream, subsequent input will not be echoed to the output. If “ECHO1” is inserted (on a line by itself) in the input stream, all subsequent input will be echoed.

19. **ELECTRONS (Forward Only)**

Syntax: ELECTRONS

Default: electron source if no PHOTONS primary keyword is used.

This keyword defines the source particles to be electrons rather than photons.

20. **ELECTRON-EMISSION (ITS ACCEPT Forward Only)**

Syntax:

```
E
L
E
C
T
R
O
N
-
E
M
```
rameter (4)
]
[parameter (5)
]
[parameter (6)
Example:

ELEC
Default: Electricemi
This keyword signals that electron emission is to be tallied. If [parameter(1)] is positive, all electrons entering the zone number specified will be terminated. If [parameter(1)] is negative, all electrons entering the zone number specified (by the absolute value of the parameter) will continue to be transported in the zone.

**WARNING:** In CG, electron emission tallies will be made if they passed through a specified surface to exit the zone they were in prior to entering the electron-emission zone. In many cases the two adjacent zones will share a surface, so that the surface that is passed through to exit the prior zone will be the same as the surface passed through to enter the electron-emission zone, but this is not always the case so the user is encouraged to give this point careful consideration. In CAD, tallies are made on the surfaces of the electron emission zone as particles enter the zone.
The number of surfaces on which tallies are desired is specified with [parameter(2)]. This must be followed immediately with 2 [parameter(2)] lines. The user must specify the surface indices of the body numbers for which tallies are desired using [parameter(3)] and [parameter(4)], respectively. Figure 2 on page 77 illustrates how many of the body types used in ACCEPT have their surfaces numbered. The rectangular parallelepiped (RPP) is numbered the same as the BOX where \( A_1 \) points in the positive x-direction, \( A_2 \) points in the positive y-direction, and \( A_3 \) points in the positive z-direction. The right circular cylinder (RCC) is numbered the same as the TRC.

The subsurfacing (equidistant subdivision) of each surface must be specified using [parameter(5)] and [parameter(6)] to request the number of divisions in the two dimensions. The meaning of these “U” and “V” parameters depend upon the surface being subsurfaced and are discussed below. The output will refer only to a single subsurface number, corresponding to incrementing the [parameter(6)] value first. For example, the 6th subsurface in a “33” subsurfacing corresponds to the 2nd U-division and the 3rd V-division. The U and V coordinates are always incremented by increasing in their assigned coordinate axes. For example, on an RPP U and V always increase from the minimum Cartesian dimension to the maximum Cartesian dimension regardless of the surface. On a BOX U and V always increase with respect to moving along the \( A_1 \), \( A_2 \), and \( A_3 \) vectors away from the \( V \) position.

With a CAD geometry emission zone, the \texttt{BODY} keyword and the subsurfacing divisions are still required but they will not be used. With a CAD geometry part, there is no subsurfacing capability. Each face is a single surface tally. If spatial resolution is desired, the CAD geometry should be modified to cut the faces into smaller patches. For ACIS geometry, surface numbers are assigned in the same order as when they are viewed in Cubit. For Cholla geometry, each file is considered to contain a single surface. In neither case, surfaces are renumbered based on the order they are encountered in the .prm file.
Currently, combining ACIS and Cholla formats into a single emission zone is not allowed.

For binning in energy and angle, the same secondary keywords apply as for ELECTRONESCAPE (NBINE, NBINT, NBINP). Directions for electron emission tallies are relative to the local surface normal. That is, zero degrees theta is the normal into the emission zone. Theta is always defined as into the emission zone (no matter how bodies may be used to construct the zone or how bodies may border the zone), so there should never be a theta greater than 90 degrees. Zero degrees phi is defined differently depending upon the surface. The 90 degree phi coordinate is determined based on a right-hand rule of zero theta and zero phi.

In some cases zero degrees phi may be a user-defined referenced direction. To specify such a reference direction, the input line with the two subsurfacing integers may contain the keyword AZ, followed by any body number except an RPP. The zero reference vector is the component of the V vector of the AZ body that is perpendicular to the H vector of the subsurfaced body. If that definition of the reference vector is null, the code attempts to use the +i, +j, and +k vector for the zero azimuth vector, in that order. If no AZ body is specified, the +i vector will be used as the body vector. This keyword may be applied to all surfaces of a TRC, RCC, and TOR. For the planar surfaces of a TRC or RCC, the referenced direction also defines the zero direction for any angular division specified with the NBINP sub-keyword.

With a CAD geometry emission zone, the zero azimuthal reference direction is the positive x-coordinate axis. If the surface normal coincides with the x-coordinate axis, then the zero azimuth is the positive y-coordinate axis. If present, the AZ keyword will have no effect on the tallies.

**Keywords for**

**This feature is currently available for the surfaces of the following bodies:**
(a) RPP – The U-V surfaces are determined as follows: for constant-x planes, they are the y and z divisions; for constant-y planes, they are the z and x divisions; for constant-z planes, they are the x and y divisions, respectively. Zerodegrees phi is defined by the U coordinate axis.

(b) BOX – The U-V surfaces are determined as follows: for constant-A1 planes, they are the A2 and A3 divisions; for constant-A2 divisions, they are the A3 and A1 divisions; for constant-A3, they are the A1 and A2 divisions, respectively. Zerodegrees phi is defined by the U coordinate axis.

(c) WED – The U-V surfaces are determined in the same manner as for the BOX, except there is no surface 4 and surface 2 is uses the vector A1 - A2 for U divisions. The rectangular divisions of surfaces 5 and 6 are not modified for the triangular surfaces of the wedge, so only half of the divisions may be tallied. Zero degrees phi is defined by the U coordinate axis.

(d) RCC – On the planar surfaces 1 and 2, the U-V divisions are in angle about the axis of the cylinder and in radius, respectively. On the cylindrical surface, the U-V divisions are in angle about the axis of the cylinder and in the axial coordinate (or height) of the cylinder, respectively. The zero phi direction on the planar surfaces is determined by the zero angle about the axis, which is defined by the AZ logic. The zero phi direction on the cylindrical surface is in the direction of the H vector of the body.

(e) TRC – The U-V divisions are determined in the same manner as for the RCC, except that the zero phi direction on the conical surface is in the direction of the point of the cone. Note that the direction to the point of the cone depends upon the respective radii of the bases of the TRC, rather than on the H vector of the body.

(f) SPH – The U-V divisions are azimuthal about the laboratory z axis with respective to the positive-x axis and in polar angle with respect to the positive-z axis. The zero phi direction is tangential to the surface of the sphere in the direction of the positive-z pole of the body.

(g) TOR – The U-V divisions are the poloidal-phi and toroidal-phi directions. This is analogous to the division of the cylindrical surface of the RCC, where the torus is a cylinder with its axis wrapped in a circle. Thus, the RCC azimuthal angle about the axis is the TOR poloidal angle, and the RCC axial coordinate is the TOR toroidal angle. The U divisions begin at the outer-most radius of the torus and are incremented such that the 90 degree poloidal angle is the H vector. The V divisions begin at a location specified by the AZ logic and are incremented such that the 90 degree toroidal angle is determined by the H vector crossed with the AZ vector. The zero phi direction is tangential to the surface of the torus, such that it is in the same direction as the H vector of the torus on the outside and in the opposite direction of the H vector of the torus on the inside.

21. ELECTRON-ESCAPE (Forward Only)

Syntax: ELECTRON-ESCAPE

Default: Differential electron escape is not tallied.

This keyword signals that differential electron escape is to be tallied. The following are secondary keywords associated with this primary keyword that describe the bin structure used in tallying electron escape.
Key words for ITS

(a) **NBINE**

Syntax: NBINE [parameter(1)] [keyword]

ITS Example: NBINE5USER

```
0
.8 0.6
0.4 0.2
0.0
MITS
Example:
NBINE5 USER
10 20 30
40 50
Default:
In ITS,
10 bins
of equal
width
are
used. In
MITS,
the bin
structure
```
corresponds to the electron group structure.
In ITS, choices for [keyword] are:

i. **LOG** – Logarithmic grid spacing, with some parameter \( x \) such that \( E_{\text{cutoff}} = x E_0 \), where \( E_{\text{cutoff}} \) is the cutoff energy and \( E_0 \) is the source energy, and the grid values are defined as \( E_{i+1} = xE_i \).

ii. **USER** – User defined energy grid. The code will then read lower bound energies (MeV) for the number of energy bins specified by \([\text{parameter}(1)]\) in descending order. The maximum lower bound must be less than the maximum source energy.

For either of these tertiary keywords, the user must ensure that the energy is less than or equal to the global electron cutoff (if less than, the grid will be truncated).

In MITS, this keyword allows the user to “collapse” the default bin structure (which is one bin for every group in the electron group structure) into fewer bins.

i. **USER** – The **USER** secondary keyword must be present, and this line must be followed by \([\text{parameter}(1)]\) numbers (integers) which specify group indices. The lower-bound energies of the specified groups form the lower-bound energies of the escape bins. The indices should appear in strictly increasing order.

(b) **NBINT**

Syntax: **NBINT** \([\text{parameter}(1)]\) \([\text{keyword}]\)

Example: **NBINT5USER**

```
10.0 30.0 90.0 135.0 180.0
```

Default: 18 bins of 10 degrees each up to 180 degrees.

This keyword allows the user to define a polar bin structure for recording electron escape. \([\text{parameter}(1)]\) specifies the number of polar bins.

If the **USER** secondary keyword is present, then this line must be followed by \([\text{parameter}(1)]\) numbers which specify angle bins in ascending order up to 180 degrees.

If the **DIRECTION-SPACE** (**CYLTRAN or ACCEPT Only**) keyword is used, \([\text{parameter}(1)]\) polar bins are internally generated and azimuthal bins are generated such that
angular bins are approximately equal in size. See the warning under sub-keyword
NBINP.

If no secondary keyword is present, then [parameter(1)] equal width bins
will be defined.

(c) **NBINP (CYLTRAN or ACCEPT Only)**

Syntax: NBINP [parameter(1)] [USER]

Example: NBINP5USER

8

E

K e

y w

o r

d s

f o

r

I T S

10.0 30.0 140.0 235.0 360.0

Default: 1 bin of 360 degrees.

This keyword allows the user to define an azimuthal bin structure for recording electron escape. [parameter(1)] specifies the number of azimuthal bins. If the **USER** secondary keyword is present, then this line must be followed by [parameter(1)] numbers which specify angle bins in ascending order up to 360 degrees. If the USER keyword is not present, then [parameter(1)] equal width bins will be defined. **Warning:** NBINP cannot be used with the NBINT/DIRECTION-SPACE option.

22. **ELECTRON-FLUX (Forward Only)**

Syntax: ELECTRON-FLUX [parameter(1)] [parameter(2)]

Example: ELECTRON-FLUX35
Default: No electron flux tallied.

This keyword signals that electron flux is to be tallied in all subzones for input zones [parameter(1)] through [parameter(2)]. The automatic subzoning features of the ITS codes are discussed in more detail in the Subzoning section. If either parameter is omitted or 0, flux will be calculated in all zones. The same secondary keywords apply as for ELECTRONESCAPE (NBINE, NBIINT, NBINP).

Calculation of electron flux in zones where macroscopic fields have been specified is not allowed. The user must ensure that the zone dependent electron cutoff energies (see keyword CUTOFFS) for zones [parameter(1)] through [parameter(2)] are all equal.

23. ELECTRON-SURFACE-SOURCE (Adjoint Only)

Syntax: ELECTRON-SURFACE-SOURCE

Default: No forward electron surface sources in adjoint mode.

This keyword signals that electron escape is to be tallied. The following keywords are secondary keywords associated with this primary keyword that describe the bin structure used in tallying electron escape. Directions are in the LOCAL-NORMAL frame unless the DELTA0-AVE subkeyword is used.

(a) NBINE

Synt
ax:
NBI
NE
[parameter(1)]
USE
R
Exa
mple :
NBI
NE5
USE
R
111
Defa ult:
The bin struc
ture corre
sponds to
the elec
tron
group struc
ture.

This keyword allows the user to "collapse" the default bin structure (which is one bin for every group in the electron group structure) into fewer bins. [parameter(1)] specifies the number of energy bins. The USER secondary keyword must be present, and this line must be followed by [parameter(1)] numbers (integers) which specify group indices (before inversion, for adjoint). In adjoint mode, the upper-bound energies of the specified groups form the upper-bound energies of the escape bins. The indices should appear in strictly increasing order.

8.

Keywords for ITS
(b) **NBINT**

Syntax: NBINT [parameter(1)]

[keyword] [keyword] Example:

NBINT6USER COSINE-LAW

5. 10.5 22.25 45 70 90

Default: Nine cosine-law sources within one of nine equal polar-angle bins with azimuthal symmetry about the z-axis.

This keyword allows the user to specify the number, angular-extent, and type of angular distribution of forward sources for an adjoint calculation with electron surface sources. [parameter(1)] specifies the number of source polar-angle bins. Without either the USER or DIRECTION-SPACE keyword, this will generate [parameter(1)] equal solid-angle polar-angle bins. When the USER keyword is used, the following line must contain [parameter(1)] values which are the upper bounds of the polar-angle bins. When the DIRECTION-SPACE (CYLTRAN and ACCEPT Only) keyword is used, [parameter(1)] polar bins are internally generated and azimuthal bins are generated such that angular bins are approximately equal in size. DIRECTION-SPACE can only be used with the DELTA0-AVE setting.

The final keyword describes the source angular distribution within these bins. It should be one of the following (COSINE-LAW is the default):

i. **ISOTROPIC** The forward-source is uniform in angle within the specified angular bin, e.g. a thin surface of a radioactive material.

ii. **COSINE-LAW** The forward source has a cosine-law distribution with respect to the surface normal, e.g., this corresponds to the "isotropic-flux" sources of cosmic particles.

iii. **DELTA0-AVE** The forward source is the average, within the specified angular bin, of all plane-wave sources with normals within the specified angular bin. For each plane wave, the source is a delta function normal to the plane, $\delta(\Omega-1)$.

(c) **NBINP (CYLTRAN or ACCEPT Only)**

Syntax: NBINP [parameter(1)] [USER]

Example: NBINPSUSER

10.0 30.0 140.0 235.0 360.0

Default: 1 bin of 360 degrees.

This keyword allows the user to define an azimuthal bin structure for recording angular distribution of
forward sources for an adjoint calculation with electron surface sources. [parameter(1)] specifies the number of azimuthal bins. If the USER secondary keyword is present, then this line must be followed by [parameter(1)] numbers which specify angle bins in ascending order up to 360 degrees. If the USER keyword is not present, then [parameter(1)] equal width bins will be defined. NBINP cannot be used with the NBINT/DIRECTION-SPACE option.

24. ELECTRON-VOLUME-SOURCE (Adjoint Only)

Syntax: ELECTRON-VOLUME-SOURCE [parameter(1)]
[parameter(2)]
Example: ELECTRON-VOLUME-SOURCE 57
Default: No volume source.

This keyword signals that a forward source (i.e., the flux of adjunctons) is to be tallied in all subzones for input zones [parameter(1)] through [parameter(2)]. If either parameter is omitted or 0, the source will be calculated in all zones. The same secondary keywords apply as for ELECTRON-SURFACE-SOURCE (NBINE, NBINT, NBINP), but the angular distribution must always be ISOTROPIC.

25. ENERGY (Forward Only)

Syntax: ENERGY [parameter(1)] [keyword] [parameter(2)]
Example: ENERGY 4.3
Example: ENERGY GROUP1 (MITS Only)
Default: For ITS, 1.0 MeV monoenergetic source. For MITS, a mono-group source in the highest energy group.

In ITS, the source is specified as a mono-energetic source with energy \[\text{parameter(1)}\] in MeV. In MITS, the source may be specified as a mono-energetic or mono-group source. For an electron source, the user may specify a monoenergetic source with the energy specified by \[\text{parameter(1)}\] in MeV. For any particle species, if the keyword GROUP appears on the line, then the associated \[\text{parameter(2)}\] specifies the single group index which will be used for all the source particles. The specific energy of the individual histories will be sampled uniformly over the width of the group. If the user desires a mono-energetic photon source, the cross sections generated by CEPXS must include the appropriate single-energy (or “source-line”) group.

26. **ESCAPE-SURFACES** *(Forward Only)*

Syntax: ESCAPE-SURFACES \[\text{parameter(1)}\]

Example: ESCAPE-SURFACES3

Default: All surfaces of the escape zone (tallied as a single surface) for ACCEPT.

Both surfaces (ZMIN and ZMAX) for TIGER. All three surfaces (ZMAX, ZMIN, and RMAX) for CYLTRAN.

This specifies the number of escape surfaces for which the integral particle escape or the requested particle escape will be displayed. This keyword must be followed by \[\text{parameter(1)}\] separate lines of the secondary keyword SURFACE.

Any specification for CAD will be ignored. All escaping particles are scored in a single body/surface tally.

(a) **SURFACE** Syntax: SURFACE

[keyword] (non-ACCEPT codes) or SURFACE
[parameter(1)] BODY
[parameter(2)] (ACCEPT codes) Example:
SURFACE ZMAX (non-ACCEPT codes) or
SURFACE3BODY2
(ACCEPT codes)

This sub-keyword specifies the surface index through which the escaping particles will be calculated.

For TIGER and CYLTRAN, surface **ZMIN** refers to the minimum-z surface and surface **ZMAX** is the maximum-z surface. For CYLTRAN, surface **RMAX** is the lateral escape surface at maximum radius. For ACCEPT, it is necessary to specify both the surface index and the **BODY** number. Figure 2 on page 77 illustrates how many of the body types used in ACCEPT have their surfaces numbered. In the case of the arbitrary polyhedron (ARB), the user explicitly specifies the order in the input. The right circular cylinder is numbered the same as the truncated right-circular cone (TRC). The rectangular parallelepiped (RPP) is numbered the same as the BOX where A1 points in the positive x-direction, A2 points in the positive y-direction, and A3 points in the positive z-direction.

The user is cautioned to make sure that the desired surface has a unique description for the way the user has specified the geometry, otherwise the result may be invalid. For example, if a zone is defined as the union of two bodies and those two bodies have coincident surfaces, a particle may exit the zone surface through either one of the two body surfaces.

27. **FILE-NAMES**

Syntax: FILE-NAMES

Default: Default names (based on Fortran unit names) are used.

This keyword allows the user to specify names of files to be opened for
input and output. File names must be enclosed in single quotation marks.

(a) AREAL-DENSITY-FILE
Syntax: AREAL-DENSITY-FILE
[keyword]
Example: AREAL-DENSITY-FILE
‘areal.dat’
Default: Areal density data is written to “fort.4”.

This keyword specifies that areal density data will be written to the file named [keyword].

(b) CONNECTIVITY-DUMP-FILE Syntax: CONNECTIVITY-DUMP-FILE [keyword] Example: CONNECTIVITY-DUMP-FILE ‘connect.dat’ Default: Connectivity data needed for a connectivity read is written to “fort.9”.

This keyword specifies that connectivity data will be written to the file named [keyword].

(c) CONNECTIVITY-READ-FILE
Syntax: CONNECTIVITY-READ-FILE

[keyword]
Example: CONNECTIVITY-READ-FILE

‘connect.dat’ Default: Connectivity data is read from “fort.13”.
This keyword specifies that connectivity data will be read from the file named [keyword].
DUMP - FILE
Syntax:
DUMP - FILE
[keyword]
Default: Dumpdata needed for a restart is written to "fort.10". This keyword specifies that dump data will be written to the file named [keyword].

(e)
Example:

INTEGRATE

Intermediate File

Inte
r
O
u
t,

D
e
f
a
u
l
t:

I
n
te
r
m
e
d
i
a
t

o
t
p
u

i
s
w
r
This keyword specifies that intermediate output will be written to the file named [keyword].

(f) **FINITE-**

```
ELEMENT
 FILE
```

(A C C)
- FILE

[keyword]

Example:

FINITE - ELEMENT -
FILE

torus

default:

finite element
This keyword specifies that finite-element output will be written to the file named [keyword].

(g) PFF-

F
I
L
E

(
TS

PPFF

Only
)

Syntax:

PFFF

FILE

[keyword]

E
Electron emission, electron escape, or phosphor
onescape

distributions
are

written
to
This keyword specifies that electron emission distribution output will be written to the file named [keyword].

(h) **PLOT-FILE** *(CYLTRAN or ACCEPt Only)*
Syntax: PLOT-FILE [keyword]
Example: PLOT-FILE 8
'weasel.
dat' Default: Plot data is written to "WEASEL.OUT". This keywordspecifies that plots output will be written to the file named [keyword].

(i) RESTART-FILE Syntax: RESTART-FILE [keyword] Example: RESTART-FILE 'Restart'
Default: Restart data is read from "fort.14". This keywordspecifies that restart data will be read from the file named [keyword].

(j) XSECTION-FILE Syntax: XSECTION-FILE [keyword] Example: XSECTION-FILE 'XSfile'
Default: Cross sections are read from "fort.11". This keywordspecifies that cross sections will be read from the file named [keyword].

28. FINITE-ELEMENT-FORMAT (ACCEPT Forward Only)
Syntax: FINITE-ELEMENT-FORMAT [CHARGE] [DOSE] [NO-SIGMA] [LIST] [LISTZONE] Example: FINITE-ELEMENT-FORMAT CHARGE Default: no finite-element file
This keyword signals the creation of a Tecplot R
format file written to “fort.3”. The file contains charge deposition
(secondary keyword CHARGE) and/or energy deposition (secondary
keyword DOSE) for each subzone. If neither secondary keyword is used,
both energy and charge deposition will be written.

The NO-SIGMA secondary keyword will suppress the statistical
uncertainty information
from being printed to the finite-element file. The LIST secondary
keyword will suppress all node and element information. This option will
simply list the dose and/or charge deposition data. Unlike the default
behavior, the list capability will include the dose for all zones, whether
they were subzoned or not. Counting each zone that is not
subzoned as a single subzone, the line number in the finite-element file will
correspond to the subzone number for the geometry.

The LIST-ZONE secondary keyword has the same effect as the LIST
secondary keyword, but
it also includes zone and subzone numbers in the listing of deposition
data. Without the LIST keyword, each ITS zone that is subzoned is
written as a separate Tecplot zone. Each subzone of an ITS
zone represents an element in the Tecplot zone. The energy and charge
deposition data are written as volume-averaged values associated with
each element. Since Tecplot expects nodal values, each element node
has energy and/or charge deposition values but only the first node of each
element represents the correct volume-averaged value.
Another robust method for viewing data is to use Ensight [33]. Ensight will display element data but will not read an Tecplot file. A conversion program called “its2exo” is part of the ITS distribution in the Tools directory. This program will convert the data from the Tecplot format to an Exodus format.

29. **GEOMETRY** This keyword signals the beginning of the geometry information. The choice among the usages depends on which of the member codes of ITS has been selected: TIGER, CYLTRAN, ACCEPT, or CAD. Keyword usage information and a detailed discussion of the GEOMETRY keyword input and syntax requirements is contained in the section specific to the member code chosen. The ACCEPT Geometry section is also relevant to the use of CAD geometry.

30. **HISTORIES**

   Syntax: HISTORIES [parameter(1)]

   Example: HISTORIES 100000

   Default: 1000 histories

   This specifies the total number of primary particle histories to be followed. [parameter(1)] cannot be greater than 2,147,483,647 (i.e., $2^{31} - 1$). To simulate more histories, the HISTORIES-PER-BATCH keyword should be used. HISTORIES and HISTORIES-PER-BATCH are mutually exclusive keywords.

31. **HISTORIES-PER-BATCH**

   Syntax: HISTORIES-PER-BATCH [parameter(1)]

   Example: HISTORIES-PER-BATCH 10000

   Default: 1000 total histories

   This specifies the number of primary particle histories to be followed per batch. [parameter(1)] cannot be greater than 2,147,483,647 (i.e., $2^{31} - 1$). To simulate more histories, the number of batches should be increased. HISTORIES and HISTORIES-PER-BATCH are mutually exclusive keywords.

32. **INCLUDE-FILE**
Syntax: INCLUDE-FILE
Example: INCLUDE-FILE
GeometryInp

Default: All input is read from a single file.

This allows the user to include input parameters from other files. With this keyword, a user can eliminate redundant data between separate input decks. For example, the geometry data can be in one file, and that file can be included in multiple input decks.

33. **LINE-TALLY-WITH-CONTINUUM** *(Forward Only)*

Syntax: LINE-TALLY-WITH-CONTINUUM
Default: Line radiation is tallied separate from the continuum for photon escape and flux.

This keyword triggers line radiation to be tallied in the corresponding energy bin of the continuum radiation for photon escape and flux. This does not affect annihilation line radiation tallies, which are treated separately with the ANNIHILATION-LINE-TALLY-WITH-CONTINUUM keyword.

34. **MAGNETIC-TRAP-LIMITS** *(CYLTRAN or ACCEPT Only)*

Syntax: MAGNETIC-TRAP-LIMITS [parameter(1)] [parameter(2)]
Example: MAGNETIC-TRAP-LIMITS 10 1E+5
Default: Limits are 3 and 1000000, respectively.
With this keyword, the user can change the counter limits for determining that a particle is trapped in a magnetic field in an avoid. Two types of traps are detected: a magnetic mirror trap, in which a particle may bounce back and forth within the trap, and a particle that is circling within the magnetic field with no velocity parallel to the magnetic field lines. Both of these conditions are detected in the code by counting occurrences of incidents that may indicate the condition, with count limits set by parameter(1) and parameter(2), respectively. In both cases there is a trade-off: a low count limit may be more computationally efficient (by spending less effort tracking trapped particles) but may be less accurate (by killing particles that are not trapped). The first counter measures the number of times that the particle changes direction with respect to the magnetic field. The second counter measures the number of times that successive timesteps move the particle in a direction that is strictly perpendicular to the magnetic field lines.

Since the transport of a particle between reflections in a magnetic mirror may be quite expensive, a relatively low number is recommended for the first counter. Since there are typically 1000 timesteps for a particle to complete a circle in a magnetic field, a relatively high number is recommended for the second counter. Neither limit may be set lower than 2.

35. **MICRO (MITS Forward Only)**

   Syntax: MICRO Default: Energy and charge deposition
   determined by folding the flux into the appropriate cross section (e.g., restricted stopping power for electron energy deposition), with an additional microscopic contribution for particles that fall below cutoff.

   With this keyword, energy and charge deposition are entirely microscopic. This has been observed to be more efficient for calculating charge deposition, but less efficient for calculating energy deposition.

36. **NEW-DATA-SET**

   Syntax: NEW-DATA-SET
Example: NEW-DATA-SET
Default: one run.

This keyword signifies that the data set for a particular Monte Carlo run has been read and that the dataset for a new Monte Carlo run follows. Its purpose is to permit multiple Monte Carlo runs within a single code execution. **Its usage is an exception to the rule that the primary keywords are order independent.**

The input data must be given for each problem, and the input data sets for the different problems must be separated from one another by a line containing this keyword. Different cross section files can be used for different data sets, if they are specified using the FILENAMES/XSECTION-FILE keyword.

For CAD calculations, the prm file and CAD geometry files will not be reread for each run. Therefore, all of the problems must use the same CAD calculation parameters and the same CAD geometry.

37. **NO-COHERENT** *(ITS Only)*

Syntax: NO-COHERENT
Default: Coherent photon scattering will be included in the calculation.

This keyword is intended for development purposes only. This keyword deactivates the simulation of coherent photon scattering.

38. **NO-DEPOSITION** *(Forward Only)*

Syntax: NO-DEPOSITION [CHARGE] [DOSE]
Example: NO-DEPOSITION CHARGE
Default: Energy and charge deposition are tallied for every subzone.
This keyword suppresses tallying of energy and charge deposition. Using only the CHARGE secondary keyword suppresses tallying of charge deposition. Using only the DOSE secondary keyword suppresses tallying of energy deposition. Omitting both secondary keywords or using both will suppress tallying of both energy and charge deposition.

This keyword will reduce memory requirements, because memory is not allocated for the tallies that are suppressed by this keyword. Since the deposition is not tallied, it cannot be obtained using a code restart.

39. **NO-DEPOSITION-OUTPUT (Forward Only)**

Syntax: NO-DEPOSITION-OUTPUT
Default: Energy and charge deposition output is written to "fort.12" and unit 6 output.

This keyword suppresses energy and charge deposition output. This may prove useful when the FINITE-ELEMENT-FORMAT keyword is used with numerous subzones. The output can be obtained later by performing a code restart without this keyword.

40. **NO-DETAILED-DEPOSITION (Forward Only)**

Syntax: NO-DETAILED-DEPOSITION [CHARGE] [DOSE]
Example: NO-DETAILED-DEPOSITION CHARGE
Default: Detailed energy and charge deposition are tallied for every subzone.
This keyword suppresses tallying of energy and charge deposition details. These “details” indicate whether deposition was due to primary or secondary particles for ITS, and whether the deposition was due to particles at the cutoff energy or higher for MITS. Using only the CHARGE secondary keyword suppresses detailed tallying of charge deposition. Using only the DOSE secondary keyword suppresses detailed tallying of energy deposition. Omitting both secondary keywords or using both will suppress detailed tallying of both energy and charge deposition. This keyword will reduce memory requirements, because memory is not allocated for the tallies that are suppressed by this keyword. Since the deposition is not tallied, it cannot be obtained using a coder restart.

41. **NO-GEOMETRY-TABLE**

Syntax: NO-GEOMETRY-TABLE

Default: Geometry-dependent input settings are written to output.

This keyword suppresses the printing of geometry-dependent input settings.

42. **NO-INCOH-BINDING (ITS Only)**

Syntax: NO-INCOH-BINDING

Default: Incoherent photon scattering will include binding effects.

This keyword is intended for development purposes only. This keyword causes incoherent photon scattering to be simulated in the Klein-Nishina or free-electron approximation.

43. **NO-INTERMEDIATE-OUTPUT**

Syntax: NO-INTERMEDIATE-OUTPUT

Default: Intermediate output is written to “fort.12”.

This keyword specifies that output will be written only upon completion of the calculation.

44. **NO-KICKING (ITS Only)**

Syntax: NO-KICKING Defaults:

Terminal processing of electrons and positrons includes kicking, except when EBFIELDS is enabled.
This keyword is intended for development purposes only. The “kicking” of electrons and positrons is an approximation that moves the particle to account for transport at lower energies. This is based on the remaining practical range of the particle. Using this keyword will cause the particle energy and charge to be locally deposited. When EBFIELDS is enabled, there is no kicking, and using this keyword is redundant.

8.

Key

words

for

ITS

45. NO-KNOCKONS (ITS Only)

Syntax: NO-KNOCKONS
 Defaults: Secondary knock-on electrons are produced.

This keyword is intended for development purposes only. Production of secondary knock-on electrons is disabled, however primary electron energy loss and energy-loss straggling are not affected. See the NO-STRAGGLING keyword.

46. NO-OVERLAP-OUTPUT (CAD Only)

Syntax: NO-OVERLAP-OUTPUT
 Default: Overlaps in CAD geometry are accumulated and reported.

This keyword suppresses accumulation and reporting of overlaps in CAD geometry. For large parallel calculations, this may improve the parallel efficiency when static load balancing is being used.
47. **NO-STRAGGLING (ITS Only)**

   Syntax: NO-STRAGGLING
   Defaults: Energy-loss straggling is applied to electrons.

   This keyword is intended for development purposes only. Energy-loss straggling is disabled for all electron transport.

48. **NO-SZDEPOSITION-OUTPUT (Forward Only)**

   Syntax: NO-SZDEPOSITION-OUTPUT
   Default: Energy and charge deposition output is written to “fort.12” and unit 6 output for all zones and subzones.

   This keyword suppresses energy and charge deposition output for all subzones. This may prove useful when the FINITE-ELEMENT-FORMAT keyword is used with numerous sub-zones. Unlike the NO-DEPOSITION-OUTPUT keyword, this keyword will allow deposition quantities for zones to be printed out. If the NO-DEPOSITION-OUTPUT keyword is used, this keyword has no effect.

49. **OVERLAP-OUTPUT-MAX (CAD Only)**

   Syntax: OVERLAP-OUTPUT-MAX 20
   Default: A maximum of 10 overlaps in CAD geometry are accumulated and reported.

   This keyword sets the limit on the accumulation and reporting of overlaps in CAD geometry. For large parallel calculations, large values will decrease the parallel efficiency when static load balancing is being used.

   Setting this to zero disables the accumulation and reporting of overlaps in CAD geometry (see NO-OVERLAP-OUTPUT).
50. **PFF-FORMAT (ITS PFF Only)**

Syntax: PFF-FORMAT[SIGMA]

Example: PFF-FORMATSIGMA

Default: no pff file

This keyword signals the creation of a pff-formatted file written to "distrib.pff". The file contains one of: (1) electron emission distributions for each subsurface, (2) electron escape distributions for each surface, or (3) photon escape distributions for each surface. Only one of these three will be given in the pff file, with priority given in the order listed above among those distributions requested in the input deck. The optional secondary keyword **SIGMA** will include statistical uncertainty information in the pff file.

51. **PHOTONS (Forward Only)**

Syntax: PHOTONS [parameter(1)]

Example: PHOTONS1

Default: electron source

This keyword defines the source particles to be photons rather than electrons. For ITS only, if [parameter(1)] is 0 or omitted, then unscattered photons will be excluded from photon flux and escape scores. Otherwise, unscattered photons will be included in photon flux and escape scores.

1. **PHOTON-ESCAPE (Forward Only)** See ELECTRON-ESCAPE
2. **PHOTON-FLUX (Forward Only)** See ELECTRON-FLUX
3. **PHOTON-SURFACE-SOURCE (Adjoint Only)**

See ELECTRON-SURFACE-SOURCE, with the following additions.
For the PHOTON-SURFACE-SOURCE keyword, the
DIRECTION-SPACE secondary keyword has the following optional secondary keywords. Syntax: . . .
DIRECTION-SPACE [keyword][keyword][keyword] [parameter(1)]
Example: NBINT6DIRECTION-SPACE DELTA0-AVE
RAYTRACE RAYPRINT

2

5
0

9
0
.
1

7
5
.
5
2
7
0

D e f a u l t :
F
rather than a ray - trace
Although not strictly a secondary keyword of DIRECTION-SPACE, the DELTA0-AVE sec ondary keyword is required whenever DIRECTION-SPACE is present. The secondary keyword RAYTRACE causes an uncollided kerma calculation to be performed. This calculation performs a ray-tracing activity for every angle bin produced by 8.

Keywords for ITS the direction space procedure. The photon surface source is attenuated along the centroids of each angular bin using the total photon interaction
cross section. The resulting photon source is folded with the photon kerma cross section for the specified detector material. The kerma is presented in the usual direction space output format. The RAYTRACE keyword is not allowed with the USER secondary keyword of NBINT. No other output is produced unless the additional secondary keyword RAYPRINT is used.

The RAYPRINT secondary keyword allows the user to print out the ray segment data for any or all of the rays generated with the RAYTRACE keyword. Ray segment data contains all the information needed to perform a 1-D transport calculation along the path of a ray. RAYPRINT with no parameters prints ray segment data for all rays. If [parameter(1)] is present, it is the number of rays for which the user is requesting ray segment data and requires [parameter(1)] pairs of angles beginning on the next line of input. The angle pairs correspond to the theta (polar angle, 0 to 180 degrees) and phi (azimuthal angle, 0 to 360 degrees) of each angular bin for which the user wants ray segment data printed. The specified angles do not have to be exactly the angular bin centroids, ITS will find the angular bin that the given pair falls in and print the ray segment data for that centroid. The RAYPRINT secondary keyword restricts which ray data is written to the arealdensity file if any restrictions are specified (see the AREAL-DENSITY-OUTPUT keyword).

55. PHOTON-VOLUME-SOURCE (Adjoint Only)

See ELECTRON-VOLUME-SOURCE

56. PLOTS-OVAL-RESOLUTION (ACCEPT Only)

Syntax: PLOTS-OVAL-RESOLUTION parameter(1)
Example: PLOTS-OVAL-RESOLUTION 36
Default: 360

With this keyword, the user can change the number of points used to represent curved edges of bodies in plots. Increasing this value may also increase the maximum number of trajectory points plotted (see the TRAJECTORY-POINTS keyword).

57. PLOTS (CYLTRAN or ACCEPT Only)

   CYLT
   RAN

Syntax:
PLOTS
Example:
PLOTS0
305
ACCEPT
Syntax:
PLOTS
[parameter(1)]
[parameter(2)] ...
[parameter(7)]
ACCEPT
Example:
PLOTS3
050590
180
0505180
90
050500
Default:
No plots are generated.
This keyword causes geometry and/or particle track data to be written to a file in a form that can be plotted. The plot data can be customized with the sub-keywords. Particle track data can only be written from the master process, so trajectories will not be plotted if the DYNAMIC-MPI keyword is used.

For CYLTRAN this keyword is used to plot the geometry for any of the CYLTRAN codes. With fields, the electron and positron trajectories are plotted in zones where macroscopic fields are defined. Use of this keyword will produce a \( \rho - z \) plot of that portion of the problem cylinder bounded by \([\text{parameter}(1)]\) through \([\text{parameter}(4)]\), which define the minimum \( \rho \), the maximum \( \rho \), the minimum \( z \), and the maximum \( z \), respectively, in cm. If the parameters are left blank, the entire problem cylinder will be plotted.

For ACCEPT use of this keyword will produce \([\text{parameter}(1)]\) parallel projections of the body specification as given under the GEOMETRY keyword. \([\text{parameter}(2)]\) through \([\text{parameter}(7)]\) are repeated \([\text{parameter}(1)]\) times on separate lines. \([\text{parameter}(6)]\) and \([\text{parameter}(7)]\) specify the spherical polar angles \( \varphi \) and \( \theta \), respectively, in degrees that define the direction from which the geometry is to be viewed. \([\text{parameter}(2)]\) through \([\text{parameter}(5)]\) specify the minimum \( x \), the maximum \( x \), the minimum \( y \), and the maximum \( y \), respectively, in cm of the plotted projection. If the PLOTS keyword is used, all parameters are required; there are no defaults. With fields, the electron and positron trajectories are plotted in zones where macroscopic fields are defined only on the final projection.

(a) ORBITS (ITS CYLTRAN or ACCEPT Only)

Syntax: ORBITS [\text{parameter}(1)]

Example: ORBITS 10

Default: Electron and positron trajectories associated with every 5th source particle of the first batch will be plotted.

Electron and positron trajectories associated with source particles of the
first batch that are multiples of [parameter(1)] are to be plotted in those zones for which macroscopic fields have been defined. A blank for [parameter(1)] will cause all electron and positron trajectories of the first batch to be plotted. This keyword can only be used when the EBFIELD primary keyword is also used to activate fields.

(b)
Syntax:

PAGE HEADER

[ keyword ]

Example:
variables="XROT","YROT"
Default: The page header is a blank line.

(c) BREAK-

INDICATOR

(f

CY

L

TRAN

or

r
ACCEPTED

Only
)

Syntax:

BREAKER

[ k

C

E

P

T

O

n

l

y

S

y

n

tax
:

B
R
E
A
K

- I
N
D
I
C
A
T
O
R

] k
zone
Default: The break indicator is a blank line.

(d) PLOT-3DAXIS (CYLTRAN or ACCEPT Only)
Syntax: PLOT-3DAXIS
Example: PLOT-3DAXIS
Default: The coordinate axes are not indicated on the plots.
This keyword will cause the 3D coordinate axes to be drawn on the plots.

58. **POSITION** *(Forward Only)* Syntax: POSITION

Default: The source is located at the origin.

This keyword defines the position of the source. It must be followed by one of the following sub-keywords to further describe its spatial distribution:

(a) **POINT**

Syntax: POINT [parameter(1)]

or POINT [parameter(1)] [parameter(2)] [parameter(3)]

Example: POINT 2.0

or POINT 0.0 0.0 2.0 (non-TIGER codes)

Default: The source is a point at the origin.

This sub-keyword specifies that the source is a point. For TIGER, the single location is the z coordinate of the point. For the non-TIGER codes, the three parameters specify the x, y, and z coordinates, respectively.

The default directional distribution is mono-directional in the positive z-direction, but the reference direction and source distribution may
be specified by the DIRECTION keyword.

(b) **LINE** *(CYLTRAN or ACCEPT Only)*

Syntax: LINE Example: LINE  
3.2 8.1 5.5  
9.1 3.4 3.0 Default: There is no default LINE distribution.

This keywordspecifies the source as a line. The command line following the keyword contains the x, y, and z coordinates of one end of the line. The next command line containsthe x, y, and z coordinates of the other end of the line. The sourceis sampled uniformly along the line segment.

The default directional distribution is mono-directional in the positive z-direction, but the reference direction and source distribution may be specified by the DIRECTION keyword.

(c) **DISK** *(CYLTRAN or ACCEPT Only)* Syntax: DISK [parameter(1)]  
[parameter(2)] [parameter(3)]

Example: DISK 0.0 0.0 2.0

Default: There is no default DISK spatial distribution.

---

This keywordspecifies the source as a disk. Thethree parameters specifythe x, y, and z coordinates of the location of the center of the disk.

The default angular distribution is mono-directional in the
positive-z direction. The orientation is normal to the source direction (i.e., in the x-y axis). Both the source direction and orientation are controlled by the DIRECTION keyword.

This sub-keyword should be followed by the following tertiary keyword:

i. **RADIUS**

Syntax: `RADIUS [parameter(1)] [keyword]`

Example: `RADIUS 3.5 RADIAL-BIASING`

Default: Zero radius point source.

[parameter(1)] specifies the radius of the disk source. This parameter may be followed by the keyword **RADIAL-BIASING** that will cause source particles to be sampled uniformly in radius (rather than uniformly in area).

(d) **ANNULUS** *(Cyltran or Accept Only)* Syntax: `ANNULUS [parameter(1)] [parameter(2)] [parameter(3)] [keyword][parameter(4)]`

Example 1: `ANNULUS 0.0 0.0 2.0 4.0 2.0 RADIAL-BIASING`

Example 2: `ANNULUS 0.0 0.0 2.0 PROFILE4 0.0 0.5 0.8 1.0 2.1 2.4 2.9 3.1`

Default: There is no default ANNULUS spatial distribution.

This keyword specifies the source as an annulus. The first three parameters specify the x, y, and z coordinates of the location of the center of the annulus. The default angular distribution is mono-directional in the positive-z direction. The orientation is normal to the source direction (i.e., in the x-y axis). Both the source direction and orientation are controlled by the DIRECTION keyword.

If the optional PROFILE keyword is not present, then the following line must contain two parameters that specify the outer and inner radii of the annulus. These parameters may be followed by the keyword **RADIAL-BIASING** that will
cause source particles
to be sampled uniformly in radius (rather than uniformly in
area).

If the PROFILE keyword is present, the user may specify a
radial distribution to be sampled from. [parameter(4)] following
the PROFILE keyword specifies the number of radial bins.
Starting on the following line, [parameter(4)] values specify the
cumulative probability array. This is followed by [parameter(4)]
values specifying the corresponding radial array.

(e) **RECTANGLE (CYLTRAN or ACCEPT Only) Syntax:**

```
RECTANGLE [NORMAL] [REVERSE] [A1GRID
parameter(1)] [A2GRID
parameter(2)]
Example: RECTANGLE

1.0 1.0 0.0
1.0 2.0 0.0
3.0 2.0 0.0
```

Example: RECTANGLE NORMAL REVERSE A1GRID2A2GRID2

```
1.0 1.0 0.0
1.0 2.0 0.0
3.0 2.0 0.0
0.6 0.2 0.2 0.0 Default: There is no default
```
RECTANGLE spatial distribution. This keyword specifies the source as a rectangle. The three lines following the keyword must contain the x, y, and z coordinates of vectors $V_1$, $V_2$, and $V_3$ specifying 3 corners of the rectangle. $(V_1-V_2)$ must be orthogonal to $(V_3-V_2)$. The default reference direction is in the positive-Z direction. An alternative reference direction and angular distribution may be specified with the DIRECTION keyword. If the NORMAL sub-keyword is used, the reference direction will be defined by $(V_1-V_2) \times (V_3-V_2)$. The NORMAL sub-keyword will override the reference direction given by the DIRECTION keyword, but any angular distribution will still be valid. If the NORMAL sub-keyword has been used, the REVERSE sub-keyword may be used to reverse the reference direction to be defined by $(V_3-V_2) \times (V_1-V_2)$. An arbitrary rectangular source grid may be specified by further specifying the number of grid divisions along each direction of the rectangular source and the source strength within each grid cell. A parameter following the A1GRID sub-keyword specifies the number of divisions in the $(V_1-V_2)$ direction. A parameter following the A2GRID sub-keyword specifies the number of divisions in the $(V_3-V_2)$ direction. Then, following the 3 vectors specifying the corners of the rectangle, the user must provide $[\text{parameter}(1)] \times [\text{parameter}(2)]$ values specifying the relative source strength of each grid cell. Cell assignments are made by first incrementing in the $(V_1-V_2)$ A1GRID direction and secondarily in the $(V_3-V_2)$ A2GRID direction. Regardless of source strengths input per cell, results are normalized to a single source particle over the entire source. The code uses uniform random sampling of source positions within each rectangular cell. If $[\text{parameter}(1)]$ and $[\text{parameter}(2)]$ are both unity or omitted, then there is only one grid cell, the relative strength must be 1.0, and the user should not attempt to specify the source strength within the cell.

(f) **SURFACE** Syntax: SURFACE [keyword] (non-ACCEPT) or SURFACE [parameter(1-3)] [keyword] (CYLTRAN) or SURFACE [parameter(1)] BODY [parameter(2)] [keyword] (ACCEPT)

Example: SURFACE ZMAX (non-ACCEPT) or SURFACE 1.0 2.5 5.0 OUTWARD (CYLTRAN) or SURFACE 3BODY 2 (ACCEPT)

Default: There is no default for the surface source option.

This sub-keyword specifies the surface from which source particles will be started. For TIGER and CYLTRAN, surface **ZMIN** refers to the minimum-z surface and surface
**ZMAX** is the maximum-z surface. For CYLTRAN, surface **RMAX** is the lateral escape surface at maximum radius. For CYLTRAN, an arbitrary cylindrical surface can be specified. [parameter(1)] is the lowerZ coordinate. [parameter(2)] is the upperZ coordinate. [parameter(3)] is the radius of the cylindrical surface. By default the reference direction is inwardnormal, but
Figure 2. Surface indices for ACCEPT bodies

The reference direction may be defined as outward normal by using the keyword `OUTWARD`. The reference direction cannot be changed with the DIRECTION keyword. For ACCEPT, it is necessary to specify both the `SURFACE` index number and the associated `BODY` number. The body referred to need not be part of the actual zone description of the geometry. Figure 2 on page 77 illustrates how many of the body types used in ACCEPT have their surfaces numbered. In the case of the arbitrary polyhedron (ARB), the user explicitly specifies the order in the input. The right circular cylinder is numbered the same as the truncated right-circular cone (TRC). The rectangular parallelepiped (RPP) is numbered the same as the BOX where \( A_1 \) points in the positive \( x \)-direction, \( A_2 \) points in the positive \( y \)-direction, and \( A_3 \) points in the positive \( z \)-direction. By default, the reference direction is inward normal, but the reference direction may be defined as outward normal by using the keyword `OUTWARD`. The reference direction cannot be changed with the DIRECTION keyword.

(g) **UNIFORM-ISOTROPIC-FLUX** (CYLTRAN or ACCEPT Only) Syntax: UNIFORM-ISOTROPIC-FLUX [parameter(1)]
Example: UNIFORM-ISOTROPIC-FLUX [parameter(1)] Default: There is no default
UNIFORM-ISOTROPIC-FLUX source.
This keyword allows for the simulation of a uniform isotropic radiation field. The reference direction is inward normal, and the distribution is cosine-law on the surface. The reference direction cannot be changed with the DIRECTION keyword.

For CYLTRAN, no parameter is necessary; a cylinder sufficient to surround the problem is automatically used.

For ACCEPT, [parameter(1)] specifies the index of the body over which the source will be sampled. **WARNING:** There is no diagnostic to ensure that anywhere outside of this body is the escape zone.

(h) **VOLUME**

Syntax: VOLUME [parameter(1)] [SHELL] [parameter(2)] [ZONE] [parameter(3)]

Example: VOLUME 3

Example: VOLUME 3SHELL 1.5

Example: VOLUME 2ZONE 5

Default: There is no default for the volume source option.

For TIGER, [parameter(1)] is the input-zone index.

For ACCEPT, [parameter(1)] is a body index. For a SPH or RCC body, the keyword **SHELL** may be used with [parameter(2)] specifying the inner radius of the spherical or cylindrical shell. For
ACCEPT, the body referred to need not be part of the actual zone description of the geometry. The keyword ZONE may be used with [parameter(3)] to specify an input zone that will be used to further specify the extent of the source. All source particles will be inside both the specified body and the specified zone.

59. PRINT-ALL

Syntax: PRINT-ALL Default: Only the cumulative results for the final batch will be written to the output file.

This primary keyword causes the cumulative results from each batch to be written to the output file. In parallel calculations, other control keywords may prevent every batch from being written to output.

60. PULSE-HEIGHT (ITS Only)

Syntax: PULSE-HEIGHT [parameter(1)] [parameter(2)]
Example: PULSE-HEIGHT47
Default: No spectrum of absorbed energy will be calculated.

This keyword causes the spectrum of absorbed energy to be calculated for input zones [parameter(1)] through [parameter(2)]. These parameters correspond to the order of the input zones as those zones were defined. If the parameters are left blank, the spectrum of absorbed energy will be calculated for the entire geometry. Certain biasing schemes, such as those activated by the sub-keywords SCALE-BREMS and SCALE-IMPACT, are inconsistent with this calculation; PULSE-HEIGHT will cause them to be deactivated (a message so informing the user is written to the output file). The following secondary keyword describes the energy bin structure used in tallying the spectrum of absorbed energy.

(a) NBINE Syntax: NBINE [parameter(1)] [keyword]
Example:
NBINE6USER
1.99999 1.0 0.5 0.25
0.00001 0.0 Default:
ten bins of equal width
plus total absorption
and escape (i.e., 12
bins total).

If [keyword] is not specified then [parameter(1)] is the
number of desired equal-width bins plus 2, to account for
both total absorption and escape. The highest energy bin is
only total absorption if the source is mono-energetic. If
[keyword] is USER, [parameter (1)] is the number of bin
energies to be read. The only choice for [keyword] is:

i. **USER** - User defined energy grid. The code will then read
the lower bounds of the energy bins (MeV) in descending
order as in the above example. The maximum lower
bound must be less than the maximum source energy.
In the above example, the first lower bound and the last
two lower bounds were chosen to ensure that total
absorption (full source particle energy absorbed in the
selected region) and total escape (no energy absorbed
in selected region for a given source particle),
respectively, would be accounted for.

Note that the primary keyword alone, with no other parameters or
keywords, will result in the calculation of the spectrum of
absorbed energy for the entire geometry using the default bin
structure.

61. **RANDOM-NUMBER**

S

y
n
m
p
l
e :
R A N D O M - N U M B E R

4
2
6
5
6
4
1
5
4
2
D e f a u l t
t :
0
( converted
to 5
1
9 )

[parameter(1)]isthe initialrandomnumberseedfortheMonteCarlorun. Thiskeywordcan
be used to start a run with the final random number from an earlier run.
For RNG1, this keyword can also be used in debugging to isolate the offending primary history. For a similar purpose, the more sophisticated user can use this keyword in conjunction with a print of the initial random number seed of a source particle, IRSAV.

For RNG2 and RNG3, the state of the RNG is more than a single seed, but this keyword can be used to specify a batch. See the RESTART-HISTORY keyword for a method of debugging
with those generators.

See the Random Number Generators section for further discussion of issues concerning random number routines.

62. READ-CONNECTIVITY

Syntax: READ-CONNECTIVITY
Default: Geometry connectivity information is read from a file.

Geometry connectivity information is read from the "fort.13" file. This can be a file generated by ITS using the DUMP-CONNECTIVITY keyword, or it can be a file generated by some other means. Incorrect connectivity information will not cause the code to give inaccurate results, but it may decrease the efficiency of the calculation.

The connectivity information used by ITS is based on each body entry in a zone description in the input geometry. Even if a single body is used numerous times in describing a zone or in describing multiple zones, the code maintains connectivity information for each usage of the body. The connectivity information indicates which zones a particle may enter upon
leaving the body definition. Thus, the connectivity file must contain at least one entry for each body-entry in the geometry description.

The format of the connectivity file is as follows: The first line must contain a single number indicating the number of connectivity values to follow. This must be at least two times the number of body entries in the zone definitions. The following data is the connectivity array and forms a kind of linked list. The 

76. REFLECTION-ZONE (ACCEPT and ACCEPTP Only)

Syntax: REFLECTION-ZONE [parameter(1)]
Example: REFLECTION-ZONE 20
Default: No reflection zone.

Particles undergo specular reflection at the boundaries of the selected zone. Source particles cannot be initiated in the reflection zone, and no particles can enter the reflection zone. This keyword cannot be used with EBFIELDS or with a CAD zone.

64. RESTART

Syntax: RESTART
Default: A new calculation is performed instead of a restart.

The problem is restarted at the batch number of the data in the “fort.14” file. A dump file must have been written, saved, and named “fort.14” (see FILE-NAMES keyword for alternative names). The batch size on the restart run must be the same as those on the dump file to permit accurate computation of statistical uncertainties. If specified otherwise, the batch size will be set equal to the batch size used to generate the dump file. The number of batches specified in the restart input file should
be the desired number of additional batches.
“Infill” batches represent gaps in the simulation that created the restart dump. If a calculation finishes successfully, there will be no infill batches in the dump file. If the dump file contains infill batches, those will be run in addition to the number of batches requested. No output of results will be written until all infill batches have been completed.

65. **RESTART-HISTORY (RNG2 and RNG3 Only)**

Syntax: RESTART-HISTORY
Default: A new calculation is performed instead of a restart.

This keyword specifies that a restart will be executed using the state of the random number generator specified in the file “rngstate.dump”. This file is written when the program exits due to an error in execution. It contains the state of the random number generator at the start of the particle history in which the error occurred. Only the offending history will be executed. This allows the user to repeat the single history in which an error occurs.

In parallel, the restart must be performed with static load balancing on one processor, since the RNG state is not passed to subtasks.
66. **SIMPLE-BREMS** (*MITS Only*)

Syntax: SIMPLE-BREMS
Default: more accurate bremsstrahlung distributions.

This keyword is intended for development purposes only. This keyword specifies that bremsstrahlung angular distribution is sampled from a simple approximation to the Bethe-Heitler formula instead of the combination of formulas by Koch and Motz.

67. **SOURCE-SURFACES** (*MITS Adjoint Only*)

Syntax: SOURCE-SURFACES [parameter(1)]
Example: SOURCE-SURFACES3
Default: All surfaces of the escape zone (tallied as a single surface) for ACCEPT.

Both surfaces (ZMIN and ZMAX) for TIGER.

This specifies the number of surfaces on which either ELECTRON-SURFACE-SOURCEs or PHOTON-SURFACE-SOURCEs will be tallied. These specified surfaces should correspond to adjunction-escape surfaces. This keyword must be followed by [parameter(1)] separate lines of the secondary keyword SURFACE.

Any specification for CAD will be ignored. All escaping particles are scored in a single body/surface tally.

(a) **SURFACE** Syntax: SURFACE [keyword] (non-ACCEPT codes) or SURFACE [parameter(1)] BODY [parameter(2)] (ACCEPT codes) Example: SURFACE ZMAX (non-ACCEPT codes) or SURFACE3BODY2 (ACCEPT codes) This sub-keyword specifies the surface index through which the
escaping particles will be calculated. For TIGER and CYLTRAN, surface 
**ZMIN** refers to the minimum-z surface and surface 
**ZMAX** is the maximum-z surface. For CYLTRAN, surface 
**RMAX** is the lateral escape surface at maximum radius. For 
ACCEPT, it is necessary to specify both the surface 

---

**Keywords** for IITS index and the **BODY** number. Figure 2 on page 77 illustrates how many of the body types used in ACCEPT have their surfaces numbered. In the case of the arbitrary polyhedron (ARB), the user explicitly specifies the order in the input. The right circular cylinder is numbered the same as the truncated right-circular cone (TRC). The rectangular parallelepiped (RPP) is numbered the same as the BOX where **A1** points in the positive x-direction, **A2** points in the positive y-direction, and **A3** points in the positive z-direction.

68. **SPECTRUM**

Syntax: SPECTRUM [parameter(1)] [keyword] [keyword] [keyword] 
Example: SPECTRUM5
Example: SPECTRUM5NUMBER-PER-BIN

1.00 0.80 0.76 0.53 0.00
5.0 4.0 3.0 2.5 2.0
Example: SPECTRUM5NUMBER-PER-BIN-PER-MEV

0.40 0.08 0.46 1.06
5.0 4.0 3.0 2.5 2.0
Example: SPECTRUM5NUMBER-PER-BIN-PER-MEV

0.20 0.04 0.46 1.06
5.0 4.0 3.0 2.5 2.0

Default: mono-energetic source (ITS) or mono-group source (MITS)

In forward mode, this keyword specifies that the energy distribution is a spectrum. [parameter(1)] is the number of energy grid points describing the spectrum (or one more than the number of energy bins in the spectrum). The spectrum follows this keyword, decreasing monotonically to 0.0. The corresponding spectrum energy grid is given on the next line. If the NUMBER-PER-BIN or NUMBER-PER-BIN-PER-MEV keywords appear, the code will convert the distribution to a cumulative distribution internally. The examples given above result in identical spectra.

In any of these formats, the spectrum does not have to be normalized (e.g., the cumulative distribution of the spectrum does not have to begin with 1.0). If the spectrum is not normalized, the code will produce a warning, normalize the spectrum, and proceed with the calculation. The distribution is normalized based only on the spectrum data provided. If a portion of the spectrum falls below the cutoff energy, source particles sampled from below the cutoff will not be tracked (but information about the fraction of such rejected particles will appear in the output).

The spectrum may contain line sources and ranges of the spectrum with zero probability, such as in the following example:

Example: SPECTRUM4

1.0 0.5 0.5 0.0

1.3325 1.3325 1.1732 1.1732

8 K e
The user may bias the energy sampling of source particles. If the **BIASED** keyword appears on the same line as the **SPECTRUM** keyword, the first spectrum read is the spectrum sampled from, and the second spectrum is the true spectrum of the source particles. Both spectra must be on the same energy grid. In the following example, the 1.3325 MeV line will be sampled 3 times more often by the code than the 1.1732 MeV line (but the particle weights will be adjusted to account for the fact that the true source has equal probability of each line).

Example: `SPECTRUM 4 BIASED`

```
1.0 0.25 0.25 0.0
1.0 0.5 0.5 0.0
1.3325 1.3325 1.1732 1.1732
```

In adjoint mode, the **SPECTRUM** keyword allows the detector response to be calculated by folding with multiple forward source spectra during a single calculation.

Syntax: `SPECTRUM [parameter(1)] ADJOINT-SPECTRA [keyword] [keyword]` Example:

```
SPECTRUM 2 ADJOINT-SPECTRA PHOTON
5
2
3
1
```
<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>.</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>1.00 0.80 0.76 0.53 0.00</td>
<td>14.2 13.1 10.1 8.6 5.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>.</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>.</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.</td>
<td>5</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>.</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>.</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>.</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>.</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Example: SPECTRUM1ADJOINT-SPECTRA
PHOTON NUMBER-PER-BIN 5
0.462 0.0924 0.5313 1.2243
14.2 13.1 10.1 8.6 5.4

Default: The detector response is only calculated by folding with a flat forwardspectrum.

In adjoint mode, the secondary keyword **ADJOINT-SPECTRA** must be included on the same line as the SPECTRUM keyword, as well as a keyword specifying the type of source particle as one of **PHOTON** or **ELECTRON**. [parameter(1)] specifies the number of forward spectra. [parameter(1)] lines must follow, each containing two parameters: the first parameter specifies the number of energy grid values (the number of bins plus one) in the corresponding spectrum, and the second parameter specifies the magnitude (or source strength) of the spectrum. Then, 2[parameter(1)] lists must follow. For each spectrum, the first list is the distribution, and the second list is the corresponding energy bin grid. The energies need not correspond to the energy divisions on the cross section set, however source energies cannot include energies for which cross sections are not available. The keywords **NUMBER-PER-BIN** or **NUMBER-PER-BIN-PER-MEV** can be added to the keyword line to specify those formats for the spectra, otherwise the format is assumed to be a cumulative distribution of the number of particles per bin.
WARNING: The results are always multiplied by the source strength factor. It is also possible to specify an unnormalized spectrum. While it is possible to apply a source strength factor to an unnormalized spectrum, this is not likely to be a desired feature. If no source strength factor is specified, none will be applied (that is, the factor will be set to 1.0).

The default is scoring with a flat forward spectrum and is included in the output if this keyword is not used. If this keyword is used to specify forward spectra, output will be generated based on folding with the forwardspectra. In either case results will be given in the energy bin structure specified with the NBINE sub-keyword of the SURFACE-SOURCE selected. However, in the case of the default, results will correspond with folding against a flat spectrum of unit strength for each energy span reported. That is, the unit strength is applied to each energy span, not to the entire energy span of the problem. If this keyword is used, results will correspond to the response due to source particles in the energy span reported.

69. TASKS (MPI Only)

Syntax: TASKS [parameter(1)] [parameter(2)]
[parameter(3)] Example: TASKS 50 100 1.1 Default: Number of processors available is used. Intermediary output after every batch. No termination due to stray processes.

This keyword applies only for parallel processing. [parameter(1)] specifies the number of processors to which batches can be distributed. If [parameter(1)] is negative, the master process will only distribute work and process results, but static load-balancing will still be used (unless the DYNAMIC-MPI keyword is used). [parameter(2)] specifies the number of batches between intermediary outputs. [parameter(3)] is a factor multiplying the running average of the batch time such that if any batch time
exceeds this time, the batch is considered to be in an infinite loop, and if all other processors have completed their calculations, the run will be terminated.

If the run is terminated for an assumed infinite loop, the random number seed for that batch is output so that, subsequently, that batch alone can be run by using that random number seed as the parameter for the RANDOM-NUMBER keyword. If the code itself detects an error condition that is not an infinite loop and calls ABORTX, the number of random numbers to the beginning of the offending history is output and the “rngstate.dump” file is written. The user can subsequently run only the offending batch (with the RANDOM-NUMBER keyword) or the offending history (with the RESTART-HISTORY keyword).

70. **TITLE**

    Syn
ta
    ax
    :
    T
    I
    T
    L
    E
    [data]
Example:

TITLE

Adjoint

Dose calculation
n
i
n
A
l
b
o
x
i
n
S
a
t
e
l
l
i
t
e
G
P
S
-
4
D
e
f
e
a
u
l
t
:


This keywordsignals that the next line of input contains [title data], which is a title of up to 80 columns that will be written to the output file and will be used as the title on any plots that are generated.

---

**71. TRAJECTORY-POINTS** *(CYLTRAN or ACCEPT Only)*

Syntax: TRAJECTORY-POINTS parameter(1)
Example: TRAJECTORY-POINTS 1000
Default: 500

Withthiskeyword, the user can change the maximum number of points plotted per particle track. A change is required
only when specified by ITS output. For CYLTRAN, this value will automatically be at least 4. For ACCEPT, this value will automatically be at least one more than the resolution for plotting curved edges (see the PLOTS-OVAL-RESOLUTION keyword).

9. Summary of ITS-CAD Keywords

This section contains a listing of keywords relevant to the prmfile used with the CAD codes. There are no defaults. The ITS Simulation section must always be included. The other sections may always be included, and they will only be read if the geometry engine has been included in the executable. More detailed descriptions of the syntax and use of these keywords are contained in the Keywords for ITS-CAD section.

<table>
<thead>
<tr>
<th>KEYWORD</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ITS SIMULATION</strong></td>
<td></td>
</tr>
<tr>
<td>ITS BEGIN</td>
<td>Marks start of section</td>
</tr>
<tr>
<td>INPUT FILE</td>
<td>Specifies the ITS input file</td>
</tr>
<tr>
<td>OUTPUT FILE</td>
<td>Names the ITS output file</td>
</tr>
<tr>
<td>MODE</td>
<td>Usage of CAD and CG geometry</td>
</tr>
<tr>
<td>KD TREE PARAMS</td>
<td>Parameters for efficiency grid</td>
</tr>
<tr>
<td>ITS END</td>
<td>Marks end of section</td>
</tr>
<tr>
<td><strong>ACIS GEOMETRY</strong></td>
<td></td>
</tr>
<tr>
<td>ACIS BEGIN</td>
<td>Marks start of section</td>
</tr>
<tr>
<td>ACIS END</td>
<td>Marks end of section</td>
</tr>
<tr>
<td><strong>CHOLLA GEOMETRY</strong></td>
<td></td>
</tr>
<tr>
<td>CHOLLA BEGIN</td>
<td>Marks start of section</td>
</tr>
<tr>
<td>CHOLLA END</td>
<td>Marks end of section</td>
</tr>
</tbody>
</table>

Last Modified Date: 2008/04/17 22:45:29
10 Keywords for ITS-CAD

This section contains the keywords for the prm file (parameter file). This input file is required when using the CAD preprocessor definition to compile. The ITS-CAD codes do not necessarily imply that CAD geometries are being used, since the MODE keyword can be used to specify a CG ONLY calculation. Refer to the section on Running ITS for more information on setting up the prm file section of a CUI file or using a prm file otherwise.

There are multiple sections of input in the prm file, with each section isolated by BEGIN and END keywords. The ITS simulation section is within the ITS BEGIN and ITS END keywords. It contains the names of the ITS input and output files and the simulation parameters. All keywords within the ITS section are required. Other sections specify the simulation geometry. If the executable includes a geometry engine (e.g., ACIS), then a section must be specified for that engine (e.g., ACIS BEGIN and ACIS END) even if that geometry section is empty.

All keywords should be entirely uppercase. For keywords with sub-keywords or parameters, the parameter or sub-keyword must appear on the same line as the keyword. Underscores do not denote spaces and must be included in the keywords.

Comments are allowed in a parameter file and are preceded with the # symbol. Anything to the right of a comment symbol is ignored. Extra characters after valid input is an error and will cause an error message and an abort. Blank lines are ignored. Text between sections of input is ignored. A missing * END card is an error and will cause an error message and an abort. A repeated or missing keywords is an error and will cause an error message and an abort. There are no default values. Missing data is an error, will cause an error message and an abort.

The geometry specified in the prm file may be assigned to zone numbers between 0 and 1 million. The zone numbers do not need to be sequential, and the numbering may contain gaps. However, the assigned zone numbers will be condensed for use by ITS. Thus, if 7 unique zone numbers are used in the prm file (e.g., 10 51 1201350), then ITS will consider these to be zones 1 through 7, numbered as (4751236). That is, in this example, the body assigned as 0 will be referred to as zone 1, the body assigned as 51 will be referred to as zone 7.
This keyword is required.

Syntax: INPUT FILE [keyword]
Example: INPUT FILE its.inp
Default: none.

This keyword specifies the name of the ITS input file as [keyword].

3. OUTPUT FILE

Syntax: OUTPUT FILE [keyword]
Example: OUTPUT FILE its.out
Default: none.

This keyword specifies the name of the ITS output file as [keyword].

4. MODE

Syntax: MODE [keyword]
Example: MODE HYBRID
Default: none.

This keyword specifies the transport mode as one of: CAD ONLY, HYBRID, MIRROR CG, or CG ONLY.

5. KD TREE PARAMS

Syntax: KD TREEPARAMS
We recommend using the parameter settings shown in the example, but users may wish to study the effect of other parameter settings on the efficiency (and accuracy) of their calculations.

parameter(1) sets the maximum depth of any branch of the kd-tree.

parameter(2) sets the maximum desired number of faces per leaf of the kd-tree. The code will attempt to refine the kd-tree until either this constraint is satisfied or the maximum tree depth has been reached.

parameter(3) sets the tree depth at which the cost-function heuristic changes from a solid-angle heuristic to an octree (midpoint). This is only active when the user has specified a solid-angle heuristic with [parameter(6)]=-1.

parameter(4) sets the axis flag, which determines upon which Cartesian axis to place the nextsplit-plane. A value of 0 or 2 starts with a simpler rotation through the 3 axes. Any other value starts with the longest axis of the cell which will be split. Additionally, the behavior is different for values greater than 1. For such values, the split-plane location will be accepted only if it “usefully” divides the axis in such a way that there is at least one object completely to the right and another object completely to the left of the split-plane. If such a “useful” split-plane is not found, such a split-plane will be searched for on the next (by rotation) axis. If all three axes do not result in a “useful” split-plane, the cell will not be divided any further. For parameter(4) values less than or equal to one, a split-plane will always occur on
the chosen axis (i.e., at the “default” location, if an optimized one cannot be
determined -see parameter(7)).

parameter(5) sets the split-plane flag, which is only used when the cost
function is something
other than an octree (see parameter(6)). For the value of 2, the split plane
is placed at a
midpoint between two nearest (but greater than 1.E-6 cm) bounding-box
boundaries. For
any other value, the split-plane is placed on one of the bounding-box
boundaries. At this
time, only the valueof2is functional. parameter(6) sets the cost function,
which determines the heuristic for placing the split-plane. A valueof0isan
costree, which simply means the split-plane is placed at the midpoint
of a lengthofa cell. A valueof-1 is the solid-angle heuristic of
Havran. Any othervalue,W, is a heuristic which minimizes the function
(W*Nsp+abs(Nleft -Nright)), whereNleft is the number of objects completely
to the left of the plane, Nright is the number completely to the right, and
Nsp is the number straddling the split-plane.

parameter(7) sets the default split-plane location, which is only used for
parameter(4) values less than 2. A zero value places it on the midpoint
(simlar to an octree), while 1 places it on the maximum boundary. The
valueof0is recommended.

6. **ITS END**
   
   Syntax: ITS END
   
   This keyword is required.

7. **ACIS BEGIN**
   
   Syntax: ACIS BEGIN
   
   This keyword is required if the ITS executable is built with an ACIS
   geometry engine included. Spatial Corporation markets the 3D ACIS
   Modeler as a commercially available product [34]. This version of ITS
   supports linking with the 3D ACIS Modeler version R12, R15, or R16,
which must be purchased directly from Spatial Corporation.

Between the ACIS BEGIN and END statements, the user may supply the ACIS geometry. Each line should contain the name of a file and the zone number to which the geometry belongs. For example: acispart.sat2 Each ACIS file may contain information for one zone or multiple zones. If the satfile contains more than one ACIS body and the user specifies a positive zone number, then the code will sequentially number each ACIS body as a unique zone number beginning with the specified zone number. If the user specifies a negative zone number, the code will use the absolute value and assign every ACIS body in the file to that single zone. It is also possible to assign information in multiple files to a single zone, such as: acispart1.sat3 acispart2.sat3 It is also possible to combine multiple geometry descriptions into a single zone description, such as describing some zone faces with Cholla facets and other zone faces with ACIS surfaces.

8. **ACIS END**

   Syntax: ACIS END

   This keyword is required if the ITS executable is built with an ACIS geometry engine included.

10. **CHOLLA BEGIN**

    Syntax: CHOLLA BEGIN

    This keyword is required if the ITS executable is built with a Cholla geometry engine included. The Cholla engine allows for the use of facet geometry in the format written from Cubit.

    Between the Cholla BEGIN and END statements, the user may supply the Cholla geometry. Each line should contain the name of a file and the zone number to which the geometry belongs. For example: /home/geometry/chollapart.fac3 Each Cholla file may contain information for only one zone, but a zone may be described by information in more than one file, such as: chollapart1.fac3 chollapart2.fac3 It is also possible to combine multiple geometry descriptions into a single zone description, such as describing some zone faces with Cholla facets and other zone faces with ACIS surfaces.

10. **CHOLLA END**

    Syntax: CHOLLA END
This keyword is required if the ITS executable is built with a Cholla geometry engine included.

11 TIGER Geometry

The geometry of the TIGER codes is the simplest of the ITS member codes. It is strictly one dimensional. A particle trajectory is described only in terms of the z coordinate of position and the z direction cosine. Nevertheless, this is often all that is necessary, and, because the TIGER codes are the fastest and simplest to use, they should always be considered. They are especially useful in obtaining accurate answers to questions involving very basic transport phenomena.

11.1 Problem Geometry

Beginning at \( z = 0 \), layers are stacked along the positive z-axis according to the order in which they are read in as described under Geometry Syntax. For each layer, the user must define: (a) the material index, (b) the number of subzones into which the layer is to be divided for purposes of scoring charge deposition, energy deposition, and particle flux (see the discussion of Automatic Subzoning), and (c) the thickness of the layer. The material indices are defined by the order in which the materials are specified in executing the cross-section generating code.

Interior voids must not be defined. Voids within the geometry have no effect upon one-dimensional transport. Omission of voids allows for increased efficiency in the calculation and is a requirement for TIGER geometry descriptions.
11.2 Conventions for Escaping Particles

In addition to quantities internal to the problem geometry such as charge deposition, energy deposition and particle flux, radiation that escapes may also be scored. Because geometry is defined as infinite slabs with only a finite z-dimension, particle escape is classified as one of:

1. Radiation that escapes from the maximum-z boundary of the problem.
2. Radiation that escapes from the minimum-z boundary of the problem.

In forward mode, these definitions may be applied by the user with the ESCAPE-SURFACES keyword to specify where escaping particles are to be tallied. In adjoint mode, the SOURCESURFACES keyword specifies where escaping adjunction particles are to be tallied. The default in either mode is to perform tallies on both surfaces.

11.3 Geometry Syntax

Syntax: GEOMETRY[parameter(1)]

[parameter(2)] [parameter(3)] [parameter(4)]

Example: GEOMETRY3

310.1

1
1
0

1
2
.
0

2
5
0
.
1
5

1
1
.

T
I
G
E
12 CYLTRAN Geometry

There are a variety of experimental problems in which the symmetry requirements of the CYLTRAN codes are satisfied to a good approximation. This is especially true in those experiments for which the radiation source is itself a cylindrical beam, as in the case of many pulsed and steady-state electron accelerators. The only essential requirement, however, is that the material geometry, as specified by the input zones, be cylindrically symmetric. The trajectories themselves are fully three dimensional. In a code modification, the more sophisticated user may wish to define a nonaxisymmetric source, in the case of CYLTRANM, a non-axisymmetric field configuration, along with whatever azimuthal tallies he desires. Note that the logic is already included for scoring azimuthally-dependent escape distributions (see keywords ELECTRON-ESCAPE and PHOTONESCAPE) and azimuthally-dependent charge deposition, energy deposition, and particle fluxes (see keywords ELECTRON-FLUX and PHOTON-FLUX, the remainder of this section, and the discussion of Automatic Subzoning).

12.1 Problem Geometry

The material geometry for the CYLTRAN codes consists of a right circular
cylinder of finite length, the axis of which coincides with the z axis of the Cartesian system that describes the particle trajectories. The location of this cylinder, hereafter referred to as the problem cylinder, along the z axis is completely arbitrary. The entire volume within the problem cylinder must be specified in terms of material or void input zones, each of which is bounded by two and only two cylinders coaxial with the z axis and two and only two planes perpendicular to the z axis.

The material configuration is then conveniently described by the half section of the problem cylinder obtained by passing a plane through its axis. An example of such a half section is shown in Figure 3. The horizontal base line is the axis of the problem cylinder, and the other horizontal lines are labeled by the radii of the corresponding cylindrical boundaries. The vertical lines are labeled by the z coordinate of the corresponding plane boundaries. The solid lines are actual material boundaries; the broken lines are not. The dashed lines are employed either to complete the perimeter of the problem cylinder half section or to break more complex zones of a given material (e.g., those having L-shaped half sections in Figure 3) into the simpler input zones required by the code (i.e., zones whose half sections are rectangles). The dotted lines describe subzoning of a given input zone for purposes of obtaining charge deposition, energy deposition, and flux profiles.

Each zone in Figure 3 is bounded by solid and/or dashed lines and contains a material index (circled). A zero index defines an avoid zone; otherwise, the material indices are defined by the order in which the materials are specified in executing the cross-section generating code. Each of these input zones requires a single input card for its description. The dotted lines illustrate subzoning of a particular input zone into equal axial and/or radial increments. Azimuthal subzoning is also possible as described under the Geometry Syntax section. It also follows from that discussion that separate input cards describing these subzones are not required. This description is accomplished internally by the code using the subzoning parameters specified on the input card describing the input zone. This feature allows the user to obtain three-dimensional charge deposition, energy deposition, and flux profiles within a given input zone with a single input card.

The following input cards describe the problem geometry illustrated in Figure 3.

GEOMETRY6

12.
CY
LT
RA
N
Ge
om
etry
Figure 3. Example of the half section of a problem cylinder

The numbers in the lower right hand corners of the subzones demonstrate how the code internally numbers these subzones. Subzone numbers are immediately assigned as each geometry card is read. Therefore, subzones are numbered before the next card is read. In the example, the second card, which describes the second input zone, generates 6 subzones (subzones 2-7). The next input zone, therefore, has a subzone number of 8. It is important that the user understand this numbering scheme in order to properly interpret spatially-dependent outputs.

12.2 Conventions for Escaping Particles

In addition to quantities internal to the problem cylinder, such as charge deposition, energy deposition, and particle flux, radiation that escapes from the problem cylinder may also be scored. Because the geometry is defined in
cylindrical coordinates of thickness z and radius r, particle escape is classified according as one of:

1. Radiation that escapes from the maximum-z boundary of the problem cylinder.
2. Radiation that escapes from the minimum-z boundary of the problem cylinder.
3. Radiation that escapes from the maximum- \( r \) curved lateral boundary of the problem cylinder.

In forward mode, these definitions can be applied by the user with the ESCAPE-SURFACES keyword to specify where escaping particles are to be tallied. In adjoint mode, the SOURCESURFACES keyword specifies where escaping adjunction particles are to be tallied. The default in either mode is to perform tallies at all three surfaces.

12.3 Geometry Syntax

Syntax: GEOMETRY[parameter(1)]

[parameter(2)] [parameter(3)] ... [parameter(9)]

Example: GEOMETRY3

-0.5 0.20 0.0 10.531150
0.2 12.45 0.0 10.510001
-0.5 12.45 10.5 12.052

Default: no default [parameter(1)] is the number of input zones. Immediately after the keyword line there must follow a series of [parameter(1)] lines, one for each input zone, containing [parameters(2)] through [parameter(9)]. These parameters specify the minimum z boundary, the maximum z boundary, the minimum \( \rho \) boundary, the maximum \( \rho \) boundary, the material, the number of \( \phi \) subzones, the number of \( \rho \) subzones, and the number of z subzones. All boundaries are given in cm. When the parameters trailing the material index are left blank, no subzoning is imposed.
13 ACCEPT Geometry

The ACCEPT codes provide experimenters and theorists with a method for the routine solution of coupled electron/photon transport through three-dimensional multimaterial geometries described by the combinatorial method. In the combinatorial scheme, the problem input zones are built up of primitive bodies, this is in contrast to more traditional schemes that define the zones in terms of bounding surfaces. We find the combinatorial method of specifying input zones in terms of solid bodies to be simpler, more intuitive, and less ambiguous than specification in terms of boundary surfaces. The combinatorial scheme also learns as the calculation progresses; at any particular time it makes use of information obtained from past experience in order to improve the efficiency of its search procedures used in particle tracking. This same learning ability precludes the requirement, typical of many other geometry schemes, for inputting a substantial amount of tracking information.

With the ACCEPT codes the user employs the combinatorial-geometry method in order to describe the three-dimensional material configuration of the problem. This task is accomplished in five distinct steps:

1. Define the location and orientation of each solid geometrical body required for specifying the input zones.
2. Specify the input zones as combinations of these bodies.
3. Specify zones to be subzoned and subzoning schemes, if necessary.
4. Specify the volumes of the subzones, if necessary.
5. Specify the material in each input zone.

The following sections discuss these five steps. Each section is divided into a general discussion of the geometry concepts and a more specific discussion of the data required by the code.

The data is inserted in free format form with spaces or commas as delimiters. Syntax is discussed in Sec. 13.7 and simple examples can be found in its/Tests/RegTests/Input.
Due to limitations in the accuracy of the geometry tracking algorithm used in ITS, we discourage using geometry with important features smaller than 1E-6 cm. In some cases, bodies with such small dimensions will trigger input errors.

13.1 Specification of Input Bodies

13.1.1 Body Definition

The combinatorial-geometry method requires a library of geometrical body types from which the user may choose in order to describe his problem configuration. The information required to specify each body type in a three-dimensional Cartesian system is as follows:

1. Rectangular Parallelepiped (RPP) – Specify the minimum and maximum values of the x, y and z coordinates that bound a rectangular parallelepiped whose six sides are perpendicular to the coordinate axes.

![Diagram of a rectangular parallelepiped with dimensions Xmin, Xmax, Ymin, Ymax, Zmin, Zmax.](image)
Figure 4. Rectangular Parallelepiped (RPP)

Figure 5. Sphere (SPH)

1 Sphere (SPH) – Specify the components of the radius vector \( \mathbf{V} \) to the center of the sphere and the radius \( R \) of the sphere.
2 Right Circular Cylinder (RCC) – Specify the components of a radius vector \( \mathbf{V} \) to the center of one base, the components of a vector \( \mathbf{H} \) from the center of that base to the center of the other base, and the radius \( R \) of the cylinder.
4. Right Elliptical Cylinder (REC) – Specify the components of a radius vector $V$ to the center of one of the elliptical bases, the components of a vector $H$ from the center of that base to the center of the other base, and the components of two vectors $R_1$ and $R_2$ that define the major and minor axes, respectively, of the bases.

1. Truncated Right-Angle Cone (TRC) – Specify the components of a radius vector $V$ to the center of one base, the components of a vector $H$ from the center of that base to the center of the other base, and the radii $R_1$ and $R_2$ of the first and second bases, respectively.

2. Ellipsoid (ELL) – Specify the components of the radius vectors $V_1$ and $V_2$ to the foci of the prolate ellipsoid and the length of the major axis $R$. This ellipsoid must be prolate – to
specify an oblate ellipsoid use the ELR (see next primitive) instead of the ELL. The foci of a prolate ellipsoid are on the major axis, the axis about which the ellipse is rotated to form the body. The center of the body lies halfway between the foci. The square of the length of the major axis equals the sum of the square of the length of the minor axis plus the square of the distance between the foci.
1 Ellipsoid of Revolution (ELR) – Specify the components of a radius vector $\mathbf{V}$ to the center of the ellipsoid, the components of vector $\mathbf{H}$ from the center of the ellipsoid to the apex along the axis of revolution, and the semi-axis length $\mathbf{R}$ in a direction perpendicular to the axis of revolution. Notice this ellipsoid maybe either prolate ($|\mathbf{H}| > \mathbf{R}$) or oblate ($|\mathbf{H}| < \mathbf{R}$).

2 Wedge (WED) – Specify the components of a radius vector $\mathbf{V}$ to one of the corners and the components of three mutually perpendicular vectors $\mathbf{a}_1, \mathbf{a}_2,$ and $\mathbf{a}_3$ starting at that corner and defining the wedge such that $\mathbf{a}_1$ and $\mathbf{a}_2$ are the two legs of the right triangle of the wedge. **Warning:** If the wedge is to be used as a subzoning body for CAD geometry the $\mathbf{a}_1$, $\mathbf{a}_2$, and $\mathbf{a}_3$ vectors must form a right-handed system.

3 Box (BOX) – Specify the components of a radius vector $\mathbf{V}$ to one of the corners and the components of three mutually perpendicular vectors $\mathbf{a}_1$, $\mathbf{a}_2$, and $\mathbf{a}_3$ starting at that corner and defining a rectangular parallelepiped of arbitrary orientation. **Warning:** If the box is

---

**Figure 10.** Ellipsoid of Revolution (ELR)
A Geometry to be used as a subzoning body for CAD geometry the \( \mathbf{a}_1, \mathbf{a}_2, \) and \( \mathbf{a}_3 \) vectors must form a right-handed system.

**Figure 11.** Right Angle Wedge (WED)

**Figure 12.** Box (BOX)

ArbitraryPolyhedron(ARB)—Specify the components of \( k \) \( (k = 4, 5, 6, 7, \text{ or } 8) \) radius vectors, \( \mathbf{V}^1 \) through \( \mathbf{V}^k \), to the corners of an arbitrary non-reentrant polyhedron of up to six sides, and specify the indices of the corners of each face by means of series of four-digit numbers between “1230” and “8765” (enter zero for the fourth index of a three-cornered face). The
digits must appear in either clockwise or counterclockwise order.

2 Torus (TOR) – The vector $\mathbf{V}$ specifies the coordinates of the centroid of the torus, the unit vector $\mathbf{H}$ specifies the axis of revolution, the major radius $R$ specifies the distance from the centroid of the torus to the center of the ellipse to be rotated, the radius $R_H$ specifies the axis of the ellipse parallel to the $\mathbf{H}$ vector, and the radius $R_p$ specifies the other axis of the ellipse. For now, only circular ($R_H = R_p$) tori are allowed.

13.1.2 Body Data

The body data begin immediately after the line containing the GEOMETRY keyword. The method of describing each of the body types is discussed in the body definitions section and
illustrated in Table 9. The description of each new body must begin a new line of input, and the first parameter on that line must be the appropriate three character code for the body type. Table 9 lists the additional input parameters required (no defaults) for each body type in their proper sequence. The user is free to distribute these parameters over as many lines as he pleases. A line with the keyword END signals that the description of all of the problem bodies is complete.

13.2 Specification of Input Zones

13.2.1 Input Zone Definition

Having defined the necessary geometrical bodies, the user must then resolve the entire problem geometry into input zones satisfying the following criteria:

1. An input zone may consist only of either a single homogeneous material or a void.
2. Every point of the problem geometry must lie within one and only one input zone.
3. The final input zone must be a void zone surrounding the rest of the problem geometry that is entered through a non-reentrant surface; any particle entering this zone is treated as an escape particle.

Input zones are specified as appropriate combinations of the previously defined bodies. Such combinations may be as simple as just a single body, or they may consist of complex intersections, unions and differences of various bodies. We illustrate the principles of input zone specification with the following examples where, for simplicity, we omit the escape zone. Each example involves only two zones, A and B, defined by the cross hatching in Figure 15.

In Figure 15a, zone A consists of a sphere, body #1, that is tangent to zone B, which consists of a right circular cylinder, body #2. Input zone specification is simply

A = +1,
B=+2. That is, inputzoneA consists of all spatial points that lie within body#1, and similarly for zone B. In Figure 15b, the sphere is inserted into a hole that has been cut in the cylinder so that

---

<table>
<thead>
<tr>
<th>Body Type</th>
<th>Real Data Defining Particular Body</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOX</td>
<td>Vx Vy Vz A1x A1y A2x A2y A2z A3x A3y</td>
</tr>
<tr>
<td>RPP</td>
<td>Xmin Xmax Ymin Ymax Zmin</td>
</tr>
<tr>
<td>SPH</td>
<td>Vx Vy Vz R</td>
</tr>
<tr>
<td>RCC</td>
<td>Vx Vy Vz Hx Hy R</td>
</tr>
<tr>
<td>REC</td>
<td>Vx Vy Vz Hx Hy R1x R1y R1z R2x R2y Hz R2z</td>
</tr>
<tr>
<td>ELL</td>
<td>V1x V1y V1z V2x V2y R V2z</td>
</tr>
<tr>
<td>TRC</td>
<td>Vx Vy Vz Hx Hy R1 R2 Hz</td>
</tr>
<tr>
<td>WED</td>
<td>Vx Vy Vz A1x A1y A2x A2y A2z A3x A3y</td>
</tr>
<tr>
<td>ARB</td>
<td>V1x V1y V1z V2x V2y V3x V3y V3z V4x V4y V5x V5y V5z V6x V6y V7x V7y V7z V8x V8y V8z</td>
</tr>
<tr>
<td></td>
<td>(see note below)</td>
</tr>
<tr>
<td>TOR</td>
<td>Vx Vy Vz Hx Hy R RH Rp Hz</td>
</tr>
<tr>
<td>ELR</td>
<td>Vx Vy Vz Hx Hy R Hz</td>
</tr>
<tr>
<td>END</td>
<td>No Data</td>
</tr>
</tbody>
</table>
Table 9. Data required to describe each body type

Note: The final line of the arbitrary polyhedron input contains a four-digit number for each of the six faces. Thirty data values are required for this body type; if there are fewer than eight corners and six faces, zero values must be entered.

A=+1,
B=+2-1. Thus, **input zone B consists of all spatial points that lie within body #2 AND not within body #1**. Input zone B is specified as the difference between two bodies.

In Figure 15c, bodies #1 and #2 consist of the same homogeneous material (or void), but they are imbedded within a second right circular cylinder, body #3, of another material. The specification is

\[ A=+1 OR +2, \]
\[ B=+3-1 OR +3-2. \]

Thus, **input zone A consists of all spatial points that lie within EITHER body #1 OR body #2**. This is an example of input zone specification as a union of bodies.

In Figure 15d, the intersection of body #1 and body #2 consists of a single homogeneous material; the rest of the space within body #3 is filled with another material. The specification is

\[ A=+1+2, \]
\[ B=+3-1 OR +3-2. \]

Thus, **input zone A consists of all spatial points that lie within body #1 AND within body #2**.
Note that:

1. The OR operator refers to all following body numbers until the next OR operator is reached or a new input zone is initiated.
2. The AND operator is implied before every body number that is not preceded by an explicit OR operator, except that the first OR operator in a union is an implied EITHER.

Though Figs. 15a and 15b are useful for demonstrating how input zones are constructed, they are not good examples of transport geometries because they are reentrant. By reentrant we mean that there are some paths by which escaping particles can reenter those geometries. They can be made non-reentrant by enclosing them completely in a non-reentrant body such as a sphere and letting the escape zone be the region outside the sphere.

### 13.2.2 Input Zone Data

Geometrical specification of the input zones begins immediately after the line containing the END parameter for the body data. The method of describing the input zones in terms of the input bodies is discussed in Sec. 13.2.1. Body numbers are determined by the order in which the bodies are read in. The description of each new input zone must begin on a new line of input, and the first parameter on that line must be a character string beginning with the letter Z. Anything following this Z, that is not separated by a delimiter, is ignored. Its sour convention is to follow the Z with the zone number to improve readability for the user, but the code will number zones in the order they are read regardless of the
numbering after the Z. The Z parameter is followed by a string of 1 3.

A CCEPT Geometry parameters that specifies the input zone following the form of the right hand sides of the equations of Sec. 13.2.1. For example, input lines describing the two input zones in Figure 15d are:

$$Z001 +1 +2$$
$$Z002 +3 -1 OR +3 -2$$

The user is free to distribute the parameters necessary for describing an input zone over as many lines as he pleases. A line with the keyword END signals that the description of all of the problem input zones is complete.

13.3 Subzone Specification

13.3.1 Subzone Definition

In Version 2.0 we began implementing automatic subzoning features into the ACCEPT codes. (See the section on Automatic Subzoning for further details.) In addition to reductions in memory requirements and runtime, this powerful option eliminates the burdensomeness of otherwise generating an input-zone description for each individual subzone. The ACCEPT codes now feature the full three-dimensional subzoning of input zones consisting of a single body of type RCC, RPP, BOX, SPH, WED, TRC, and TOR and subzoning for some multi-body input zones. Automatic subzoning is available for CAD zones based on RPP subzoning of the CAD bounding box. All of the available subzoning schemes can be used as non-conformal subzone overlays for obtaining simple profiles within complicated CG or CAD zones. Each subzoning scheme divides the sub-zone entity into equal intervals in three different dimensions based on three integers supplied by the user.

**Single Body Subzoning** The simplest type of subzoning involves a zone composed of a single body. The following is a description of how the three subzoning integers are used for these types:

1. **RPP** – The three integers correspond to subzoning along the three
Cartesian directions, x, y, and z, respectively. Here, the body-based coordinate directions are the same as those of the laboratory system. Distances are measured along the axes from the point (Xmin, Ymin, Zmin) defined in Figure 4.

2. **BOX** – The three integers correspond to subzoning along the three Cartesian directions, \( a_1, a_2, \) and \( a_3 \), respectively, as defined in Figure 12. Distances are measured from the point defined by the radius vector \( V \).

3. **WED** – The three integers correspond to subzoning along the three Cartesian directions, \( a_1, a_2, \) and \( a_3 \), respectively, as defined in Figure 11. This subzoning is similar to the BOX, except that subzones are cut by the sloped surface of the wedge. Distances are measured from the point defined by the radius vector \( V \).

4. **RCC** – The three integers correspond to subzoning azimuthally about the cylinder axis, in distance from the axis (radially), and in distance along the cylinder axis (axially) from the center of the base defined by the radius vector \( V \) in Figure 6, respectively.

5. **TRC** – The three integers correspond to the same subzoning definitions used for the RCC: azimuthal, radial, and axial.

6. **SPH** – The three integers correspond to subzoning azimuthally about the laboratory z axis as measured with respect to the positive x axis, in polar angle as measured with respect to the laboratory z axis, and in distance from the center of the sphere (radially), respectively.

7. **TOR** – The three integers correspond to subzoning in the poloidal angle, the cross section radius of the torus, and the toroidal angle (angle about the axis of revolution). This is analogous to the subzoning of an RCC, where the torus is a cylinder with its axis wrapped in a circle. Thus, the RCC azimuthal angle is the TOR poloidal angle, the RCC radial coordinate is the cross section radius, and the RCC axial coordinate is the toroidal angle. Currently, subzoning is restricted to circular tori.

The poloidal angle is the angle between the cross section radius vector \( r \) (currently, \( r = R_H = R_p \)) and the major radius vector \( R \) in Figure 14. The sense of rotation for the poloidal angles is such that a radius vector \( r \) in the direction of the unit vector \( H \) is a 90-degree poloidal angle. The poloidal
Subzones are created by dividing the 360-degree angle space by the number of poloidal subzones requested.

The subzoning of the cross section radius is currently limited to circular shells since the torus must be circular. The radial subzones are created by dividing the cross section radius equally into the number of cross section subzones requested.

The toroidal angle about the axis of revolution proceeds as the calculation of any azimuthal angle about an axis. In this case, the azimuthal angle is determined by the dot product of the perpendicular component of the radial vector \( \mathbf{R} \) with the reference vector. The toroidal subzones are created by dividing the 360-degree angle space by the number of toroidal subzones requested.

RPP subzoning RCC subzoning
Multi-Body Subzoning There are currently 8 multi-body subzone entities available in ITS. These subzone entities can be divided into 2 types: closed shells and cylindrical-like shells. The closed shells are the SPH-SPH and TOR-TOR. Both of these require that the two bodies be concentric. The SPH-SPH subzoning is shown in Figure 16. The TOR-TOR must be composed of tori that share an axis of rotation, as shown in Figure 17. Subzoning is based on the same coordinate systems as the single body subzoning. The number of “radial” subzones specify the number of subzone layers between the inner and outer boundary of the shell.

The other 6 multi-body subzone entities all use cylindrical-like body-based coordinates for subzoning: azimuthal, radial, and axial. The azimuthal and axial boundaries have their usual meaning. The radial subzone boundaries are equally spaced between the inner and outer zone boundaries for all axial
coordinates. This is best illustrated in the curved subzone boundaries of the SPH-RCC. For the SPH-RCC, the center of the sphere must lie on the axis of the RCC. In all other cases the bodies defining the inner and outer radii must be coaxial with each other. For the TRC-TRC, the wide parts of the frusta need not be on the same side.

Figure 16. SPH-SPH subzoning
Figure 17. TOR-TOR subzoning

RCC-RCC subzoning RCC-TRC subzoning TRC-RCC subzoning
Most of these zones are defined entirely as the subtraction of one body from another. However, two of these zones require a third body to complete the definition of the input zone. The definition of the SPH-RCC requires the union with another RCC, but the dimensions of the RCC are dictated by the SPH and RCC desired. The second RCC must provide an extension of the two planes perpendicular to the axis of the subtracted RCC and must be large enough in radius to encompass all of the subzoned region. The planes of the RCCs need not be symmetric about the center of the sphere. The RCC-TOR will typically require the subtraction of another cylinder with radius equal to the torus radius of revolution and other parameters identical to the first RCC. (This is done to eliminate the hole in the center of the torus.) If employing one of these entities as an explicit subzone overlay, then only the two bodies need to be specified.

**Automatic CAD Subzoning** Am method for subzoning CAD zones is provided. The user specifies the zone number that should be subzoned and the desired number of divisions in the x-, y-, and z-coordinates. ITS uses the bounding box of the CAD zone to create an RPP. This RPP serves as a subzone overlay for tallying purposes.

While this is a simple and automated subzoning technique, the resulting overlay may not be an efficient or reasonable subzoning scheme for the CAD zone. In some cases, a user may be able to manually specify a more efficient (nearly conformal) subzoning scheme for the CAD zone.
Subzone Overlays Subzone overlays (alternatively referred to as non-conformal subzoning) can be specified either implicitly or explicitly. The alternative is conformal subzoning. Conformal subzoning requires that the zone be defined by the same bodies as the subzoning scheme (with two exceptions stated in the description of multi-body subzoning that require an additional body to complete the zone description). Non-conformal subzoning requires that the zone be contained entirely within the subzone overlay. If this is not the case, errors can result due to invalid subzone indexes calculated during the transport process.

Implicit subzone overlays can be used by incorporating the subzone entity as the first body (or bodies) in the zone description. In this case, the overlay can be extracted from the zone description.

Explicit subzone overlays use different combinatorial descriptions for the input zone and the subzoning entity, and therefore use different geometrical descriptions for the transport process and for the subzone tallying process. In these terms, CAD subzoning is always explicit, but can be automated by having ITS convert the CAD bounding box into an RPP overlay.

13.3.2 Subzone Data

The automatic subzoning capability is invoked in the following way. In the subzoning section of the GEOMETRY data, the keyword SUBZONE must appear followed by a parameter that specifies the number of the zone to be subzoned. If the CG zone description begins with the body number(s) that define the subzoning scheme (as either a conformal subzoning or an implicit non-conformal subzone overlay) or if one wishes to automatically subzone a CAD zone based on the RPP of the bounding box, then nothing else needs to appear on this line. To impose an explicit subzone overlay upon either a CG zone or a CAD zone, the keyword OVERLAY must be followed by a combinatorial description of the overlay scheme. Only 1 or 2 bodies are needed for the OVERLAY description. The multi-body subzoning descriptions should consist of one body minus another body.

The line immediately following must contain three integers that define the number of equal-increment subzones into which the zone is to be divided along the three coordinate directions. The three orthogonal directions corresponding to these three integers are defined in Sec. 13.3.1.

Additional information is required for multi-body subzoning and for azimuthal subzoning. For multi-body subzoning, there is a set of tertiary keywords defining various input zones to be subzoned that consist of the difference between two bodies. In this case, one of the following keywords must follow the three subzoning integers: RCC-RCC, RCC-TRC, TRC-TRC,
TRC-RCC, SPH-SPH, TOR-TOR, SPH-RCC, and RCC-TOR.

To perform azimuthal subzoning on any cylindrical, conical, spherical, or toroidal subzone entity, a reference direction is required to define the zero azimuth. The reference direction for a SPH or SPH-SPH is defined as the positive x-axis. A reference direction may be supplied for other subzoning entities (RCC, TRC, TOR, RCC-RCC, RCC-TRC, TRC-TRC, TRC-RCC, TOR-TOR, SPH-RCC, and RCC-TOR) by reference to another body. To specify such reference direction, the input line with the three subzoning integers should contain the keyword **AZ**, followed by any body number except an RPP. The zero reference vector is the component of the V vector of the **AZ** body that is perpendicular to the H vector of the subzoned body. If no **AZ** body is specified, the +i vector will be used as the body vector. If the definition of the reference vector is null, the code attempts to use the +i, +j, and +k vector for the zero azimuth vector, in that order.

As an example, to impose an explicit non-conformal subzone overlay on input zone A in Figure 15c (zone number 1) based on the surrounding RCC body (body number 3), one would specify:

```
SUBZONE1OVERLAY+3 110 10
```

A line with the keyword END signals that the description of the problem subzoning is complete.

### 13.4 Volume Specification

#### 13.4.1 Volume Definition

The specification of volumes is optional. The code will attempt to normalize flux results by volume. The DEPOSITION-UNITS keyword allows the user to request that energy and charge deposition results be normalized by VOLUME or by MASS (density times volume). The code will default to using a volume of 1.0. The user should consider whether accurate volumes are needed for the normalization of the results that are of interest.
The volumes of the problem subzones may be specified through one of several automated or manual methods that are available. If automatic subzoning has not been requested for any input zone, the input zones are the same as the problem subzones. The volume of the escape zone is never specified. If the user wishes to supply the volumes for a run in which he has requested subzoning, he must ensure that the volumes are specified in the proper sequence.

A general scheme for the precise calculation of the volumes of zones defined by the combinatorial method is not possible. The user may select an option via \([\text{parameter}(1)]\) associated with the GEOMETRY keyword. The default value of 0 will cause the code to set all volumes to 1.0 cm\(^3\). A value of 1 allows the user to read in the volumes as described below. If the geometry is such that a satisfactory method exists for calculating the volumes internally, the user may set \([\text{parameter}(1)]\) equal to 2 and use a code modification to insert the necessary logic at the proper place in Subroutine VOLACC.

A value of 3 triggers the code to automatically calculate subzone volumes. The volumes of input zones that are not subzoned are set to 1.0 cm\(^3\). Volumes can be obtained for some zones. In CAD, the CAD-VOLUMES keyword will make the code calculate the volumes of all zones. In CG, a general volume calculation algorithm is not available. The volumes of zones for which conformal subzoning is available can be calculated by requesting only 1 subzone (1 interval in each of the 3 dimensions). However, the user needs to be aware that these volumes are only available for the simple single body or multi-body combinations for which subzoning is available. Attempts to calculate volumes for more complicated zones may result in non-conformal subzoning (see the discussion of Automatic Subzoning for more detail) and inaccurate volumes.

A value of 4 for the first parameter on the GEOMETRY keyword has the same effect as option 3, except that the code attempts to determine whether subzones are inside or outside of a zone (and adjust volumes accordingly) and the user may then overwrite subzone volumes for selected zones as described below. The method for determining whether subzones are outside of a zone is based on simply checking the mid-point of subzone against the zone definition. This method will be adequate if subzone boundaries conform to the boundaries of the zone in some manner. For subzone schemes that do not conform to zone boundaries, the user must supply the volumes for the intersection of the subzones with the zone.

If the value of \([\text{parameter}(1)]\) is negative, the logic for setting subzone volumes will proceed based on the absolute value of the parameter, but printing of the volumes to the output file will be suppressed.
13.4.2 Volume Data

If \( parameter(1) \) associated with the GEOMETRY keyword is equal to 1, the array containing the volume data for the problem subzones is inserted immediately after the line containing the END parameter for the data specifying the subzoning schemes. The input zones are numbered according to the order in which they are read. If an input zone is not to be subzoned, then it becomes a single subzone. Therefore, if automatic subzoning has not been requested for any input zone, the subzones are identical with the input zones (except that the escape zone is not included among the subzones). If an input zone is to be subzoned, the subzones are numbered by incrementing the body-based coordinates in an order corresponding to the inverse of the order of the three integers specifying the subzoning (see previous subsection). For example, if an input zone consisting of an RPP is to be subzoned, the subzones are generated by first incrementing the z coordinate, then they, and finally the x. The volume array must contain an entry for each problem subzone (no defaults), excluding only the escape zone.

If \( parameter(1) \) associated with the GEOMETRY keyword is equal to 4, then the code first calculates all subzone volumes analytically. For any subzone with a midpoint outside of the zone, the volume is set to zero. These volumes can be overridden for subzones of user-specified zones. These appear immediately after the line containing the END parameter which terminates the input zone descriptions. The first line should give the index of the desired zone. The following lines should include the subzone index (a dummy value that is ignored) and the subzone volume (in \( \text{cm}^3 \)). Any additional parameters will be ignored. There should be one line for each subzone, even for subzones that lie entirely outside of the original zone (that have zero volume). This format is repeated for each zone. The user can specify as many zones (that are subzoned, of course) as desired. This section is terminated with a line containing the keyword "END".

If \( parameter(1) \) is not equal to 1 or 4, these volume input data are omitted, and no "END" statement is required.

13.5 Material Specification

13.5.1 Material Definition

A material index is assigned to each input zone. A zero index defines a void zone; otherwise, the material indices are defined by the order in which the
materials are specified in executing the cross-section generating code. The method of inputting the material indices is described in the Geometry Syntax section.

**13.5.2 Material Data**

Materials are referred to only by index number. This index corresponds to the order the material cross sections were generated by the XGEN or CEPXS, which is the same as the order of the material requests in the input deck for those codes.

**13.6 Conventions for Escaping Particles**

Any particle entering the escape zone is assumed to have escaped. In CG, the user is responsible for constructing a geometry such that this is true. In CAD, the escape zone is automatically determined.

In forward mode, the ESCAPE-SURFACES keyword may be used to indicate the surfaces on which escape particle tallies are desired. In adjoint mode, the SOURCE-SURFACES keyword may be used to indicate surfaces on which escape adjuncton particles are to be tallied as sources contributing to the response. The default in either mode is to perform total escape tally (equivalent to a single surface surrounding the geometry) for all particles entering the escape zone. If using these keywords, the user is responsible for ensuring that the surfaces listed under these keywords are a complete description of escape surfaces in forward mode and a representation of the surface-source quantity desired in adjoint mode.

**13.7 Geometry Syntax**

Syntax: GEOMETRY[parameter(1-2)] [SUBZONE-ONLY] [VOID] [CAD-VOLUMES]
(Input Body Descriptions)
END

(Input Zone Descriptions)
END

(Subzoning Descriptions)
END

[parameter(3)]

Example: GEOMETRY01

RCC
0.0 0.0
0.0 0.0
0.0 1.0
0.25

RCC
0.0 0.0
0.0 0.0
0.0 1.0
0.5

S

P

H

0
.0
0

0
.0
0

0
.5

2
.0

S
\begin{align*}
\text{P} &\quad \text{H} \\
0 &\quad 0 \\
0 &\quad 0 \\
0 &\quad 0 \\
0 &\quad 0.5 \\
3 &\quad 3 \\
0 &\quad 0 \\
E &\quad N \\
D &\quad 1 \\
z &\quad +1 \\
z &\quad 2 \\
z &\quad +2 \\
- &\quad 1 \\
z &\quad 3 \\
z &\quad +3
\end{align*}
Z 4 
+ 4 
Z 5 
- 4 

END

SUBZONE1
110 10

SUBZONE2
11010 RCC-RCCAZ4

END
3
1
2
1
0

Default: no default This keyword signals the beginning of geometry input for the ACCEPT codes. The absolute value of [parameter(1)] determines the option for setting the subzone volumes in cm$^3$:
(a) 0 (default) causes all volumes to be set internally to 1.0.
(b) 1 causes the code to read volumes from the input stream.
(c) 2 requires that the user provide the necessary logic for computing the volumes at the appropriate place in subroutine VOLACC.
(d) 3 causes volumes of all subzones to be calculated, while the volumes of input zones that are not subzoned are set to 1.0 cm$^3$.
(f) 4 has the same effect as 3, but subzone volumes for user specified zones may be overwritten.

If [parameter(1)] is negative, printing of volumes to the output file is suppressed. Tracking debug is turned off or on according to whether [parameter(2)] is set equal to 0 (default) or not, respectively. **WARNING:** The tracking debug feature may produce substantial amounts of information. We recommend that a single history be simulated to assess the size of the resulting output file produced. The **SUBZONE-ONLY** sub-keyword causes ACCEPT to process input through the geometry, write a FINITE-ELEMENT-FORMAT file, and then quit. No further processing of input is performed and no particle transport is performed. This feature is used to produce a refined subzone structure without energy or charge deposition results. The resulting file can be used when mapping the results of a previous calculation performed on a coarse subzone structure onto this refined subzone structure. There is a tool to perform this mapping called MAPPER. The **SUBZONE-ONLY** feature is an exception to the rule that keywords are order independent. When the **SUBZONE-ONLY** keyword is used, no input will be read that appears beyond the data of the GEOMETRY keyword.

The **VOID** sub-keyword causes all zone materials assignments to be set to zero. The input for zone materials is still required; however, all material assignments will be disregarded. This feature is useful if performing a stochastic volume calculation.

The **CAD-VOLUMES** sub-keyword causes the code to calculate volumes of all CAD zones. Warnings will be included in the output for any part for which the volume cannot be calculated.
What follows the keyword line is the list of primitive bodies used to construct 
the combinatorial geometry followed by an END line, the list of input zones 
constructed from the 
1 
3 
.A 
.C 
.C 
.E 
.P 
.T 
G 
.e 
m 
.e 
. 
tr 
.y 
.primitive bodies followed by an END line, the list of subzoning specifications 
followed by an 
END line, and possibly a list of zone volumes depending upon the setting of 
[parameter(1)]. Immediately after this information, there must follow a series of 
lines, one for each input zone, with each containing [parameter(3)]. These 
parameters specify the material for each input zone.
1 
4 
.C 
A 
D 
G 
e 
om 
.e 
t 
.y 
. 

Last Modified Date: 2008/04/17 22:45:35

14 CAD Geometry

This document discusses the special requirements of the GEOMETRY 
keyword and special considerations when constructing or adapting a CAD
model to be used in ITS. The Cholla library provides a facet geometry capability with ITS for interrogating CAD models. Links are also available for coupling ITS with the ACIS modeler. Spatial Corporation markets the 3D ACIS Modeler as a commercially available product [34]. This version of ITS supports linking with the 3D ACIS Modeler versions R12, R15, and R16, which must be purchased directly from Spatial Corporation.

14.1 GEOMETRY Keyword

CAD geometries may have different requirements than combinatorial geometry (CG) depending on the MODE selected in the prm file. The alternative modes are CAD ONLY, HYBRID, MIRROR CG, and CG ONLY. For the MIRROR CG, and CG ONLY modes, the GEOMETRY keyword requirements are the same as the ACCEPT requirements when the code is not compiled with the CAD option. However, the GEOMETRY keyword has different requirements for the CAD ONLY and HYBRID modes.

There are several terms that must be defined to describe CAD ONLY and HYBRID geometry requirements:

The "escape zone" in CAD geometry is not strictly defined. A particle leaving a CAD geometry zone and not directed toward another CAD zone is in the escape zone. In HYBRID mode, the escape zone must be defined in CG and must surround all CAD zones.

The "undefined void" in CAD geometry is any region that is not otherwise defined in the CAD description. This is the void space between CAD zones, which is not explicitly defined.

The "defined void" in HYBRID mode is the CG equivalent of the CAD undefined void. It is the second-to-last zone specified and should define all CG space that is not otherwise defined. (It is possible to have other CG and/or CAD zones that are explicitly defined as being voids.) All of the space defined by CAD zones must be enclosed in this defined void.

14.1.1 CAD ONLY

In the CAD ONLY mode, combinatorial geometry features can be used to describe the source distribution. That is, the POSITION of the source (or in adjoint mode, the LOCATION of the DETECTOR-RESPONSE) may reference a body in the first section of the GEOMETRY input. Body information can also be used to specify subzone overlays. Other body and zone information may be present in the input file, but all CG zone definitions will be disregarded.

The material assignments section of the GEOMETRY keyword must be used to assign material to CAD zones. There must be a material assigned to each CAD zone plus 2 additional void assignments at the end of the list for the undefined void and the escape zone.

Sources may exist in the escape zone. However, if a source particle initiated in the escape zone does not intersect the problem, a warning will be printed to the output file stating that the source particle was rejected. A simple method for avoiding numerous warning messages (due to a source with an
angular distribution which only causes some particles to intersect the geometry) is to use the HYBRID feature to specify only a defined void and an escape zone such that the source is not in the escape zone.

14.1.2 HYBRID

Both CG and CAD geometry descriptions are used. The last two zones in the CG description must be a defined void and an escape zone. The defined void corresponds to CAD’s undefined void and must be specified as the escape zone minus all other space defined in CG. The escape zone must have boundaries such that a particle entering cannot possibly reenter the problem geometry.

If a particle is known to be in a CG zone, the CAD geometry will not be interrogated, and vice versa. Therefore, overlapping geometry descriptions may yield inaccurate results. Specification of non-overlapping geometries is user responsibility; no checks performed in the ITS code, and no errors will be generated.

Material indices in the GEOMETRY keyword should first assign materials to each of the CAD zones, then assign materials to each of the CG zones. Since a defined void and escape zone are included in the CG description, no additional material assignments are required.

14.1.3 MIRROR CG

This mode is intended for development purposes and is not recommended to users.

The CAD and CG models should describe exactly the same geometry. Any discrepancies in particle distance-to-boundary and particle zone location inquiries during transport will be printed to the output file. The transport will proceed using the CG results in the event of a discrepancy.

14.1.4 CG ONLY
Code compiled with the CAD option will execute as if the CAD option had not been used, except that the input and output files will be opened with names specified in the prm file.

**14.2 Conventions for Escaping Particles**

Any particle entering the escape zone is assumed to have escaped. In HYBRID and CG ONLY modes, the user is responsible for defining a CG escape zone such that this is true. In CAD ONLY mode, the escape zone is automatically defined.

The ESCAPE-SURFACES and SOURCE-SURFACES keywords are not functional for CAD geometry. All particles escaping the geometry are tallied into a single surface.

**14.3 CAD Models**

CAD models must be “healed” to be used in ITS. Gaps between CAD surfaces will result in lost particles in the transport process. Each time a particle is lost, a diagnostic is written to the output file. Large numbers of lost particles can result in inaccurate results and may indicate an inadequately healed geometry.

**14.4 Geometry Syntax**

The format of the geometry keyword for CAD is the same as for the ACCEPT codes. However, there are different requirements regarding what information must be present and differences in how the information is used.

Bodies must be specified if the code is not running in CAD ONLY mode. In CAD ONLY mode, CG bodies may be used to specify the spatial distribution of a source.

Zones are required if the code is not running in CAD ONLY mode. If zones are present in CAD ONLY mode, they are ignored. For mirroring, the zones
must be identical to the CAD description in both geometry and zone numbering.

Although bodies and zones are optional in CAD ONLY mode, the END statements must be included even if the sections are otherwise empty.

In HYBRID mode, CG zones may be superimposed on the CAD geometry. The escape zone must be specified as the last zone. The escape zone should surround the CAD and CG geometry. An “defined” void (corresponding to a CAD “undefined” void) must be specified as the second to last zone. This must encompass all space defined by CAD geometry.

Materials must be specified for zones in all modes. In CAD ONLY mode, materials should be assigned to all CAD zones, and two additional void assignments should be made for the undefined void and the escape zone. For HYBRID calculations, material assignments should be made for all CAD zones followed by material assignments for all CG zones. Since the undefined void and escape zone are part of the CG description, no extra assignments are required.

15. Suggestions
for Efficient Operation

15 Suggestions for Efficient Operation

The general operational limitations on the member codes of ITS are defined by the scope of the keyword input. An attempt has been made to implement automatic memory allocation for the arrays in ITS that are most likely to change from one calculation to another. Some arrays still have hard-wired dimensions. If while processing the keyword input, an array dimension required by a particular problem exceeds the default allocations, the execution aborts with a message to that effect being written to the output file. These dimensions can be found and modified in the its/Code/Modules/params.M file.

WARNING: Care must be taken in reducing a hard-wired allocation to zero since this may result in the upper bounds of the dimensions of certain arrays being set to zero; this will result in a fatal error since the lower bounds of the dimensions of most arrays is one.

Perhaps more important is the fact that the choice of certain input parameters can markedly affect the efficiency of the calculation; that is, the user's ability to obtain statistically meaningful output in a reasonable amount of time:

1. Obviously, the number of histories should be kept as small as possible. All member codes provide the user with estimates of the statistical uncertainties of the output data. Assuming that these uncertainties vary like the square root of the number of histories, these estimates then serve as a guide to the ultimate choice of the number of
histories. The user must decide what level of statistical accuracy is acceptable for his or her particular application.

2. To achieve good parallel efficiency, the number of BATCHES should be an integer multiple of the number of processors. The workload is distributed in batches. Having fewer batches than processors will result in some processors remaining idle. The first parameter of the TASKS keyword can be used to specify the number of processors desired or the code can determine the number of processors available.

3. In our experience dynamic load balancing, DYNAMIC-MPI, is only preferable over static load balancing in three situations: (1) the master process is located on a processor that should not be performing a large amount of computation but is preferable for I/O operations, (2) the processors on the system are expected to perform at different speeds so the calculation must be load balanced dynamically, or (3) the computer system is very unstable such that dynamic error handling is more efficient in making progress toward completing the calculation. In the first case, the user should consider using a negative value for the first parameter of the TASKS keyword.

4. The number of BATCHES should be at least 20 and should not be excessive. For calculation running on hundreds of processors for more than a day, it is not excessive to have thousands of batches. However, there is communication, processing, and output overhead associated with beginning and ending batches and having too many can affect the efficiency.

5. Electron cutoffs should be as large as possible. For example, if the source is monoenergetic, a global electron cutoff equal to 5 or 10 percent of the source energy may be adequate (depending upon the quantity of interest). Because the logarithmic energy grid used in the electron transport technique becomes much finer at low energies, following electron histories down to low energies becomes very time consuming.

6. Similarly, electron trapping energies should be as large as possible. For example, consider the simulation of photoemission by low-energy photon sources. Because accurate simulation of boundary crossings is important, electron cutoffs must be low. On the other hand, if bremsstrahlung production is not important, as is likely in this case, electron trapping energies may be as high as the maximum source energy.

7. The requested energy, angle, and spatial resolutions should be no higher than necessary. Demanding excessive resolution only makes it more difficult — i.e., costly — to obtain statistically meaningful output.

8. Processing large sets of tally data can affect the efficiency of the code.
This can impact the efficiency of the calculation by stressing the memory limits of the system, increasing time for processing results, increasing the time for parallel communications, or increasing the time for writing output. Users should consider whether all of the requested tallies are desired. For problems with large numbers of zones or subzones or with highly differential tallies, users should consider using the following keywords to manage I/O and memory usage: NO-INTERMEDIATE-OUTPUT, NO-DEPOSITION-OUTPUT, NO-SZDEPOSITIONOUTPUT, NO-DEPOSITION, NO-DETAILED-DEPOSITION, LINE-TALLY-WITH-CONTINUUM, or ANNIHILATION-LINE-TALLY-WITH-CONTINUUM.

Finally, the judicious use of a number of other variance reduction options can markedly increase the efficiency of certain calculations. Specific examples of these are discussed under the keyword BIASING. Some of these are discussed in even more detail in the section on Biasing. Users are warned, however, that the reckless and indiscriminate use of biasing procedures can lead to misleading results. Any use of biasing schemes should be carefully considered and scrutinized.

16 Output Files

This section describes in detail the information present in output files. The output file can be divided into three types of information: pre-processing information (anything preliminary to the Monte Carlo calculation), processing information (generated while the Monte Carlo calculation is performed), and results. There may not be any information generated while the Monte Carlo calculation is being performed. The output file is organized into sections generally designated by new carriage control pages ("1" in the first column). Each
section of the output file is presented separately here. The meaning of each statement or value (and its units) that may appear in the output file is discussed. Many sections discussed here may not appear in a given output file, as the output depends upon both the code options and output options requested.

16.1 Pre-Processing Information

16.1.1 Output Header

Next, the title “Program ITS” is given, followed by the version number and date of the code release. The authors of the software are listed, and author contact information is provided.

The next section states the preprocessor directives used to obtain the executable with which the calculation was performed.

A numerics interrogation section is included. This is mostly for testing purposes, to determine whether test failures may be due to different numerical representations on a new platform or using a new compiler.

16.1.2 CAD Parameters (CAD Only)

If the CAD preprocessor directive is used, the flow logic integer requested for the calculation is stated with a table for determining the meaning of the flow logic integer. Also, the number of CAD zones that have been read is stated.

The prm file is echoed.

16.1.3 Reading Input

Unless the “ECHO 0” keyword is used, input is echoed to the output as it is read. This provides the user with information about when and why an error occurs during the reading of input. However, for large input files, it may be desirable not to echo the input to the output file, so as to minimize the size of the output file. The entire input file is included in a job file, so this section may be redundant.

Minimal processing is performed while the input is being read, but some processing information may be reported in this section as it is performed. An example of this is (for ACCEPT) information on subzone volumes and how they were obtained. For CAD calculations, negative volumes indicate that the volume of the subzone intersected with the CAD zone and the volume of the CG subzone were in agreement. This means that the subzone lies entirely within the zone, and therefore CG-based electron trapping logic may be activated in that subzone. The negative values are merely informative, and absolute values are used for all result normalizations.
16.1.4 Reading Cross Section Data

For ITS (not MITS), this section includes:

The title of the XGEN calculation that produced the cross section file. The number of cross section sets available in the cross section file is printed. This is the number of materials for which data was generated using XGEN.

Foreach cross section set in the file the following parameters (used by XGEN to produce the cross sections) are stated:

- The density of the material in g/cm$^3$.
- The “detour” of the material. This is the ratio of the practical range to continuous-slowing-down-approximation range for an electron at the maximum energy of the set.
- The “IBL” factor is Seltzer’s empirical modification to the Blunck-Leisegang formulation for sampling from a truncated collisional energy-loss straggling distribution for electrons.
- For each element in the material:
  - Z is the atomic number of the element.
  - A is the mean atomic weight of the element.
  - W is the weight fraction of the element in the material.
- ITRM is the level of data contained in the cross section file. If less than 5, ITS cannot be run.
- ISGN is the cross section model: 1=Mott Electron, 2=Mott Positron, 3=Screened Rutherford Electron, 4=Screened Rutherford Positron. This should always be 1, unless the XGEN code has been modified.
- ISUB is the number of electron substeps taken per step in the condensed history algorithm. If the number of substeps per step is allowed to vary across the energy grid, then ISUB applies only to the first energy span. The XGEN default is for ISUB to remain constant across the energy grid, but see INDEX/JSUB under the DATAPREP DATA for more information. The XGEN keyword SUBSTEP can be used to control the ISUB value.
– INAL is the option used in XGEN for calculating eta in the Mott elastic cross section.
– ICYC is the option used in XGEN for generating the electron energy grid. ICYC=1 means that a logarithmic scheme has been used (see the next parameter), and this should always be the case unless the XGEN code has been modified.
– NCYC is the parameter determining the spacing of the electron energy grid. Successive energies are related by \( E_{i+1} = \frac{1}{N_{CYC}} \cdot E_i \). The XGEN keyword STEP can be used to control the NCYC value.
– NMAX is the number of electron energy grid values. The XGEN keyword ELECTRON-GRID-LENGTH can be used to control the NMAX value.
– EMAX is the maximum energy in MeV of the electron energy grid. The XGEN keyword ENERGY can be used to control the EMAX value.

\[ 1 \]
\[ 6 \]

Output

Files

– EMIN is the minimum energy in MeV of the electron energy grid.
– RMAX is the maximum electron range in g/cm\(^2\) in the material.
– MMAX is the number of angular bins in the multiple scattering distribution. The XGEN keyword ELECTRON-ANGLE-BINS can be used to control the MMAX value, as well as the distribution of angular bins.
– The value of INDEX is stated. This is a three digit number. If the first digit IDST is 1, then the cumulative multiple scattering distributions are to be read into ITS from the cross section set. If the second digit IAVE is 2, then the average cosines for the multiple scattering distributions are to be read. If IAVE equals 1, then ITS cannot be run. If the third digit JSUB is 2, then the number of substeps per step for the condensed history algorithm is allowed to vary across the energy grid, in which case the number of substeps is read from the cross section set.

A list of the data sets read from the cross section file for each material.

For MITS, this section includes:
The first line of the title of the CEPXS calculation.

- The number of materials for which data is available in the cross section set. This is the number of unique material compositions for which data was generated using CEPXS.

  The number of unique materials (both composition and density) labeled as material-densities. This is greater than or equal to the number of unique material compositions. (Some data is identical for materials with the same composition but different densities, and less memory is required by taking advantage of this.)

  The number of energy groups for all species.

  The particle species coupling scheme used to generate the cross sections.

  If electrons are included in the cross sections, the electron energy group structure in MeV.

  If photons are included in the cross sections, the photon energy group structure in MeV, and information about fluorescence lines, if any are included in the photon cross sections. Fluorescence information includes the element and shell-transition generating the line.

For each material:

- The density of the material in g/cm$^3$.
- The “detour” of the material. This is the ratio of the practical range to continuous-slowing-down-approximation range for an electron at the maximum energy of the set.
- For each element in the material:
  - $Z$ is the atomic number of the element.
  - $A$ is the mean atomic weight of the element.
  - $W$ is the weight fraction of the element in the material.
The scheme used to generate Fokker-Planck scattering-angles. The scattering angles may be energy-and material-dependent to mimic ITS substeps, or they may be constant. Either way, the maximum scattering angle will be written to the output (as the cosine of the angle).

The treatment of positrons. Positrons may be tracked, not tracked, or treated as electrons. In the last two cases, annihilation will occur at the pair interaction site.

If photons are included in the cross sections, the group into which annihilation radiation is produced, and whether that group is a line group.

16.1.5 Processing Input

This section will only contain information if errors or warnings are generated while processing input and cross section data. Checks are performed for inconsistencies and other errors in input keywords, cross section data, energy ranges, etc. These messages are generally preceded by “>>>>>” and include the words “ERROR” or “WARNING”. A warning is generally generated instead of an error if an inconsistency has been resolved by the code in such a way as to allow the calculation to proceed. The user should always check for such warnings to determine if the calculation performed was the desired calculation.

16.1.6 Storage Requirements vs. Allocations

This section compares the size of array dimensions required with the allocation for those arrays. The user may be able to decrease some array allocations to cope with memory constraints. Generally, an error will have been generated before this point if an array has insufficient allocation. Hard-wired array dimension parameters are specified in its/Code/Modules/params M.F and may be over-ridden in the calc params routine.

16.1.7 Geometry-Dependent Input

Geometry-dependent biasing settings and material assignments are listed for each zone.

For TIGER and CYLTRAN, the spatial extent of each zone is then listed. Since this section can be large if the geometry consists of many zones, the NO-GEOMETRYTABLE keyword can be used to suppress this section of the output.

16.1.8 Source Information

The energy, spatial, and directional distribution of the source is stated. In adjoint, this is the distribution of the adjunction source (as specified by the DETECTOR-RESPONSE).
16.1.9 Output Options

The number of batches and histories per batch are stated.

Optional output requests are stated here. For each differential quantity requested, the binning structure is stated. For example, if ELECTRON-FLUX is requested, then the energy, polar-angle, and azimuthal-angle binning structure for the electron flux is stated. Keywords that will trigger binning information to be printed here are: ELECTRON-or PHOTON-in combination with -SURFACE-SOURCE, -VOLUME-SOURCE, -ESCAPe, or -FLUX, ELECTRON-EMISSION, and PULSE-HEIGHT.

16.1.10 Physical Options

The options used for modeling physics are listed. This includes physics for which input keywords provide switches or cross section scaling, and physics that may or may not be included, such as fluorescence lines and positron annihilation.

16.2 Monte Carlo Output

16.2.1 Parallel Processing (MPI Only)

The following information is printed before the Monte Carlo calculation begins:

Whether the load distribution is static or dynamic. This is directly related to use of the DYNAMIC-MPI keyword.
Number of processors. This is the number of processors determined to be available.
Master option. If equal to 1, then the master process performs Monte Carlo batch calculations. If equal to 0, then the master does not perform Monte Carlo calculations. The latter is always the case for dynamic load balancing.
Number of tasks(requested). This is the number of processors that the user requested.
Number of tasks (adjusted). If the number of processors requested was greater than the number of processors available, then the number actually used is stated.

Intermediate print. This is the frequency (in number of batches) with which the intermediate output will be written.

Allowed time factor. This number (if greater than zero) is the factor by which batch times are allowed to deviate from the average batch time. That is, if a batch requires more time than the average batch time multiplied by this factor and all other processes are completed, then the run is terminated.

The following information is printed during the Monte Carlo calculation:

- For static load balancing calculations at the start of each cycle, the number of tasks performing Monte Carlo calculations and the initial random number seed.
- The subtask number and the random number seed assigned to initiate a batch. This is printed for the master task (subtask 0) and for batches corresponding to the intermediate output print frequency.
- The task number, the number of random numbers used, and the batch time. This is printed when the master finishes a batch or when a batch, corresponding to the intermediate output print frequency, returns its results to the master.

### 16.2.2 Monte Carlo Errors

Errors that occur during the Monte Carlo calculation will appear before the start of the results. This information does not have a section header. The information will appear to be at the end of the physical options section, or for **MPI** it will appear to be part of the parallel processing section. These messages generally are preceded by “>>>>>”, include the word “ERROR”, and list information that may be useful for understanding the error.

### 16.3 Results

Except in the diagnostics tables containing accounting information, every output quantity is followed by a one-or two-digit integer that is an estimate of
the one-sigma statistical uncertainty of that quantity expressed as a percentage of the quantity. Details of the method used to obtain these statistical data are given in the Statistics section.

Almost all quantities in the results section are carried over during a dump and restart. Exceptions to this are random number information that is specific to the latest batch or cycle and some timing data that is specific to the latest batch or current run.

16.3.1 Diagnostics

Timing information:

The percent of the problem completed. In the output file, this states that the problem is 100 percent complete, unless the PRINT-ALL keyword is used. This can be a useful marker to determine the progress of a calculation, either in the intermediate output file or in the output file if the PRINT-ALL keyword is used.

The number of batches remaining in the calculation is printed.

The average time per batch is printed.

Random number seed and usage information:

The initial random number seed of this cycle. In serial, a cycle is the entire run. For MPI with static load balancing, a cycle is one set of batches performed in parallel. For example, to perform 20 batches on 5 processors requires 4 cycles of 5 batches each. For MPI with dynamic load balancing, a cycle has no meaning, and the random number seed reported is the seed for the next batch.

The initial random number seed of this batch. In serial, this batch is the last batch calculation performed. For MPI, this batch is the last for which a set of results is received by the master task.

The initial random number seed for the next batch. This is the seed that would be used if the calculation continued to perform another batch. This seed will be used if a restart is performed. This seed may be used with the RANDOM-NUMBER keyword to initiate a calculation using an independent series of random numbers.

The number of random numbers generated in this cycle.

Cumulated number of random numbers generated. This is the total number of random numbers used in the current run. This counter is reinitialized when the NEW-DATA-SET keyword is used, but it will be maintained through restart.

Average source energy in MeV. In adjoint, this refers to the adjuncton source. This includes tallies for the energy of source particles that are determined to have energies below the cutoff energy that are not tracked.

For ACCEPT sources that include zone rejection, several diagnostics are provided on the efficiency of the source location sampling.
For calculations that include electric or magnetic fields, a list of tallies precedes the standard diagnostics tables. These are mostly self-explanatory. An electron trapped in a magnetic mirror in a void will never terminate “naturally”, so when such a condition is identified the electron is terminated, and a counter indicates the number of times this occurred. The “corner problem” arises in some cases when a particle’s curved trajectory in a field would go around a corner and approach a surface tangentially. An algorithm for coping with the corner problem has been implemented, and the counter records the number of times the more computationally expensive algorithm is invoked.

For CYLTRAN the number of times that a particle is rejected due to geometry problems is reported.

For ITS CYLTRAN and ACCEPT the number of times that “kicking” is terminated due to geometry errors is reported. Kicking is the termination procedure for electrons in the continuous-energy codes. It consists of moving the electron a random fraction of its residual practical range in rectilinear motion.

For ITS (not MITS), there are two diagnostics tables. The history table reports the number of events simulated:

- **PRIM** – primary histories simulated.
- **SEC** – secondary electron histories simulated. This is the sum of KNOCK, PE, PAIR, COM, and AUGER. An electron history is not simulated unless the electron is produced above the cutoff energy and is not immediately trapped.

- **KNOCK** – knock-on electrons simulated.
- **PE** – photo-electrons simulated.
- **PAIR** – pair-production electrons and positrons simulated.
- **COM** – Compton-produced electrons simulated.
- **AUGER** – Auger-produced electrons simulated.
- **BREM** – bremsstrahlung photons simulated.
RAD – unscaled bremsstrahlung events sampled to account for radiation energy-loss straggling of electrons.

XRAY – fluorescence photons simulated.

REJ. LAND – number of times that sampling of electron energy-loss straggling was rejected, either to preserve the mean energy loss or because the energy loss for the step would have been greater than the initial electron energy.

REJ. PEAL – number of times that sampling of the photo-electron emission angle yielded a scattering cosine with an absolute value greater than one. (The scattering cosine was set to 0.99999.)

PRIM STEPS – condensed history steps simulated for primary electrons.

SEC STEPS – condensed history steps simulated for secondary electrons.

O

u

t
p
u

NBLK – number of times that sampling of electron energy-loss straggling would have resulted in electron energy gain. (Energy loss was set to zero.)

INCOH. SCAT – incoherent (Compton) photon scattering events.

COH. SCAT – coherent photon scattering events.

The second ITS table reports the number and energy of secondaries produced. This includes particles produced below cutoff that are not tracked, except in the case of knock-on electrons. The ENERGY is the mean energy in MeV of this type of secondary produced per primary history. The AVE ENERGY is the average energy in MeV of this type of secondary produced per interaction that produced this type of secondary. The NUMBER/PRIMARY is the estimated mean number of secondaries of this type that would be produced per primary history in an unbiased calculation (i.e., without cross section scaling). The NUMBER GENERATED is the number of production events simulated by the code.

FIRST KNOCK – knock-on electrons produced by primary electrons.
TOTAL KNOCK – knock-on electrons produced by all electrons.
PHOTO-ELECTRON – photoelectric interactions producing electrons.
PAIR – pair production interactions producing electrons and positrons.
COMPTON – incoherent Compton scattering events imparting energy to electrons.
AUGER – Auger electrons produced by either photon or electron interactions.
FIRST BREMSSTRAHLUNG – bremsstrahlung photons produced by primary electrons.
TOTAL BREMSSTRAHLUNG – bremsstrahlung photons produced by all electrons.
X-RAY(P-IONIZATION) – fluorescence x-rays produced by photon interactions. For standard codes (not PCODES), this applies only to K-shell fluorescence x-rays.
X-RAY(E-IONIZATION) – fluorescence x-rays produced by electron interactions. For standard codes (not PCODES), this applies only to K-shell fluorescence x-rays.
ANNIHILATION QUANTA – photons produced by positron annihilation.

For MITS, the history diagnostic table reports the number of events simulated. All tallies disregard non-unity particle weights that may arise due to biasing.

Primary histories – primary particle histories initiated. (In adjoint, these are adjunct source particles and may include a mix of particle species.)
Unscattered primary photons – photons that do not interact from initiation to escape.
Electron escape scores – all electrons entering the escape zone through any surface.
Photon escape scores – all photons entering the escape zone through any surface.
Cutoff scores – all particles slowing down (in adjoint, speeding up) pasta cutoff energy.

Boltzmann and FP interactions – any type of scattering or
absorption interaction with the medium. The breakdown of these interactions follows with descriptions indented.

- **FP interactions** – electron Fokker-Planck angular scattering interactions.
- **Electron elastic interactions** – electron angular scattering interactions without correlated energy loss.
- **Photon coherent scattering** – photon angular scattering interactions without energy loss.
- **Electron inelastic interactions (non-absorption)** – an electron interaction from which a particle emerges, other than Fokker-Planck or elastic scattering. As the transport is performed, it is possible to have an electron interaction produce a photon but not an electron, since the particles emerging from an interaction are sampled without preserving the identity of the original particle.
- **Photon inelastic interactions (non-absorption)** – a photon interaction from which a particle emerges, other than coherent scattering.
- **Electron inelastic interactions (absorption)** – an electron interaction from which no particles emerge.
- **Photon inelastic interactions (absorption)** – a photon interaction from which no particles emerge.

**CPSL too small for scor/cpsl** – In adjoint only, if an isotropic surface-source is desired, this tally records how many times an escaping adjuncton had an angular cosine too small to allow for the necessary angular weighting of the score. Currently, the angular cosine limit is

\[ 1 \times 10^{-13} \]

**Non-physical upscatters (in adjoint, non-physical downscatters)** – If cross sections are regenerated using differing energy group structures for different particle species, then it is possible to have a cross section for producing particles in one group (A) from another group (B) that extends lower in energy. If a particle has an energy in group B below the lower energy bound of group A, and the sampling of an interaction produces a particle in group A, then the particle produced must have a higher energy than the interacting particle that produced it. This is non-physical. Such interactions are allowed, but they are tallied so that the user may know how many such interactions occur in a simulation.

**Exponent argument less than \(-88\) in forcing logic** – In **CYLTRAN** and ACCEPT only, if collision forcing is used for photons, the code may attempt to use an exponential argument less than \(-88\). The value \(-88\) is used instead to avoid underflow errors on some platforms. This parameter (defined as C88 in the Modules/params M.F file) can be adjusted.

**Exponent argument less than \(-88\) in sampling distance to collision** – In **TIGER**, the problem described in the previous bullet can arise whether collision forcing is used or not. Exponent argument less than or equal to \(-88\) in next-event logic, score was not tallied. If the NEXT-EVENT-ESCAPE keywords used (or if the logic is used due to a PHOTON-ESCAPE or PHOTON-SURFACE-SOURCE request), then scores are not tallied for which the probability of
escape is small (and may cause an underflow error on some platforms). Note that because the tally is proportional to the probability of escape which is small, this is unlikely to affect results. The C88 parameter can be modified to determine the effect. Using a value

\[
16
\]

Output

Files

greater than \(88\) (C88 less than \(88\)) may result in a speed up, since ray-tracing is halted for any escape path when the probability falls below \(C88\).

Exponent argument greater than \(88\) in next-event logic, score was tallied. This tally is the complement of the previous tally. If next-event logic is used for photon escape, then the sum of these two tallies should equal the number of photon escape scores (unless there have been lost particles during the next-event logic).

The number of source particles and total energy in MeV of source particles rejected for being below the cutoff energy. (In adjoint, this means adjuncton source particles.) Because energy spectra are allowed to extend below global cutoff energies, this tally allows the user to assess the significance of source particles that were not tracked and whether the cutoff energy must be lowered to achieve an accurate simulation.

For the PCODES, a table of the number of electron impact ionizations is given for each shell, followed by a similar table for photoelectric ionizations.

16.3.2 Integral Electron Emission (ITS ACCEPT Only, ELECTRON-EMISSION keyword)

For each surface on which electron emission is calculated, the number and energy of electrons is reported. This is the average number per source particle and the average cumulative electron energy per source particle. Tallies are only recorded for electrons with energies above the electron cutoff energy.
For each surface, a row in the table is labeled by the surface index and body index supplied by the user. The last row of the table reports tallies for any electrons emitted into the cavity through a surface not specified by the user. Any tallies in this row may be an indication of an error in the problem specification.

16.3.3 Integral Escape (Forward Only)

For a photon source, the number and energy escape fractions for unscattered primary photons is printed.

For ITS (not MITS), the number escape fraction (that is, number per source particle) is given for electron generated secondary electrons (E-SEC), photon-generated secondary electrons (P-SEC), and annihilation photons. The number escape fraction is given for x-rays generated in each material. For the non-PCODES, x-rays are only generated for the K-shell. This information is given for each surface requested or all surfaces by default (see the ESCAPE-SURFACES keyword for more information).

The next section contains tables of the number and energy escape fractions for each particle species. For all codes, information is given for electrons and photons. For MITS, information is also given for positrons. For ITS, positrons are counted as electrons here. This information is given for each surface requested or all surfaces by default (see the ESCAPE-SURFACES keyword for more information).

For ITS, the following values are stated:

- Energy escape fraction below cutoff. These electrons and positrons are not included in any other output tallies.
- Net charge escape fraction below cutoff. These electrons and positrons are not included in any other output tallies.
- Net charge escape fraction above cutoff. This quantity may be different than the number escape fraction given for electrons in the above table,

Output Files

- Net charge escape fraction above cutoff. This quantity may be different than the number escape fraction given for electrons in the above table,
because this quantity assigns a negative weight to escaping positrons.

The energy of cutoff photons assumed to have escaped is stated. This quantity should be zero, unless the CUTOFF-PHOTONS-ESCAPE keyword has been used.

16.3.4 Boundary Currents (TIGER Only)

The electron boundary currents at each material interface are reported. For both transmission (particles going in the positive-z direction) and reflection (particles going in the negative-z direction), the number currents and energy currents are listed with the percent statistical uncertainty. The currents will include electrons below the cutoff energy, unless the NO-KICKING keyword is used.

Positrons are scored with electrons in this tally, in the same way and also with positive weights.

16.3.5 Energy and Charge Deposition (Forward Only)

A new section is initiated for every 50 subzones, and header information is repeated, unless one of the NO-DEPOSITION-OUTPUT or NO-SZDEPOSITION-OUTPUT keywords are used. Results are normalized for a single source particle. The total energy and charge deposition and three subdivisions of each are provided, unless the NO-DEPOSITION or NO-DETAILED-DEPOSITION keywords are used. For ITS, the subdivisions are based on the physics resulting in the deposition. For MITS, the subdivisions are based on the mechanics of the code. For ITS (not MITS), the four columns are PRIM, E-SEC, P-SEC, and TOTAL.

PRIM is deposition directly by the primary particle. For photon sources the PRIM energy deposition should be zero, since only electrons deposit energy.

E-SEC is the redistribution of energy or charge due to the secondaries of an electron (or positron), i.e., knock-on electrons or Auger electrons produced by electron (or positron) ionization. In ITS, the primary electron energy-loss associated with production of knock-on electrons is not directly correlated with the production of knock-on electrons. All primary energy-loss is recorded under PRIM. The E-SEC tally records a negative energy deposition for the production of a knock-on electron and records positive energy deposition for all subsequent energy-loss interactions by the secondary electron. Therefore, since this is a redistribution tally, it is not uncommon to find negative energy deposition tallies for E-SEC.

P-SEC is the redistribution of energy and charge due to the secondaries of a photon, e.g., a Compton electron. Note that this includes the energy of a photo-electron produced below the cutoff energy and therefore not tracked. As with the E-SEC tally, it is not uncommon to find negative energy deposition tallies for P-SEC, since this is a redistribution tally.

TOTAL is all deposition in the subzone. This should equal the sum of the other three.

For MITS, the four columns are MICRO, TRACK, FLCUT, and TOTAL.
MICRO is deposition due to microscopic deposition tallies. This should only be non-zero if the MICRO keyword is used.

Track is deposition due to flux-fold tallies. If using the MICRO keyword, this is only due to continuous-slowing-down of electrons.

FLCUT is deposition by particles falling below the cutoff energy or electrons being trapped.

TOTAL is all deposition in the subzone. This should equal the sum of the other three.

For TIGER, the energy deposition section appears before the charge deposition section. Energy deposition is given in MeV cm$^2$/g. Charge deposition is in electrons cm$^2$/g. The columns for both sections report the subzone number, the material number for the subzone, the minimum-z edge of the subzone in cm, the maximum-z edge of the subzone in cm, the minimum-z edge of the subzone in g/cm$^2$, the maximum-z edge of the subzone in g/cm$^2$, and the four deposition values (with percent uncertainties). At the end of each table is a row of totals across all subzones in the problem.

For CYLTRAN, the energy deposition section appears before the charge deposition section. Energy deposition is given in units of MeV, and charge deposition is given in units of electron charge (i.e., one electron deposits a charge of 1.0, and one positron deposits a charge of -1.0). The columns for the energy deposition table report the subzone number, the material number for the subzone, the mass of the subzone in grams, the volume of the subzone in cm$^3$, and the four deposition values (with percent uncertainties). The columns for the charge deposition table report the subzone number, the material number for the subzone, the minimum-z edge in cm, the maximum-z edge in cm, the inner radius edge in cm, the outer radius edge in cm, the lower azimuthal boundary in degrees, the upper azimuthal boundary in degrees, and the four deposition values (with percent uncertainties). At the
end of each table is a row of totals across all subzones in the problem.

For ACCEPT, energy and charge deposition information are contained in the same table. Unless the DEPOSITION-UNITS keyword is used, energy deposition is given in units of MeV, and charge deposition is given in units of electron charge (i.e., one electron deposits a charge of 1.0, and one positron deposits a charge of -1.0). The first column contains the subzone number. If subzoning is not activated, the zone number is the same as the subzone number. If subzoning is activated in the problem, the start of each zone is marked by a line stating the “INPUT ZONE NUMBER”, and for the zones that are subzoned, an additional line states the total deposition quantities for the zone and the following line notes how the subzone numbers are incremented within the zone. The second column states the material number for the subzone. The remaining columns report the four deposition values (with percent uncertainties) for energy deposition and then for charge deposition. At the end of the table is a row of totals across all subzones in the problem. The totals are not normalized by the DEPOSITION-UNITS scaling factors, so they are always in units of MeV for energy deposition and electron charge for charge deposition.

The energy conservation fraction is the sum of all energy accounted for by escape and deposition divided by the average source energy.

The charge conservation fraction for an electron source is the sum of all charge accounted for by escape and deposition in units of electron charge. The charge conservation fraction for a photon source is one minus the sum of all charge accounted for by escape and deposition. For either source, this statistical quantity should converge to one.

16.3.6 Particle Flux (Forward Only)

All flux estimates are obtained via a track-length tally and are normalized to one source particle. Results are normalized by the volume of subzone, the energy interval (for energy differential output.

Files
tables), and the angular interval (for angle differential tables). The volume used for this normalization may not be the actual volume of the subzone, depending upon the geometry volume option requested. Unless a negative GEOMETRY flag is used, the volumes of all subzones are printed in the Reading Input section of the output (see section 16.1.3), and these volumes are used for the normalization. WARNING: In ACCEPT, automatic subzone volume calculation may not accurately describe the intersection of subzones with the zone when nonconformal subzone overlays are used. In CYLTRAN and TIGER, subzone volumes are always accurate because of the simplified geometries.

In TIGER, the energy spectrum flux is stated in units of #/MeV, the energy spectrum and angular distribution flux is stated in units of #/MeV/sr, and total angular distribution flux is stated in units of #/sr. In CYLTRAN and ACCEPT (if the subzone volumes are accurate) the energy spectrum flux is stated in units of #/cm$^2$/MeV, the energy spectrum and angular distribution flux is stated in units of #/cm$^2$/MeV/sr, and total angular distribution flux is stated in units of #/cm$^2$/sr.

**Electron (ELECTRON-FLUX keyword)** For ITS (not MITS), the “electron left at flux cutoff energy” is listed before the energy spectrum of the electron flux. This is given as a table of subzone numbers. The first row contains the subzone numbers. The second row (in columns aligned with the subzone numbers) contains the number of electrons per source particle left at the flux cutoff energy in each subzone and the associated percent uncertainty. Each row contains up to ten subzone quantities.

A table of “energy spectrum of electron flux” appears for each subzone of the zones requested with the ELECTRON-FLUX keyword. For each energy interval requested, the electron flux and percent uncertainty are stated.

A table of “energy spectrum and angular distribution of electron flux” is given for each sub-zone of the zones requested. For ACCEPT and CYLTRAN, the azimuthal interval is stated in the header for each table. For TIGER, the azimuthal interval is always implied to be 0 to 360 degrees, since it is a one-dimensional code that cannot resolve the azimuthal direction. Column headers state the polar (theta) interval in degrees for the corresponding header. (If the direction-space option is used, the azimuthal and polar information appear in the opposite locations.) The energy interval associated with the flux values is stated at the start of each row. The flux value and percent uncertainty is then stated for the energy interval given at the start of the row, the polar interval stated above the column, and the azimuthal interval stated in the table header.

At the bottom of each energy-angular flux table, the total angular distribution of flux is stated for the entire energy interval for which flux was tallied. This total energy interval is stated at the start of the row. Again, the angular intervals correspond to the polar interval in the column header and the azimuthal interval in the table header.

**Photon (PHOTON-FLUX keyword)** A table of “energy spectrum of photon
“flux” appears for each subzone of the zones requested with the PHOTON-FLUX keyword. For each energy interval requested, the photon flux and percent uncertainty are stated. The flux from continuum radiation and line radiation are listed separately. The continuum radiation appears in the same format as the electron flux values. At the start of each row containing line radiation information, the energy of the line is stated. Annihilation radiation (if applicable) appears first. Then, line radiation fluxes are given for each possible transition in each material. The transition producing the line radiation is stated with the line energy at the start of each row. (Only K-shell transitions are simulated in the standard ITS codes. The PCODES allow more detailed line radiation. For line radiation fluxes)

16.3.7 Spectrum of Absorbed Energy (ITS Only, PULSE-HEIGHT keyword)

The energy intervals for recording tallies are listed at the start of each row of data. The number of tallies and percent uncertainties in the estimate follow. The results are in units of number of tallies per MeV, and are normalized to a single history (i.e., a single source particle).

It is common practice to have the first energy interval and last energy
interval be very small. The first energy interval, spanning a range from the source energy to an energy slightly smaller than the source energy, records tallies for the total absorption of source energy within the detector. The last energy interval, spanning a range from slightly larger than zero energy to zero, records tallies for no source energy absorption within the detector.

The spectrum of absorbed energy is a pseudo-pulse height distribution. It differs from a true pulseheight distribution in that IT $\textsc{S}$ is a Class I electron transport code, meaning that it does not correlate electron energy loss with the energy imparted to knock-on electrons. (Energy loss due to bremsstrahlung production is sampled with correlation to secondary production.) Electron energy loss is sampled based on straggling distributions to determine the energy lost by an electron. The energy lost is deposited in the medium. The production of knock-on electrons is sampled separately. Energy initiating a knock-on electron is removed from the medium. In a given history, it is possible to have more energy removed from a subzone than is deposited, but statistically the energy deposition is accurate as the number of histories is increased. Pulse height distributions are tallies on a per-history basis. It is possible to have physically unrealistic negative absorbed energy tallies. A diagnostic following the absorbed energy table states the number of counts that were rejected due to negative energy deposition.

16.3.8 Electron Emission (IT$\textsc{S}$ ACCEPT Only, ELECTRON-EMISSION keyword)

The first line of each header states that the table reports the energy spectrum and angular emission distribution of electrons. The second line states which surface of which body the electron emission has been tallied for, and if applicable also reports the subsurface index for that surface. Results are normalized for a single source particle and are given in units of $#/\text{MeV}\cdot\text{sr}$ (number of electrons emitted divided by both the energy interval and solid angle interval). On the fifth line, the azimuthal interval is stated (or polar interval, if direction-space binning is used).

Column headers state the polar (theta) interval in degrees for the corresponding header. (If the direction-space option is used, the azimuthal and polar information appear in the opposite locations.) The energy interval associated with the electron emission values is stated at the start of each row. The emission value and percent uncertainty is then stated for the energy interval given at the start of the row, the polar interval stated above the column, and the azimuthal interval stated in the table header.

```
Output
```
16.3.9 Escape Spectra (Forward Only)

All escape estimates are obtained by tallying particles entering the escape zone. For ACCEPT, the accuracy of the escape tallies depends upon an adequately defined escape zone (e.g., the geometry should be non-reentrant). Escape results are normalized to one source particle, the energy interval (for energy differential tables), and the angular interval (for angle differential tables).

Separate tables of escape information are given for each surface requested. For ACCEPT, the default is to report the escape information"through all surfaces" in an integrated table. For TIGER and CYLTRAN, the default is to report escape information separately for each of the 2 or 3 possible escape surfaces. See the ESCAPE-SURFACES keyword for more information.

The energy spectrum escape is stated in units of /MeV, the energy spectrum and angular distribution escape is stated in units of /MeV/sr, and total angular distribution flux is stated in units of /sr.

Electron (ELECTRON-ESCAPE keyword) For each surface requested with the ESCAPE-SURFACES keyword, two tables of data may be given depending upon the resolution requested with the ELECTRON-ESCAPE keyword.

A table of "energy spectrum of escaping electrons" is written first. For each energy interval requested, the electron escape and percent uncertainty are stated.

A table of "energy spectrum and angular escape distribution of escaping electrons" is presented next (if angular binning was requested). For ACCEPT and CYLTRAN, the azimuthal interval is stated in the header for each table. For TIGER, the azimuthal interval is always implied to be 0 to 360 degrees, since it is a one-dimensional code that cannot resolve the azimuthal direction. Column headers state the polar (theta) interval in degrees for the corresponding header. (If the direction-space option is used, the azimuthal and polar information appear in the opposite locations.) The energy interval associated with the escape values is stated at the start of each row. The escape value and percent uncertainty is then stated for the energy interval given at the start of the row, the polar interval stated above the column, and the azimuthal interval stated in the table header.

At the bottom of each energy-angular escape table, the total angular distribution of escape is stated for the entire energy interval for which escape
was tallied. (This total energy interval is stated at the start of the row.) Again, the angular intervals correspond to the polar interval in the column header and the azimuthal interval in the table header.

Escaping positrons are included in the electron escape tallies.

**Photon (PHOTON-ESCAPE keyword)** For each surface requested with the ESCAPE-SURFACES keyword, two tables of data may be given depending upon the resolution requested with the PHOTON-ESCAPE keyword.

A table of “energy spectrum of escaping photons” is written first. For each energy interval requested, the photon escape and percent uncertainty are stated. The escape of continuum radiation and line radiation are listed separately. The continuum radiation appears in the same format as the electron escape values. At the start of each row containing line radiation information, the energy of the line is stated. Annihilation radiation (if applicable) appears first. Then, line radiation escape is given for each possible transition in each material. The transition producing the line radiation is stated with the line energy at the start of each row. (Only K-shell transitions are simulated in the standard ITS codes. The PCODES allow more detailed line radiation. For line radiation escape in MITS calculations, line radiation groups must be present in the cross section set, and therefore must be requested in the CEPXS input.)

16.

Output

Files

A table of “energy spectrum and angular escape distribution of escaping photons” is presented next (if angular binning was requested). The continuum and line radiation are stated separately. The table format is like the electron energy-angular escape, with the addition of line radiation as in the energy spectrum of escaping photons.

At the bottom of each energy-angular escape table, the total angular distribution of escape is stated for the entire energy interval for which escape was tallied. The total includes both continuum and line radiation.
16.3.10 Sources and Responses (*Adjoint Only*)

In adjoint, a separate output section exists for each source energy spectrum. If no forward source spectrum is specified, results are based on the default flat energy spectrum (i.e., a unit strength flat energy spectrum is applied to each energy range for which results are reported). An output header is written for each set of output, and a new set of output is initiated if more than 8 columns are required. Each adjoint output header first describes the detector:

**Detector response type (DOSE, KERMA, CHARGE, ESCAPE ELECTRONS or ESCAPE PHOTONS)**

Units of the response values. For TIGER or for ACCEPT surface-sources, the units are (MeV/g)/(source-particle/cm2) for dose or kerma, (electrons/g)/(source-particle/cm2) for charge, or number/source-particle for escape. For ACCEPT surface sources with an escape detector, the units are (number/cm2)/(source-particle/cm2). For ACCEPT volume-sources, the units are (MeV/g)/(source-particle/cm3) for dose or kerma, (electrons/g)/(source-particle/cm3) for charge, or (number/cm2)/(source-particle/cm3) for escape.

For escape, the energy range of particles detected.

For escape, the angular extent of particles detected.

The location of the detector. (Surface for escape; point or volume for deposition.)

For deposition, the material for the detector.

Next, the sources are described in a header section that applies to all detector-response values that follow:

The location of the source. For ACCEPT, the default is that surface sources are “through all surfaces” of the escape zone.

The angular distribution of the source (isotropic, cosine-law, or delta0-ave).

The angular extent of the source in degrees. For energy intervals only, the source extends over the full angular range. For energy and angle intervals, separate headers may be provided for sources extending over only a portion of the azimuthal range (normally) or polar range (for direction-space).

If results are normalized to one source particle, that is stated. Otherwise, the source strength is stated.

Finally, sources are described by row and column labels that apply only to detector-response values in the corresponding rows and columns:

```
1 6
```

Output
File

Row labels state the source energy ranges in MeV for each detector response value.
Column labels state the polar range (normally) or azimuthal range (for direction-space) of the sources in degrees.

16.3.11 CAD diagnostics (CAD, not CG ONLY mode)

Source particle location:

- Total calls - The number of times a source particle zone number was requested (should equal the number of histories).
- Percent right (mirroring only) - The percentage of agreement between CAD and CG.
- Percent wrong (mirroring only) - The percentage of disagreement between CAD and CG.
- Percent on boundary - The percentage of source particles for which the zone number could not be determined because the particle was located on a boundary.
- Percent unknown - The percentage of source particles for which the zone number could not be determined.
- Percent bad - The percentage of source particles for which CAD experienced a failure while trying to determine the particle location.
- Percent rejected - The percentage of source particles rejected due to failure to determine the zone number.

Distance to boundary:

- Total calls - The number of times the distance to a boundary was requested.
- Percent right (mirroring only) - The percentage of agreement between CAD and CG.
- Percent wrong (mirroring only) - The percentage of disagreement between CAD and CG.
- Percent no-hits - The percentage of attempts that failed to locate a boundary in the direction requested.
- Percent rejected - The percentage of calls that resulted in a particle being rejected due to failure to find a valid distance to boundary. This is not the same as a percentage of source particles rejected, since some of these rejected particles may be secondary particles and may have a variety of weights due to biasing.

The sidestepping functionality is currently disabled due to concerns about consistency with the geometry tracking logic. The following diagnostics remain as an indication of how efficient the algorithm might be.

- 1 sidestep attempted - The number of times that, because a boundary could not be located or the source particle location could not be
determined, the particle was pushed $1 \times 10^{-7}$ cm perpendicular to the
direction of flight, and the distance to a boundary was attempted to be
determined.

1.6.

Output

Files

2 sidesteps attempted– The number of times that, because the first
sidestep failed, the particle was pushed $1 \times 10^{-7}$ cm perpendicular to the
direction of flight and perpendicular to the direction of the first sidestep from
the location of the particle before the first sidestep was attempted. The
difference between this value and the “1 sidestep attempted” value is the
number of times that the first sidestep succeeded and transport continued with
the particle on a slightly altered path.

3 sidesteps attempted – The number of times that, because the
second sidestep failed, the particle was pushed $1 \times 10^{-7}$ cm perpendicular to
the direction of flight (in the opposite direction of the first sidestep).

4 sidesteps attempted – The number of times that, because the third
sidestep failed, the particle was pushed $1 \times 10^{-7}$ cm perpendicular to the
direction of flight (in the opposite direction of the second sidestep).

Sidesteps failed – The number of times that a
distance to boundary could not be determined either along the original
particle path or along 4 different offset particle paths.

Boundary crossing location:

Total calls– The number of times that the zone number of a particle
location was requested after a particle crossed a boundary.

Percent right (mirroring only) – The percentage of agreement between
CAD and CG.

Percent wrong (mirroring only) – The percentage of disagreement
between CAD and CG.

Percent on boundary – The percentage of particles for which the zone
number could not be determined because the particle was located on a
boundary.

Percent unknown – The percentage of particles for which the zone
number could not be determined.
Percent bad – The percentage of particles for which CAD experienced a failure while trying to determine the particle location.

Percent rejected – The percentage of particles rejected due to failure to determine the zone number.

16.3.12 Timing Data

All times are reported in seconds. In serial calculations, the timing uses the function cpu time if it is available. In parallel calculations, the timing uses wall-clock time. A statement appears in the output indicating whether a cpu timer or wall-clock timer was used.

Batch times record only the time during which the Monte Carlo calculation is being performed and exclude all I/O, pre-processing, and post-processing. In a RESTART calculation, only the cumulative Monte Carlo time, average batch time, and average history time include the times of the previous run(s).

Startup time (s) – For CAD calculations, this is the time required to process the CAD geometry and input.

Allocation sizing time (s) – This is the time to read input and cross sections to determine necessary array sizes and to allocate arrays.

Input processing time (s) – This is the time to read input and cross sections and perform preprocessing before initiating the Monte Carlo calculation.

Batch initialization time (s) – This is the time required to initialize variables before each Monte Carlo batch calculation. In a parallel calculation, this time is only computed on the master process.

Output processing time (s) – This is the time required for post-processing and writing output, including intermediate output. In a parallel calculation, this time is only computed on the master process.

Monte Carlo cycle time (s) – This is only meaningful if using MPI with static load-balancing. This is the sum across all cycles of the longest batch time for each cycle.

Longest Monte Carlo time (s) – This is only meaningful if using MPI. This is the longest batch time across all batches.
Elased execution time(s) – This is the total runtime for ITS. This includes I/O, preprocessing, Monte Carlo, and post-processing within the current run.

Cumulative Monte Carlo time (s) – This is the sum of all batch times.
Averaged Monte Carlo batch time (s) – This is the average time required to perform a batch calculation for all batch calculations performed.
Average Monte Carlo history time (s) – This is the sum of all batch times divided by the number of Monte Carlo histories simulated.

17. PCODES 140

Last Modified Date: 2008/04/17 22:45:43

17 PCODES

The SANDYL[14] code is a three-dimensional multimaterial code. Its construction was oriented toward relatively low-energy photon sources and the understanding of internal electromagnetic phenomena in complex geometries. In particular, it includes a detailed modeling of atomic shell ionization and relaxation phenomena for electron and photon energies down to 1.0 keV. The early codes of the TIGER series, on the other hand, were developed primarily for relativistic electron beam applications, where atomic shell effects usually play only a minor role and are, consequently, treated in a more cursory fashion. Nevertheless, these potential low-energy limitations were purely incidental, and there was no reason why the more complete description of atomic shell effects available in the SANDYL code could not be included in the codes of the TIGER series. Those codes in ITS that include the more detailed ionization/relaxation physics from the SANDYL code are referred to as the PCODES. The PCODES contain the logic necessary for describing ionization and relaxation of all K, L1, L2, L3, M(average), and N(average) shells having binding energies greater than 1.0 keV for elements with atomic numbers Z=1 to Z=100.[14][12]

Once a photoionization or electron impact ionization event has occurred, several different relaxation cascades are possible. A large quantity of atomic relaxation data is required for the stochastic description of these cascades. These data, together with the photoionization probabilities, are tabulated in Ref. [12], which also includes a discussion of the cross sections for electron impact ionization and details of the implementation of these processes in the PCODES.

The standard codes only include a description of the electron impact ionization of the K-shell of the highest-atomic-number element in a given material. Similarly, following either this impact ionization or a photoelectric event, these codes only model relaxation processes (production of Auger electrons and fluorescent photons) from this same shell.

18.
Electric
and
Magnetic
18 Electric and Magnetic Fields

In many instances the value of strictly collisional transport models is questionable because the actual experiments involve macroscopic electric and magnetic fields whose effects upon radiation transport not only cannot be neglected but may even be more important than the collisional effects. In order to address this situation, we have developed a model that combines sophisticated coupled electron/photon collisional transport with transport in externally applied macroscopic electric and magnetic fields of arbitrary spatial dependence.

The model allows magnetic fields in both material and void regions. Of course, magnetic fields alone will only deflect electrons without changing their energy. The procedure for combining collisional energy loss and deflection with magnetic deflection has been described elsewhere in detail.[15][35][36] Briefly stated, the rectilinear random-walk substeps of the field-free model[1][2] are replaced by numerically-integrated segments of field trajectories in vacua whose integrated areal densities are equal to those of the substeps. Sampled collisional deflections are superimposed upon the electron direction at the end of each of these vacuum-trajectory segments. The numerical integration scheme determines those locations along the segment that correspond to energy deposition and secondary production (knock-on electrons, bremsstrahlung photons and relaxation particles from electron impact ionization), as well as the intersections of the trajectory segments with material boundaries. Magnetic fields should be ignored in regions where transport is collision dominated, because the combined simulation is quite expensive in such cases, though the results are the same as for collisions alone.

Electric fields (or combined electric and magnetic fields) are allowed only in void regions. This restriction has been imposed because no sufficiently general scheme has been derived to account for the effects of changes in the electron energy produced by the macroscopic electric field upon the energy-dependent multiple-interaction collisional processes within a given substep. For those applications where electric fields are present within material media (e.g., potential buildup in dielectrics and sustaining fields in gas lasers), special algorithms[37] that depend upon the ratio of the electric potential gradient to the electronic stopping power must be introduced to handle this difficulty. Even with the restriction of electric fields to void regions, the model is applicable to a wide variety of problems – for example, problems involving accelerating diodes.

The method for accurately integrating the vacuum equations of motion in order to obtain the vacuum-trajectory segments is the essential feature of the model, whether the fields are present in material or in void regions. In voids the integration is interrupted whenever the trajectory intersects a material
boundary or a problem-escape boundary. In material regions the integration is also and more frequently interrupted whenever the areal density traversed corresponds to a location where energy is to be deposited or secondary production occurs, or equals the areal density of the appropriate substep. In the latter case, collisional scattering and energy loss are accounted for and a new trajectory segment is initiated. A fourth-to fifth-order Runge-Kutta-Fehlberg routine with automatic step-size control (RKF)[16], substantially modified to include boundary-crossing logic and other constraints, is employed to integrate the equations of motion in vacua. The reasons for this choice are discussed at length in Ref. [36].

The current algorithm includes a major improvement over the method described in Ref. [16]. The basic RKF integrator was designed to integrate over some specific interval of the independent variable. However, model applications invariably require interruption of the integration at the roots of any one of several possible constraint functions that are functions of the dependent variables. The most common example is the root corresponding to the intersection of an electron trajectory with a zone boundary. Other examples are the roots corresponding to the locations for energy deposition and secondary production. In the pre-ITS versions of the field codes we were forced to use relatively crude approximate solutions at these roots, which limited the overall accuracy of the model predictions to something substantially less than the inherent accuracy of the integrator.[16] The algorithm now includes an extended RKF procedure[17] that permits interruption of the integration at any one of a number of constraint functions of both the dependent and independent variables with an accuracy that is comparable to the inherent accuracy of the integrator. The more sophisticated user is free to add his own constraint functions for interrupting the integration. The constraint functions must be defined in subroutine CSTR.

19 Biasing Options and Variance Reduction

Biasing Options and Variance Reduction

From a practical, if not theoretically rigorous, point of view, biasing in Monte Carlo can generally be described as the distortion of the natural analog to achieve variance reduction in certain desired output quantities. Variance reduction refers to the attainment of lower statistical uncertainty for the same amount of run time or, equivalently, the attainment of the same statistical uncertainty in a lesser amount of run time. Except where absolutely necessary, biasing should be used sparingly. In any case, it should be used with great care. Reckless use of biasing (overbiasing or underbiasing) can lead to results that are erratic and/or easily subject to misinterpretation. Nevertheless, there are biasing options in ITS that are easily accessed via input keywords and that have proven very useful in specific applications.

Zone-dependent particle cutoff energies can be specified with the CUTOFFS keyword. The BIASING keyword may be used to simultaneously

Last Modified Date: 2008/04/17 22:45:28
specify global biasing parameters, the zones in which global biasing parameters are to be applied, and zone-dependent biasing parameters.

19.1 Zone-Dependent Cutoff Energies

The user has the option of varying the cutoff energy for each species from zone to zone so long as the zone-dependent cutoffs are greater than or equal to the global cutoff energy for the particle species. The option is activated via the appropriate parameter associated with the CUTOFFS keyword.

For electrons in the continuous-energy codes, when the energy of the electron in a given zone falls below the cutoff for that zone, a check is first made to see if it is trapped in the sense described under sub-keyword TRAP-ELECTRONS. If so, the history is terminated via on-the-spot deposition of charge and energy. Otherwise, except for when a magnetic or electric field is used or the NO-KICKING keyword is used, a final calculation of non-local energy and charge deposition is made based on the residual range of the electron. When a magnetic or electric field is used, a relatively low electron cutoff energy should be used because the history is always terminated via on-the-spot deposition of the charge and remaining energy of the electron. It is important to remember that there is no production of secondary particles by electrons below the zone-dependent cutoff, nor is there any contribution to electron flux and electron escape by such electrons.

For photons, and electrons in the multigroup code, when the energy of a particle falls below the cutoff for the zone the particle is in, the history is terminated via on-the-spot deposition of charge and energy. Because photons can travel relatively large distances before interacting and photon transport is generally inexpensive compared to electron transport, this cutoff energy should be used with caution.

The zone-dependent cutoff option has proven useful in problems that involve the generation of bremsstrahlung in one region and deposition caused by that bremsstrahlung in another region. A relatively high cutoff may be used in the converter zone(s) since low-energy electrons are relatively inefficient for producing bremsstrahlung. On the other hand, in the zone(s) where deposition is dominated by bremsstrahlung transport, the user may be interested in the details of the deposition from the low-energy bremsstrahlung-produced secondaries (e.g., interface effects, in which case he may not want to kill those electrons with Russian Roulette; see Sec. 19.3), or he may not want electron transport in those zones at all (bulk deposition; see Sec. 19.5).

19.2 Forced Photon Collisions

An option is available for forcing a selected fraction of photons entering a given zone or leaving a collision site within a zone to interact in that zone. The option is activated via the COLLISIONFORCING
sub-keyword. The option is useful for forcing photons to interact in certain regions where their natural interaction probability is so small as to make it difficult to obtain statistically significant results. The values chosen for the forcing fractions must be greater than zero and less than one. These are specified by the appropriate parameter for the given zone as described under the COLLISION-FORCING sub-keyword. Care must be taken not to overbias. A forcing fraction of 1.0 will prevent any photons from escaping from the given zone and will prevent them from making contributions elsewhere in the results of the calculation (e.g., prevent them from contributing to the escape fractions).

19.3 Russian Roulette

When SCALE-BREMS or SCALE-EP options are used to increase the secondary photon population, it may be desirable to reduce the number of secondary electrons generated from the interaction of this artificially high bremsstrahlung population. The ELECTRON-RR option allows the user to activate Russian Roulette in specified zones of the geometry. The default survival probability (when the feature has been activated) will return the secondary electron population to the naturally occurring number.

Although this procedure is very efficient for predicting external bremsstrahlung, it can lead to statistically poor results for the profiles of energy deposition, charge deposition, and electron flux in regions of the problem where these profiles are determined by the transport of and secondary electron production by the bremsstrahlung radiation. In such cases, the survival probability should be adjusted to ensure that a sufficiently large fraction of the secondary electrons are followed.

19.4 Next-Event Estimator for Photon Escape

For a geometry that is highly absorptive to secondary photons generated in the transport process, scoring as leakage photons only those secondary photons that actually escape the geometry while being tracked can be quite inefficient. (In the continuous-energy code, the contribution to total leakage from uncollided source photons is not scored because this contribution can usually be calculated analytically and might otherwise dominate the total leakage to the extent that the scattered contribution cannot be determined. However, a flag on the PHOTONS keyword can be used to change this default.) This is remedied by using the next-event estimator for photon leakage. With this estimator, a score is obtained each time a photon emerges from a collision. The score is simply the emergent photon weight times the probability of escape without further interaction.

This option is automatically activated as a method of variance reduction for differential leakage when the PHOTON-ESCAPE keyword is used. Otherwise, the option is not used for the default prediction of integral leakage unless explicitly activated via the sub-keyword NEXT-EVENTESCAPE.

19.5 Photon Only Transport
In some calculations, electron transport is only important in a small portion of the geometry. In this case, it is more efficient to simply turn off electron production throughout much of the geometry rather than using a high electron cutoff energy. The PHOTRAN sub-keyword provides this functionality. Zones in which electron transport is desired can be specified.

19.6 Scaling of Bremsstrahlung Production (*ITS Only*)

The user may artificially increase the bremsstrahlung production to improve the statistical accuracy of bremsstrahlung output without increasing the number of primary electron histories, which would be much more time consuming. The option is activated via the SCALE-BREMS sub-keyword. The cross sections are scaled by a factor specified by the user.

Simultaneous scaling of the cross section for electron impact ionization probability is also desirable (see Sec. 19.8) and is performed by default when SCALE-BREMS is used. The material selected as the basis for scaling the impact ionization (the second parameter associated with the SCALE-BREMS sub-keyword) should be that material which one would expect to dominate the bremsstrahlung production.

This option is used primarily for the prediction of external bremsstrahlung production (e.g., prediction of the environment of an x-ray source). Consequently, a Russian Roulette procedure may be desirable to reduce the number of secondary electrons generated. See Sec. 19.3 for details on the Russian Roulette feature.

19.7 Scaling of Electron-to-Photon Interactions (*MITS Only*)

Different types of interactions that have the same effect are not distinguished in the MITS codes. Therefore, it is not possible to simply scale the bremsstrahlung production of photons or the electron impact ionization production, as is done in the continuous-energy codes. Instead, the SCALE-EP feature allows the scaling of all photon-producing electron interaction cross sections. The cross sections of all materials are scaled by a factor specified by the user, but the scaled cross sections are only used in regions specified by the user.

19.8 Scaling the Probability for Electron Impact Ionization (*ITS Only*)

An option similar to that described in Sec. 19.6 permits the user to artificially increase characteristic x-ray production by scaling the cross section for electron impact ionization. This option is activated via the SCALE-IMPACT sub-keyword. The cross section for electron impact ionization of each material
is scaled so that an electron slowing down in that material from the maximum source energy to the global electron cutoff energy will, on the average, generate a number of ionization events equal to the value of the parameter associated with this sub-keyword. A separate scaling factor is calculated for each material. Impact ionization scale factors are rounded to the nearest integer and are never allowed to be less than one. Impact ionization scaling is implemented in the same zones in which bremsstrahlung scaling is implemented.

If SCALE-BREMS is used and SCALE-IMPACT is not, electron impact ionization is scaled. The factor for this scaling is determined such that impact ionization events between the maximum source energy and the global electron cutoff energy equal 20% of the bremsstrahlung events over the same energy range in the material given by the second parameter on the SCALE-BREMS keyword. This number of ionization events is used to determine scale factors for each material.

19.9 Scaling of Photon-to-Electron Interactions (*MITS Only*)

The user may artificially increase the production of secondary electrons to improve the statistical accuracy of dose or charge deposition in a region of the problem rather than increasing the number of histories, which may be more time consuming if relatively few photons naturally interact in the region. This option is activated via the SCALE-PE sub-keyword. The cross sections of all materials are scaled by a factor specified by the user, but the scaled cross sections are only used in regions specified by the user.

19.10 Trapped Electrons (*Forward Only*)

In certain problems where only electrons that cross certain boundaries are important, the option activated by the TRAP-ELECTRONS sub-keyword may be employed to reduce run time significantly. The parameter associated with this sub-keyword is the global electron trapping energy. In addition, zone-dependent electron trapping energies may be specified. Internally, an array of zone-dependent electron trapping energies is obtained, each element of which is the greater of the global trapping energy, the zone-dependent trapping energy, or the zone-dependent cutoff energy for that particular zone (see Sec. 19.1). The option becomes effective when an electron is trapped, that is, does not have enough energy to escape from a subzone. When an electron with energy less than the zone-dependent trapping energy is trapped, its history is immediately terminated via local (on-the-spot) deposition of its charge and remaining energy. This option is commonly used when one is primarily interested in the accurate transport of those electrons escaping from all or some portion of the problem geometry. Great care should be taken in
employing this option where production of secondaries (e.g., bremsstrahlung) may be important because there is no secondary production by electrons whose histories are terminated in this fashion. It is important to note that the contribution to leakage from the subzone of any untrapped electrons with energies above the zone-dependent cutoff is much more rigorous than that of untrapped electrons with energies below the zone-dependent cutoff because a much cruder form of transport is employed for the latter. The decision as whether an electron is trapped or untrapped is based on subzone boundaries in the TIGER codes, axial and radial subzone boundaries in the CYLTRAN codes, and subzone (in subzoned regions) or code-zone (regions separated by an OR operator in the input-zone definitions and input zones defined without the OR operator; see the ACCEPT Geometry section) boundaries in the ACCEPT codes.

20. Statistics 147

Last Modified Date: 2006/10/06 19:52:56

20 Statistics

A significant advantage of the ITS system is the computation of statistical uncertainties for virtually all output quantities. Under the default option, the total number of histories of primary particles are run in 20 equal batches. The output routine is called at the end of each batch. Immediately before each write statement, a call is made to Subroutine STATS. This routine (a) recalls the statistical variables from the previous batch corresponding to the output quantities about to be written, (b) computes the estimate of the statistical standard error (in percent) based on the number of batches that have been run, and (c) saves the statistical data from the current batch so that it will be available for the next batch. Unless the keyword PRINT-ALL is used, only the final results based on the total number of completed batches are printed out. The user may specify a number of batches other than 20 by using the keyword BATCHES as described in the Keywords for ITS section.

Under normal operation virtually every Monte Carlo output quantity is followed by a one- or two-digit integer from 0 through 99 (estimates even greater than 99 are shown as 99) that is the best estimate of the statistical standard error expressed as a percent of that output quantity:

\[
(S.E.)_N = \frac{100 \sqrt{\frac{\sum_{i=1}^{N} (xN_i - \bar{x}_N)^2}{N-1}}}{\bar{x}_N},
\]

where

\[
\bar{x}_N = \frac{N}{N-1} \cdot xN
\]
\[ \text{for } i = 1 \text{ to } N \]

\[ N_i = \sum_{i=1}^{N} \frac{X_i^2}{\mu_X^2}, \]

are the values of the quantity obtained from each batch, and \( N \) is the total number of completed batches. Should the more sophisticated user wish to add additional tallies to any of the Monte Carlo member codes, he will find subroutine STATS to be a useful utility. STATS provides estimated statistical uncertainties for all output quantities. It has three formal parameters. The first is a temporary array containing the current batch values for the quantities for which statistical estimates are desired when the routine is called. The routine returns the cumulative batch averages in this same array for printing. The second parameter is an array that will return the statistical estimates for printing. The third parameter is the number of batch values to be processed with this call (the length of the first two parameter arrays). An additional advantage to using the STATS utility is that results will be included in the arrays employed in the RESTART functionality.

21. Automatic Subzoning

Automatic subzoning refers to that feature of the ITS codes whereby the user may direct a particular member code to internally divide a given input zone into subzones for the purpose of obtaining the spatial variation of energy and charge deposition, electron flux, and photon flux within the given input zone. This powerful feature has the potential for substantially reducing (a) user time for generating the input file, (b) machine memory requirements, and (c) machine CPU time. Without this feature, the user would have to describe each subzone as a separate input zone. For example, a 10\times10\times10\text{subzoningofarectangular parallelepiped} with the ACCEPT codes would require at least 1000 lines of input without automatic subzoning.
Because the subzones are defined in terms of equal increments of the intrinsic coordinates of the input zone, there is no need for explicit storage of the boundaries of subzones, and very little memory is required to locate the subzone containing an arbitrary point within the input zone. Finally, because the internal subzone boundaries are not material discontinuities, they can be ignored by the CPU-intensive tracking logic.

Automatic subzoning and related coding has grown with the development of the ITS system, and is still not as complete as we would like. In the TIGER codes, it was already virtually complete in Version 1.0. In that version of the CYLTRAN codes, there was some capability for pseudo-subzoning a solid annulus input zone in the radial and axial directions. However, the only reduction was in the sense of Item (a) above. Once these “subzones” were generated internally, they were treated like any other input zone. Their boundary information was permanently stored, and particles were tracked through them just like any other input zone. In Version 1.0 there was no subzoning of any kind in the ACCEPT codes. In Version 2.0, we implemented full automatic subzoning in the sense of Items (a), (b), and (c) above in the ACCEPT codes for input zones consisting of a single RCC or RPP body. In the latter we allowed three-dimensional subzoning, and in the former we allowed radial and axial subzoning. No additional automatic subzoning was implemented in Version 2.0 of CYLTRAN. In Version 3.0, we substantially extended this feature. We implemented full automatic subzoning in the CYLTRAN codes, extending it to three dimensions with the addition of azimuthal subzoning. Moreover, the ACCEPT codes featured the full three-dimensional subzoning of input zones consisting of a single body of type RCC, RPP, BOX, or SPH. Since Version 3.0, automatic subzoning has been added for input zones of a single body of type TRC, WED, or TOR. Also, a new type of subzoning has been added based on multi-body input zones. These are generally shells that consist of one body subtracted from the middle of another body. These entities are the RCC-RCC, RCC-TRC, TRC-TRC, TRC-RCC, SPH-SPH, TORTOR, SPH-TOR, SPH-RCC, and RCC-TOR.

There are, however, some aspects of the coding of ITS that do not yet take full advantage of or are not completely consistent with the philosophy of automatic subzoning. We discuss them here in terms of three basic questions that are frequently asked as a particle trajectory evolves within the problem geometry:

1. Point location: In what input zone (or subzone) does a given point of the trajectory lie?
2. Tracking: For a particle at a given point in an input zone (or subzone) with a given direction, how far will it go before reaching the boundary of the given input zone (or subzone) if it continues moving in the given direction?
3. Trapping test (electrons only): For an electron at a given point in an input zone (or subzone), what is the minimum distance to the surface of that input zone (or subzone)?
In principle we would like the code to respond to such queries on a subzone rather than an input zone basis. In practice, it is felt that the overhead for doing the former may be so excessive as to negate the advantages of automatic subzoning. We have taken a middle of the road approach based on our judgment of what is the best overall choice and what is feasible at this time. Since Version 3.0, subzone-based electron trapping (see the section on Biasing) has been extended to the ACCEPT codes and to azimuthal subzones in the CYLTRAN codes.

In general, the features of the current ITS codes that use zones rather than subzones have to do with the actual scoring of certain spatially-dependent output quantities: energy deposition, electron flux, and photon flux. All of the scoring is subzone based, but there are alternative variations to what we currently use whose variance-reduction consequences have not been fully explored.

Energy deposition and scoring of volume-averaged electron flux are coupled. Both quantities are scored at the same point, as in Item (1) above, which is randomly selected along an electron random-walk substep, or partial substep if the substep encounters a material discontinuity. Because there is one score per substep, the use of automatic subzoning will not lead to variance reduction by increasing the number of scores per electron. Rather, the variance reduction from automatic subzoning is achieved because the time required to track an electron is reduced when subzone boundaries can be ignored.

There is, however, an alternative method of scoring these quantities, not yet implemented in ITS, that could potentially lead to further variance reduction in applications where the random walk substeps are greater than the subzone dimensions. We refer to this method as track-length apportioning. The segments of the substep within each subzone must first be determined. These segments times the electron weight are then scored as volume-averaged fluxes in the appropriate subzones. Similarly, the segments, as fractions of the total substep, are used to apportion the substep energy deposition among the subzones. Multiple scoring of these quantities per substep may lead to significant variance reduction. The caveat to this approach is that it requires what is equivalent to tracking, as in Item (2) above, among the subzones. It was the avoidance of tracking among subzone boundaries that was primarily responsible for the variance reduction achieved by automatic subzoning. Nevertheless, the more sophisticated user may use code modifications to implement this method. This is rather easily done for the TIGER codes (see discussion of photon flux below), but becomes progressively more difficult for the CYLTRAN and ACCEPT codes.

In the CYLTRAN and ACCEPT codes, scoring of volume-averaged photon flux is done in a fashion similar to that of electron flux. Here, however, instead
of obtaining one score per sub-step, there is one score per free-flight photon trajectory segment between collisions or input-zone boundaries. Again, scoring is at a point, as in Item (1) above, that is randomly sampled along the free-flight segment. Automatic subzoning does not increase the number of scores, and variance reductions again derive from the avoidance of tracking among the subzone boundaries. However, when the dimensions of the subzones are much smaller than the average free-flight photon trajectory segments, there could actually be a variance increase relative to the variance that would have been obtained had each subzone been defined explicitly as an input zone, since the number of scores is greatly reduced in the former case. Whether there is an overall variance increase or reduction is, of course, very problem dependent.

Because photon mean free paths tend to be much larger than electron random-walk substeps, and because one-dimensional tracking is relatively simple, it was decided to apportion the free-flight photon trajectory segments among the subzones in the TIGER codes when scoring photon flux. Thus, significant additional variance reduction, over and above that from automatic subzoning alone, may accrue to the estimation of photon fluxes with the TIGER codes, unless subzone boundaries tend to be much larger than photon mean free paths. Using code modifications, this

\[ 21. \]

**21.1 Non-Conformal Subzone Overlays**

At times, subzoning is desired for a zone that is quite complicated. For this purpose, a more general method is provided based on overlaying the zone with the subzoning of a simple entity. An input zone with arbitrary shape (that may include unions or CAD geometry zones) can be subzoned by first completely enclosing it in any simple body or one of the shells available for subzoning (RCC-RCC, etc.). The details of activating subzone overlays are described in the ACCEPT Geometry section.

Non-conformal subzoning affects the implementation of the electron trapping logic. With non-conformal subzones it is not sufficient to determine that an electron is trapped within a subzone; it is further necessary to determine whether the electron is trapped within the zone, if it is possible that the subzone lies on the zone boundary.

There are currently three methods for imposing subzone overlays. The first is to explicitly state the overlay to be used for subzoning the zone using the
OVERLAY sub-keyword. The second method, which applies only to CG zones, is to intersect the first code zone with the simple body (or bodies) in such a way that the first body (or two bodies, for a shell) in the description of the input zone specifies the entity to be used for subzoning. The additional intersection should result in a logically equivalent description of the input zone, since the original input zone description is required to be completely enclosed by that with which it is intersected (but this remains the user’s responsibility). The subzone structure is now based on the first body (or first two bodies, for a shell). In both of the first two methods, it is the user’s responsibility to choose a shape that makes sense; i.e., the subzone boundaries should describe a reasonable profile through the zone of interest. The third and most automated method, which applies only to CAD zones, is to base the subzoning on the RPP that is the axis-aligned bounding box of the zone.

The specified subzoning structure may describe some subzones that are completely outside of the original input zone. It is desirable not to include output for such subzones, and an attempt is made to exclude them from both the output file and the finite-element-format output written to the fort.3 file.

For CG, two checks can be made to determine whether to exclude a subzone from output. These checks are made if the “GEOMETRY 4” option is used. It is performed by comparing a point inside the subzone with the zone. (This point is chosen as the simple average of the eight vertices of the subzone. The vertices are subzone “corner” points that are used if finite-element formatted output is requested.) If the point is not within the zone, then the subzone is flagged for exclusion from the finite-element output. (For example, if the original zone is a quarter bend of a torus and the subzone structure is based on subzoning the full torus with the number of angular subzones around the circumference of the bend divisible by four, then only the subzones corresponding to the quarter bend of the torus will be included in the output.) Thus, if the full subzoning is not conformal, but all individual subzones are entirely inside or entirely outside of the zone, then this will be sufficient to accurately calculate subzone volumes.

The second CG check can provide a more rigorous determination of whether subzones coincide with the zone. For nonconformal subzoning to use accurate volume information, the volumes of the portion of subzones within the zone must be supplied. Volumes can be input via the “GEOMETRY1” or “GEOMETRY4” option. (These volumes can be calculated via a Monte Carlo calculation. Using a void geometry and the UNIFORM-ISOTROPIC-FLUX source, the flux of zones and subzones will be proportional to the volume.) The preliminary flag from the first check will be overwritten if the subzone has a non-zero volume. This covers the case where part of the...
subzone (containing the interior point) lies outside the input zone. It should not be possible to find the interior point inside the zone for a subzone with non-zero volume overlapping the zone. The code flags this condition as an error.

For CAD, one of two methods can be used for determining whether to exclude a subzone from output. For the “GEOMETRY 3” option, the volume is calculated for the intersection of each subzone with the zone. For the “GEOMETRY 1” or “GEOMETRY 4” option, the volumes of subzones can be input directly. (It may be useful to use the former option once and subsequently use the latter option to input the calculated volumes directly to avoid therepeated computational expensive of the volume intersections.) In either case, the output will be suppressed for subzones with zero volume.

22. Random Number Generators

Monte Carlo calculations rely on the use of pseudo-random number generators to simulate the stochastic nature of physical processes. Therefore, one must be careful to correctly implement the available random number generator (RNG). We provide the user a choice of 3 RNGs. In this document, we first discuss the properties of the RNGs provided and then share the issues of concern we have encountered in the implementation and use of RNGs.

22.1 Portable Random Number Generators

RNGs have been made available as compile options in ITSTore to relieve the user of concerns about the integrity and implementation of the intrinsic RNG on his system. All 3 RNGs are double precision and as implemented, never generate zeros or ones.

RNG1, an implementation of an RNG in the MCNP code [38], has a cycle length of $2^{46}$ and allows easy access to its seed. However, RNG1 is not implemented as a parallel RNG. RNG2 and RNG3 will function in both serial and parallel. To permit a common interface for the RNGs, RNG1 is used to seed each batch performed with RNG2 or RNG3. Regardless of which RNG is selected in the code, the user will be seeding and receiving seed information from RNG1. Because the states of RNG2 and RNG3 are quite large, they will be written to a file when required for debugging. Only the RESTART-HISTORY feature requires access to these states.

RNG2 is an implementation of the RANMAR RNG [39, 40]. It has a state space of $2^{45}$, consisting of numerous cycles with an average cycle length of...
approximately \(2^{100}\). Each batch of RNG2 is seeded with a combination of 97 random numbers from RNG1.

RNG3 is an implementation of the Mersenne Twister RNG [42]. It has a cycle length of \(2^{19937} - 1\). Each batch in a run with RNG3 is seeded with a combination of 624 random numbers from RNG1.

### 22.2 Range

The random numbers that are generated should be uniformly distributed between 0.0 and 1.0 exclusively. There are certain places in the Monte Carlo software where random numbers that are identically 0.0 or 1.0 are unacceptable, resulting in fatal execution errors.

### 22.3 Access to the Seed

The state of an RNG consists of those variables that are used to determine the next random number in the sequence. The state of RNG1 consists of a single variable. Access to this variable is required to assure independent random number sequences in separate runs, to have a restart capability, and to facilitate debugging. If an error condition is detected while the Monte Carlo is in progress, for RNG1 the starting random-number seed for the current source particle (variable RIR-SAV in randat.h) is printed within the terminating subroutine, ABORTX. For RNG2 and RNG3, the starting random-number seed for the current batch and the number of random-numbers generated in the batch before the current history (variables RIRA and CNRN(2) in randat.h) are printed, and the initial RNG state for the current source particle is printed to a file.

One symptom of incorrect implementation of an RNG may be obtaining “0” percent statistical uncertainties for all quantities; this may happen if the same seed is being used to start each batch so that the results from each batch are identical.

### 22.4 Reproducibility

ARNG is designed to produce the same sequence of random numbers for the same starting seed.

### 22.5 Cycle Length

Computer generated random numbers have a finite cycle length (the number of random numbers generated before the cycle repeats itself). One should never run so many source particles in a given run so as to exceed the RNG cycle length. A single source particle will likely use many random numbers. The number of random numbers used in a calculation is presented in the output.
22.6 Speed

Timings indicate that for our implementations RNG2 is generally faster than RNG3 and that both RNG2 and RNG3 are faster than RNG1. However, these timings vary depending on the platform and compiler.

23 Adjoint Calculations

The adjoint calculation mode is a complement to the forward calculation mode. In some situations where forward calculations are inefficient, adjoint calculations offer an efficient alternative. In general, if one is interested in a variety of responses with distributions in space, angle, and energy (e.g., dose and charge deposition distributions due to a limited number of radiation sources (e.g., a monoenergetic electron beam), then it will be more efficient to use the forward method. However, if one is interested in a limited number of responses (e.g., dose at a point) due to a large number of radiation sources with distributions in space, angle, and energy (e.g., sources from various directions), then it may be more efficient to use the adjoint method.

From linear algebra, given an inner product \((,\), an operator \(T\), and two functions \(S\) and \(R\), the following equality holds:

\[(T S, R) = (S, T^\dagger R),\]

where \(T^\dagger\) is the adjoint operator of \(T\). For our purposes in the MITS code, we can think of the \(T\) as the forward mode of the computer program, and we can think of \(T^\dagger\) as the adjoint mode of the computer program. Running MITS in both forward and adjoint modes to calculate a single response \(R\) due to a single source \(S\) should yield identical results (within statistics) since the code is solving the same problem using the same cross sections.

The forward computer program \(T\) “operates” on a single forward source \(S\), which must be described in space, energy, and angle. For Monte Carlo, we can think of the program \(T\) as tracking particles, and the particle source is specified by \(S\). The result of \(TS\) is the particle flux. If the user wants a value of dose, for example, that is the value of the inner product, then the flux must be folded with the appropriate response \(R\) to obtain that value. For dose, the
response is the restricted stopping power for the region of interest. For a single source, the particle flux $T S$ can be folded with many different responses to obtain, for example, doses in various materials throughout the geometry, dose profiles within a single material, charge deposition profiles, escaping particle distributions as a function of energy and/or angle, etc. As a convenience to the user, our codes will compute the inner product for a library of common responses.

Similarly, the adjoint computer program $T \dagger$ operates on a single response $R$, so the user must specify what single type of response (e.g., dose at a point) is desired. For Monte Carlo, we can think of the program $T \dagger$ as tracking “particles” called adjunctons, and the “adjuncton source” corresponds to the response $R$. The result of $T \dagger R$ is a particle importance map, that gives the importance of particles (as a function of species type, space, energy, and angle) to the specified response. The inner product can now be evaluated for many different types of forwardsources $S$. As a convenience to the user, our codes will compute the inner product for a library of common sources.

In forward mode, the distribution (in space, energy, and angle) and the species type of a single source must be specified using the POSITION, ENERGY or SPECTRUM, DIRECTION, and ELECTRON or PHOTON keywords. Then, a variety of responses may be selected. Energy and charge deposition are calculated throughout the problem by default. The user may request escape distributions (using the ELECTRON-ESCAPE, PHOTON-ESCAPE, and ESCAPE-SURFACES keywords), electron surface emission distributions (using the ELECTRON-EMISSION keyword), and flux distributions (using the ELECTRON-FLUX and PHOTON-FLUX keywords).

In adjoint mode, a single response must be specified with the DETECTOR-RESPONSE keyword. The user must specify the type of response as DOSE, CHARGE, KERMA, or ESCAPE and must describe some properties (such as spatial extent) of the detector. This specifies the space, energy, and angle distribution of an adjuncton source. Then, a variety of sources may be selected. The user must request at least one source, since there are no defaults. The user may specify a surface source of each species (ELECTRON-SURFACE-SOURCE and PHOTON-SURFACE-SOURCE) and a volume source of each species (ELECTRON-VOLUME-SOURCE and PHOTON-VOLUMESOURCE).

In addition to the choice of spatial distributions of sources in adjoint mode,
the user has the option of “binning” in energy and angle. This binning represents dividing the energy and angle domains into separate, independent sources of energy and angle extents given by the bin. Thus, if one divides a surface source into 9 equal polar-angle bins, then the output will contain the requested response for a source with a distribution from 0 to 10 degrees, a source from 10 to 20 degrees, etc. Such data may be post-processed assigning weighted distributions in angle and energy to determine the response to a wide variety of combinations of energy and angular source distributions. On the other hand, it may be more accurate (due to the approximation of binning the sources before folding with the actual distributions) and the output will be greatly condensed if the user is able to specify an angular distribution of interest (such as COSINE-LAW), the energy spectra of interest (using the SPECTRUM keyword), and allow the code to fold the energy and angular distributions. The choice between folding with source distributions during the calculation or in post-processing is left to the user.

24 Testing of ITS

This section discusses the tests available for code maintenance. This includes installation tests, commit tests, and regression tests. The first subsection discusses how the test script can be used to run these tests. The second subsection discusses how to perform and evaluate installation testing.

24.1 Testing Scripts

The “tests.pl” script in the Tests/RegTests directory is used to execute the installation, commit, and regression tests. The numerous flags that may be used with this Perl script are listed in Table 10 along with
a brief description of each flag’s effect. More detailed descriptions of how these flags may be used are discussed here.

The suite of tests to be drawn from may be selected using the “-commit”, “-install”, and/or “-regress” flags. All of the commit and regression tests are selected by default, if none of these flags is specified. These flags may be used in combination. They specify which tests are loaded, but the user may limit which tests are executed using additional flags.

For users with a released version of the code that does not include RNG3, the tests should be executed using the “-rng1” and “-rng2” flags. Users who do not have a CAD capability with ITS should use the “-cg” flag. Users who do not have the PFF libraries should use the “-notpff” flag. Users who do not have MPI libraries should use the “-serial” flag. Except for RNG1 tests and a few tests of features not available in parallel, the same tests are executed in parallel as in serial.

On some platforms it may be necessary to compile the tests (perhaps on a front-end platform) and execute them separately. Separating the compile and execution steps can also be useful for code debugging. For this reason the “-pre” and “-post” flags are available. The “-pre” flag will cause the script to compile the executables required and assemble the tests in subdirectories of the $HOME/tmp directory. After these tests have been run, the “-post” flag will cause the script to post process the output files. The runcalcs.pl script, discussed below, may be useful for executing the tests.

Many other flags are intended to be helpful while debugging specific executables. These flags only serve to limit the tests executed. They do not add tests. Two complementary flags are the “-its” flag, which specifies only the ITS tests, and the “-mits” flag, which specifies only the MITS tests. Another pair of complementary flags are “-cad” and “-cg”. The flags “-acc”, “-cyl”, and “-tig” specify the ACCEPT, CYLTRAN, and TIGER codes, respectively, all of which are included by the “-cg” flag. The flags “-pcodes” and “-mcodes” specify the PCODES and EBFIELDS tests, respectively. The “-std” flag specifies the standard code tests (meaning non-PCODES and non-EBFIELDS tests). The flags “-static” and “-dynamic” allow the user to determine the MPI mode to be used for all tests. By default, the MPI mode is set for each test as part of the test suite.

The flags for limiting the number of tests to be executed not only state which tests will be run but also imply that other tests will not be run. The “-its” flag states that ITS tests will be run, and implies that MITS tests will not be run. The “-cad” flag implies that CG tests will not be run, but this may be overruled by explicit use of another flag. Using the “-cad” and “-cg” flags will result in all tests being run. A more useful combination would be the “-cad” and “-acc” flags together, to include both subsets of tests. The “-pcodes”, “-mcodes”, and “-std” may also be
Table 10. Flags available with the “tests.pl” ITS testing script

<table>
<thead>
<tr>
<th>Flag</th>
<th>Effect of the Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>help</td>
<td>Displays the list of valid flags and their functions</td>
</tr>
<tr>
<td>install</td>
<td>Tests are drawn from the installation suite</td>
</tr>
<tr>
<td>commit</td>
<td>Tests are drawn from the commit suite</td>
</tr>
<tr>
<td>regress</td>
<td>Tests are drawn from the regression suite</td>
</tr>
<tr>
<td>its</td>
<td>The ITS tests will be run</td>
</tr>
<tr>
<td>mits</td>
<td>The MITS tests will be run</td>
</tr>
<tr>
<td>cad</td>
<td>The CAD tests will be run</td>
</tr>
<tr>
<td>cg</td>
<td>The CG tests will be run</td>
</tr>
<tr>
<td>acc</td>
<td>The ACCEPT tests will be run</td>
</tr>
<tr>
<td>cyl</td>
<td>The CYLTRAN tests will be run</td>
</tr>
<tr>
<td>tig</td>
<td>The TIGER tests will be run</td>
</tr>
<tr>
<td>pcodes</td>
<td>The PCODES tests will be run</td>
</tr>
<tr>
<td>mcodes</td>
<td>The EBFIELD tests will be run</td>
</tr>
<tr>
<td>std</td>
<td>The standard code tests will be run</td>
</tr>
<tr>
<td>pff</td>
<td>Only PFF tests will be run</td>
</tr>
<tr>
<td>notpff</td>
<td>No PFF tests will be run</td>
</tr>
<tr>
<td>rng1</td>
<td>The RNG1 tests will be run</td>
</tr>
<tr>
<td>rng2</td>
<td>The RNG2 tests will be run</td>
</tr>
<tr>
<td>rng3</td>
<td>The RNG3 tests will be run</td>
</tr>
<tr>
<td>serial</td>
<td>The tests will be run in serial</td>
</tr>
<tr>
<td>mpi</td>
<td>The tests will be run in parallel</td>
</tr>
<tr>
<td>static</td>
<td>The MPI mode will be static for all tests</td>
</tr>
<tr>
<td>dynamic</td>
<td>The MPI mode will be dynamic for all tests</td>
</tr>
<tr>
<td>mpicmd</td>
<td>Sets the command used to launch parallel calculations</td>
</tr>
<tr>
<td>sercmd</td>
<td>Sets the command used to launch serial calculations</td>
</tr>
<tr>
<td>platform</td>
<td>Set mpicmd and sercmd for a specific platform</td>
</tr>
<tr>
<td>compile</td>
<td>The user can specify the compile flags</td>
</tr>
<tr>
<td>pre</td>
<td>Tests will be created in $HOME/tmp, but not run</td>
</tr>
<tr>
<td>post</td>
<td>Tests that have been run will be post processed</td>
</tr>
<tr>
<td>check</td>
<td>Serial and parallel archive outputs will be compared</td>
</tr>
<tr>
<td>checknew</td>
<td>New outputs will be compared with archived outputs</td>
</tr>
</tbody>
</table>
The names of active tests will be listed

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>list</td>
<td>The names of active tests will be listed</td>
</tr>
<tr>
<td>t=testcase</td>
<td>Runs a specific test case only</td>
</tr>
<tr>
<td>cover</td>
<td>Generate code coverage instrumentation and statistics</td>
</tr>
<tr>
<td>numdiff</td>
<td>Uses numerical differencing with tolerances</td>
</tr>
<tr>
<td>rel</td>
<td>Specifies the relative difference cutoff</td>
</tr>
<tr>
<td>abs</td>
<td>Specifies the absolute difference cutoff</td>
</tr>
</tbody>
</table>

used in any combination to invoke independent subsets of tests. The flag “-t=testcase” will run a specific test case (e.g., “tests.pl t=itsaccm1 3”).

The “-mpicmd”, “-sercmd”, and “-platform” flags may be set equal to some text string. The “-mpicmd” flag allows the user to specify the command to be used for launching parallel calculations (examples: “-mpicmd='mpirun -machinefile $HOME/machines -np 5' ” will launch cad calculations as “mpirun -machinefile $HOME/machines -np5its.x prmfie”, and “-mpicmd='yod -sz5’”willlaunchcg calculationsas’yod-sz5its.xmdatoutput”). The text string that one sets mpicmd equal to should be the string that a user would enter on the command line for launching the calculation including the specification of the number of processors (all MPI test benchmarks use 5 processors) and excluding the ITS execution command itself. The same is true of the “sercmd”flag for serial calculations. This is less likely to be required, but may be needed to request a single processor on a parallel platform (example: “-sercmd='yod -sz 1’” will launch cad calculations as’yod-sz1its.x prmfie”). The “-platform” flag may be used to select from a set of platforms for which the mpicmd and sercmd flags are known. Available platform options are
crater, pegasus, redstorm, white, and q. If the mpicmd or sercmd flag is used, the platform flag will be ignored.

The “-compile” flag allows the user to specify the compiler flag to be used with the tests. By default, the tests will be run with the O3 flag. The compiler flag should be specified by setting the compile flag equal to the desired compiler flag (e.g., “-compile=-g” or “-compile=-O2”). The only flags currently available are -g, -w, -O1, -O2, -O3, and -O4. Only a single flag may be specified.

The “-check” flag compares the serial output files in the Output directory with the parallel output files in the OutputMPI directory. The “-list” flag will simply list the tests that have been set to be executed. This can be useful for evaluating which tests are specified by a given set of flags. (The “-post” flag can be used to eliminate listing of “cleanup” tests, which merely indicate that a new executable must be compiled.)

The “-checknew” flag compares existing new output files in the NewOutput or NewOutputMPI directory with the archive output files. This flag can be useful to quickly evaluate whether differences observed when the new outputs were generated were due to minor numerical differences. We recommend that tests first be run with a simple diff test. If there are tests that fail the simple diff, the “-checknew” flag can be used with the “-numdiff”, “-rel”, and “-abs” flags to determine what level of numerical difference exists in the test failures. Small numerical differences are not unusual in outputs generated on different platforms and may be acceptable. When using the “-checknew” flag, the group of tests selected should be limited to those that have been newly generated, but a subset of those tests can also be selected. Individual tests can be examined with the “-t” flag specification.

As briefly mentioned earlier, the “runcalcs.pl” script can be used to execute tests that have been prepared using the “-pre” flag. This script accepts only a few flags. A flag must be set specifying an input file, such as “-file=example”. This input file should list the tests to be executed, by listing the names of the directories in which the tests have been set up. This can be accomplished by simply listing the contents of the $HOME/tmp directory, and redirecting the list into a file, such as “ls $HOME/tmp > example”. The user may need to edit this file to remove any extraneous information.

The runcalcs.pl script will run serial tests by default, but the “-mpi” flag may be used to specify that parallel tests must be run. The runcalcs.pl script will accept flags for mpicmd and sercmd, with the same behavior as described for the tests.pl script.

By default, the “diff”-erence in comparing the expected output to the actual output is strictly character comparison. The “-numdiff” flag causes the diff to ignore absolute and relative numerical differences of less than specified tolerances. This avoids identifying acceptable
variances in output due to numerical differences in compiler and/or processor floating point calculations. The numerical differencing tool is included in the ITS distribution in the `Tools/diffUtils` directory. To build the tool, run "configure" and "make". Copy `diffUtils/src/diff` to your `$HOME/bin` directory. The "-abs" flag allows the user to specify the cutoff for absolute differences, below which the differences will be suppressed. The default absolute tolerance is $10^{-18}$. The "-rel" flag allows the user to specify the cutoff for numerical differences, below which differences will be suppressed. The default relative tolerance is $10^{-8}$.

The "-cover" flag generates code instrumentation in the executable, and instrumentation output when the code is executed. These are then collated using Intel code coverage utilities. Typically, this would be run for all test cases at once: "tests.pl -cover". This generates complete code coverage statistics for all test cases. This is normally only performed by the developers. This option may only be used with the Intel compilers ifort and icc with versions 9.1.032 or above.

### 24.2 Installation Testing

See the Installation section of the manual for a broader set of instructions on code installation. Before performing installation testing of ITS, it is recommended that the XGEN and CEPXS cross section generating codes be tested. Testing of those codes is described in the sections Running XGEN and Running CEPXS.

It is recommended that the user compile an enhanced version of the "diff" utility. This can be created by changing to the `Tools/diffUtils` directory under ITS, and running: "configure" and "make". Once this is completed
successfully, copy diffUtils/src/diff to your $HOME/bin and ensure that the $HOME/bin directory is early in your search $PATH. This causes any numbers encountered with an absolute value less than 1e-18, or relative differences less than 1e-8 to be ignored. That is, extremely small numbers are ignored, and only 8 significant digits are examined.

Most users are unlikely to be developing code modifications, so in this section we describe the steps required to test the installation of the code on a new platform. The tests should be executed using the base command “tests.pl -install”, but other flags are likely to be necessary. The specific flags used will depend upon the capabilities of the code being tested. Outside of Sandia, users will not have access to RNG3. Users without RNG3 should use the flags “-rng1 -rng2”. Only users who have purchased and installed the ACIS libraries will have the CAD capability. Users without the CAD capability should use the “-cg” flag. Users who do not have access to MPI libraries cannot test the parallel capability in ITS and should use the “-serial” flag. For MPI, the user will likely want to use the “-mpicmd” flag, or the “-pre”, runcalcs.pl, and “-post” capabilities described in the preceding section.

As the test script executes, a report will be written to the screen stating the number of lines in the differences from the expected outputs. If all tests report zero lines of difference, the tests have passed. If the script terminates with an error message, the tests have failed. If differences are reported, the user must inspect the differences (in files in the Diffs directory) to determine whether these differences constitute a code failure. Often when porting the code to a new platform or new compiler, insignificant differences will be reported. Insignificant differences may be large relative differences in very small values (-1E-19 instead of 2E-20) or small relative differences (1.218E+01 instead of 1.219E+01). Any difference in the number of random numbers used constitutes a test failure. Use the enhanced diff capability mentioned above if too many rounding or small difference errors occur. This is invoked with the “-numdiff” flag.

If the test script exits with an error, the user should determine if an ohoh file is present in the RegTests directory, which may contain information about the cause of the failure. If there is no ohoh file, the user should inspect the test subdirectory of the $HOME/tmp directory. This directory may contain relevant information or allow the user to attempt to execute the test.

After the test script exits with an error, the user should execute “interrupt-cleanup” before attempting to run additional tests. When test failures are encountered, the user may wish to
A Running XGEN

This document contains outlines for running the XGEN code. Instructions are provided for running the code by using commands or by using scripts. Known platform dependencies of the code are discussed in the last section.

A.1 Running XGEN without Scripts

CHECKOUT: Do a "cvs checkout -P xgen" to acquire a copy of the code. If necessary, tar the directory and transfer it to the desired platform.

To run on any platform, either use the sendn script (described in the Running XGEN with Scripts section) or build an executable using the following commands.

MAKEFILE SETTINGS: In the directory xgen/Code, you must alter the Defines.mk settings to specify the PCODES definition or at least remove the OPTION1 placeholder for your build of the XGEN executable.

CONFIGURE: Execute ".configure". If the platform and operating system are identified, but the proper config/mh-* and config/mt-* files are not present, then you must either identify the proper files to use or create the necessary files and alter the configure.in file accordingly. If the platform and operating
system are not identified, the config.sub file also must be altered (search for tflips as an example).

MAKE: If configure functions properly, a working Makefile will be produced. Execute this with the “make” command to produce the XGEN executable, xgen.x.

ATOMIC DATA: The code xgen.x will look for the atomic data file in the local directory in the Fortran unit9 file. This file is in the xgen directory under xgen/Code/XSdata. Dopplerbroadening data is located in the comp.dat file in the same directory. The LLNL data set, epdl97.all, must be acquired and placed in the xgen/Code/XSdata directory. Soft links may be created from the location of the files to one’s working directory as:

```
ln -s $HOME/xgen/Code/XSdata/x6.dat fort.9 ln -s $HOME/xgen/Code/XSdata/comp.dat comp.dat ln -s $HOME/xgen/Code/XSdata/epdl97.all epdl97.all
```

EXECUTION: The input and output files may be specified as command-line arguments. The first argument is the input file. The second argument is the output file (hence, there is no way to specify the output file without specifying the input file). If an input file is not specified, the file “xgen.inp” will be used. If this file does not exist, it will be created as an empty input deck, and the code will be run with default settings. If an output file is not specified, the file “xgen.out” will be used.

REGRESSION TESTS: Files for running regression tests are in xgen/Tests/RegTests. The tests can be run by executing “tests.pl”. The script will place cross section files in $HOME/cross3, output and cross section files in the directory NewOutput, and results of diff with the repository version of output and cross section files in the directory Diffs. If the LLNL data sets have not been acquired, the “-noLLNL” flag should be used.

Running XGEN

The default is to compare the expected output to the actual output using strictly character comparison. The “-numdiff” flag causes the “diff”-erence to
ignore absolute and relative numerical differences of less than “-abs=1e-18” and “-rel=1e-7”. This avoids identifying acceptable variances in output due to numerical differences in compiler and/or processor floating point calculations. The numerical differenting tool is included in the Tools/diffUtils directory of the ITS distribution. To build the tool, run “configure” and “make”. The resulting executable should be copied from its/Tools/diffUtils/src/diffto your $HOME/bin directory.

A.2 Running XGEN with Scripts

ACVS checkout maybe performed on platforms that have direct access to the CVS repository. On all other platforms, the directory of files must be tarred and transferred, and the directory options in the sendn script must always be used instead of the CVS options.

For working on platforms that do not have direct access to the repository, it is recommended that two copies of the repository checkout be set up: one copy to be used as a working directory in which code modifications and builds can be performed, and one unaltered copy that can be compared with the maintain a record of changes made to the working version. The unaltered copy should be given a name uniquely indicating the cvs version.

1. SCRIPTS: Use the command “cvs checkout -P xgen” to acquire a copy of the code and all associated files.

   (a) From the directory xgen/Scripts, copy sendn and the nxCVS script to your $HOME/bin directory. Copy the files in the Subscripts directory to a $HOME/bin/Subscripts directory. sendn and the Subscripts are identical for ITS, XGEN, and CEPXS.

2. CUI FILE: Copy the xgen/Scripts/x.cui file to your working directory, and edit it for your specific problem. The sections of a cui file are:

   (a) DRIVER SCRIPT: The script nxCVS is available as a driver script. You may substitute a customized script by including it in the first section of the cui file. Sendn will position the cui file, including the sections of the driver script. The driver script contains the commands necessary to build and execute the program, clean up after itself, and produce relevant result information in a job file. (If necessary, it will also include in the job file information useful in determining the cause of a job failure).

   (b) DEFS: The PCODES preprocessor definition may be selected in the defs section of the cui file. There are several options for running the scripts that may be set in the defs section of the cui file. Compiler flags may be specified.

   The user may also request an interactive script. The interactive script should be used on systems that require job queuing or on systems that require cross compiling. The first
the script will build the executable. The files for running the job will be located in $HOME/tmp/<jobname>. The executable is named “xgen.x”, the input file is “xdat”, and the output file should be named “output”. When the calculation is complete, the user may execute the “postproc” file in the $HOME/tmp/<jobname> directory. This will move the fort.11 section file to $HOME/cross3 directory, rename it according to the response that was given to the sendn script, and produce a job file in the directory from which the job was originally submitted.

(c) DIFFS: Patches to be applied to the code should be inserted in the “diffs” portion of the xgen.cui file. These patches can be formatted as a cvs diff or as a directory diff. Diffs can be lifted out of job files from previous calculations. Alternatively, one can checkout a copy of the code, make modifications, and then generate a diffs file. On a platform with access to the CVS repository, one can use “cvs diff > diffs” from within the xgen/Code directory. On a platform without access to the CVS repository one can perform a directory diff, but this must be performed from within the xgen/Codedirectory of the unaltered copy of the code, and it must use the -r and -b flags (in that order), such as “diff -r -b $HOME/xgenaltered/Code> diffs”. Then, the diffs file can be used in the xgen.cui file. Multiple diff files (diffs from two different executions of the diff command) cannot be patched to a single cvs file.
New files can be added to a build. Within the diffs section, a line of the following format denotes the start of a new file:
New file: <Directory/Filename> To be included in the compiling and linking of the code, then the new file must be referenced in the Makefile.in list of sources. This change in the Makefile.in can also be included in the diffs section by making the desired change in a copy of Makefile.in and using the cvs diff procedure described above.

(d) XDAT: The XGEN input should be included in the last section of the
cui file. Instructions for XGEN input are in Keywords for XGEN.

3. SENDN: Submit the job using the command “sendn x.cui <jobname>”. This script will prompt you for the following pieces of information (and possibly the 4th depending upon your responses to the first 3):

(a) CROSS SECTIONS: First, sendn will request the name of the cross section file <cross> to be created. Upon successful completion, the scripts will assign the given name to the fort.11 cross section file and place it at “$HOME/cross3/<cross>”.

(b) LOCAL COMPILIE: Second, sendn will request the location of a checked-out copy of xgen that can be used for making the executable. Thus, you may use a version of the code that you have checked out and modified. You must give a complete pathname for the base xgen directory (e.g., /scratch/temporary/xgen). No files will be deleted from the make directory as a result of the calculation, but some files may be modified if requested in the diffs section of the cui file. Any new files that have been included in the directory (other than through the diffs section of the cui file) will not appear in the job file, but they may be used if the Makefile.in has been so modified (in which case, the job file will show the reference to the new file as a difference in the Makefile.in).

Be aware that any definitions specified in the “defs” section of the cui file and any modifications specified in the “diffs” section of the cui file will be applied to the files in this directory. Attempts to change the code definitions for a calculation using the same directory will not take effect unless “make clean” has been executed. Attempts to apply the diffs in a directory where the diffs have already been applied for a previous calculation will result in an error.

CVS COMPILIE: To request a cvs checkout of the code, you may respond to this request with “none” (or press “Enter”). The cvs checkout will be performed in the $HOME/tmp/<jobname> directory and should not affect any other versions of the code.

(c) DIRECTORYDIFF for LOCAL COMPILIE: If your response to the
second request was a directory pathname, the script will request a local directory with which to perform a diff. The diff command will be used to compare the two directories and all subdirectories. The job file will not record the version number of either directory, therefore the user may not have enough information in the job file to duplicate a calculation unless the diff directory has a name corresponding to the tag used in the cvs checkout of the code. The directory name does appear in the job file.

VERSION for CVS COMPILe: If your response to the second request was “none”, the script will request the version for a cvs checkout. The command will be issued as “cvs checkout <options> xgen”. Theresponse to this request will be used for the <options> and any valid cvs flags may be used that do not contain a slash,”/”. Examples of valid syntax for responses are: -D now, -D “March 28, 2001”, -D 4:00pm, -D “3 hours ago”, -D “2 fortnums ago”, -r 1, -r release-1.0, etc. Another valid response is to simply press “Enter” with no input, which will result in the checkout of the most recent version of the code.

(d) CVSDIFF for LOCAL COMPILe: If you gave a pathname for making the executable but not for a directory diff, the script will request the version for a cvs diff. The syntax for responses to this request is the same as for specifying the cvs version for a checkout. The directory in which the executable is made will be compared with the repository using the cvs diff command. This version (that will appear in the job file) and the results of the cvs diff (that will also appear in the job file) can be used to reproduce a calculation.

CVSUPDATE/DIFF for CVS COMPILE: If you did not give a pathname for making the executable, the script will request a version for a cvs update and diff. In this case, the script will checkout a version of the code using the options in the third response, apply the “diffs” from the cui file, attempt to update to the version of the code specified in this response, and then compare the resulting code to the repository using the cvs diff command with the options specified in this response. The version requested here (that will appear in the job file) and the result of the cvs diff (that will also appear in the job file) can be used to reproduce a calculation.

For either of these options, if no option is specified for the cvs diff, the option “-D now” will be used. The date and time of the calculation, which are recorded in the job file, and the result of the cvs diff can be used to reproduce the calculation at a later date.

A.3 Platform Dependencies

The following is a list of platforms on which XGEN has been successfully built and tested. Following the name of the platform is the system type and
XGEN Code Options

Only one code option can be selected for compiling the XGEN codes. This option has been implemented as a preprocessor definition. It is necessary to choose between the standard cross section code and the PCODES version.

The code option must be specified in either the CUI file (if using the scripts to execute the code) or in the Defines.mk file (if building an executable). Refer to the documentation on Running XGEN for more details on applying definitions.

B.1 Preprocessor Definitions

PCODES (more ionization and relaxation)

C.
Summary
of XGEN
Keywords

C Summary of XGEN Keywords

Table C.1 contains a listing of keywords relevant to the XGEN code. The keywords are listed approximately in an order of importance. In addition, for each keyword the default code behavior is listed. The default behavior will be employed by the code if the keyword is not found in the input deck. More detailed descriptions of the syntax, sub-keywords, and use of these keywords are contained in the XGEN Keywords section.

<table>
<thead>
<tr>
<th>KEYWORD</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>**** MATERIALS ****</td>
<td></td>
</tr>
<tr>
<td>MATERIAL</td>
<td>AL</td>
</tr>
<tr>
<td>CONDUCTOR or NON-CONDUCTOR</td>
<td>see discussion under keywords</td>
</tr>
</tbody>
</table>
### Keywords for XGEN

The input keywords must be specified in either the CUI file (if using the scripts to execute the code) or in the input file (if executing the code manually). Refer to the documentation on Running XGEN for more details on specifying the input file in the CUI file or otherwise. An overview of the keywords is available in the Summary of XGEN Keywords section.

#### D.1 Input Notation

Most keywords should be used once and not repeated in an input file. The exception to this is the MATERIAL keyword. Sub-keywords should be used once per use of their primary keyword.

Parameters are associated with the preceding keyword appearing on the same line. If parameters are omitted, they will be set to zero. Consideration should be given to the fact that in some situations this value is invalid and will trigger an error.

Comments may be inserted in the input deck. Anything
appearing to the right of an asterisk anywhere in the input deck will be treated as a comment and ignored by the code.

Input is not case sensitive, with one insignificant exception. Character input provided with the TITLE keyword will appear in the output file exactly as provided, but the case will not affect the code in any way.

D.2 Keywords

1. BELOW-1KEV

Syntax: BELOW-1KEV
Default: Off

If “BELOW-1KEV” is inserted in the input stream, energy grids for less than 1 keV will be generated from the LLNL Evaluated Photon Data Library (EPDL97).

The new capability for attenuation of source photons below 1 keV requires the Livermore evaluated data library for photons, which is not distributed with ITS. It is available for download from the IAEA website at http://www-nds.iaea.org/epdl97/. ITS uses the EPDL97 in ENDF/B-VI format. This file should be located in the XGEN distribution at .../Code/XSdata/epdl97.all.

2. DOPPLER

Syntax: DOPPLER
Default: no Doppler

If “DOPPLER” is inserted in the input stream, Compton profiles, elemental binding energies, and shell configurations will also be generated. Only single element materials are allowed with this option.

3. ECHO

Syntax: ECHO0

D.3 Keywords
Default: Input is echoed to the output

If “ECHO 0” is inserted in the input stream, subsequent input will not be echoed. If “ECHO 1” is inserted in the input stream, subsequent input will be echoed to the output.

4. ELECTRON-ANGLE-BINS

Syntax:
ELECTRON-ANGLE-BINS [parameter(1)]

Example:
ELECTRON-ANGLE-BINS2
90 180
Default: 33 bins, unevenly distributed.

If one increases the number of substeps per step or decreases the step size, it may be necessary to use a finer binning structure for electron angular scattering to accurately represent the Goudsmit-Saunderson distribution.

This keyword must be followed by parameter(1) values. The angular binning begins with 0 degrees. The user must specify in degrees where each bin ends and the next begins. The final bin must end at 180 degrees.

5. ELECTRON-GRID-LENGTH

Syntax: ELECTRON-GRID-LENGTH [parameter(1)]

Example: ELECTRON-GRID-LENGTH 80
Default: 64, which corresponds to 8 halvings (or 0.39%) of the maximum energy.

Regardless of the value, the grid will be truncated below 1 keV.

Parameter(1) must be a multiple of 8.

In problems requiring a broad energy range of electron transport, this keyword allows the user to extend the energy range for which electron transport data is available. Alternatively, if the STEP keyword is used to refine the electron energy-loss grid, this keyword can be used to preserve the energy range over which data is available.

6. ENERGY

Syntax: ENERGY [parameter(1)]
Example: ENERGY 2.5
Default: Maximum cross-section energy is 1.0 MeV

Maximum energy in MeV for which electron cross sections will be calculated. Values may range between 1.0 GeV and 1.0 keV.

7. INCLUDE-FILE

S
y
nt
a
x:
I
N
C
L
U
D
E
-
FI
Example:
INCLUDEFILE
filename.
input
 All
This allows the user to include input parameters from other files.

8. MATERIAL

Syntax: MATERIAL [keyword] [parameter(1)] ...
Example: MATERIALTA0.25C0.75
Example: MATERIALH
Default: Aluminum

Identifies unique material (pure element, compound or homogeneous mixture) and the appropriate weight fractions for which electron and photon cross sections are to be calculated. This keyword is repeated for each
unique material. Each element symbol must be followed by a single real number, parameter(1), which is the weight fraction of that constituent, except for pure elements where a blank for parameter(1) will default to 1.0. The weight fractions must sum to 1.0.

DATA arrays containing 100 atomic symbols (e.g., Ta for tantalum) along with corresponding default values for the electrical characterization (conductor/non-conductor), mass density, and state (solid/liquid or gas) at normal pressure and temperature (zero °C and one atm) are included in the code to simplify the input for pure materials. The default properties of elements are provided in Table D.2. To override these defaults to construct compound materials, the following secondary keywords associated with this primary keyword may be used.

(a) **CONDUCTOR/NON-CONDUCTOR**

Syntax: NON-CONDUCTOR

Default: A pure element with a Z of 1, 2, 7, 8, 9, 10, 17, 18, 35, 36, 53, 54, 85, or 86 is a non-conductor; otherwise, the element is a conductor. A compound/mixture is a non-conductor if any one of its constituent elements is a non-conductor by default; otherwise, it is a conductor.

The only collective effect in the ITS Monte Carlo model is the density-effect correction to the electronic stopping power. The value of this correction depends on whether the transport region is a conductor or non-conductor. The user may explicitly define any material to be a conductor or a non-conductor via the appropriate keyword. However, if a material so defined as a conductor consists of constituent elements, all of which are non-conductors by default (e.g., pure water), the material will be redefined to be a non-conductor, and the user will be so informed via a message in the output file.

(b) **GAS**

Syntax:

```
S
y
n
taxa
x:
G
A
S
```
Default:

Normal state for elements an
words

used
to

specify

that

t
this

mate
sure and temperature. The material state is used in calculating the density effect contribution to the electronic stopping power.

(c) DENSITY

Syntax: DENSITY [parameter(1)]
Example: DENSITY 2.0

Default: Normal density for elements – no default for compounds! Density (in g/cm$^3$) of the target material at normal pressure and temperature.

(d)
Syntax:

DENSITY - RATIO

[ parameter (1) ]
Example:

DENSITY - RATIO

0.5

Default:

Default:
n
s
i
t
y
r
a
t
i
o
i
s
1
.0
R
a
t
i
o
o
f
th
e
a
c
t
u
a
l
density to the density of the target
material and normal pressure and temperature (used in calculating density effect contribution to electronic stopping powers).
STEP

Syntax:

STEP

[parameter (1)]

Ex
Sample:
SUBSTEP 10

Default:
Calculated
intrainnacle as a function of atomic number
which each microscopic electron step
is divided. The default values have been empirically determined; other values should not be used without careful consideration of their effects on the condensed history model.

9. PRINT-ALL

Syntax: PRINT-ALL
Default: Abbreviated cross-section tables will be printed

This keyword will cause all except differential electron cross-sections to be printed out.

10. STEP

Syntax: STEP [parameter(1)]  Example: STEP 12  Default:
Successive electron energies are related by \( E_{i+1} = 2^{-\frac{1}{8}} E_i \)

[parameter(1)] determines the spacing of the electron energy grid and the size of the macroscopic electron steps. Successive energies are related by \( E_{i+1} = 2^{-\frac{1}{[\text{parameter}(1)]}} E_i \). The default value has been empirically determined; other values should not be used without careful consideration of their effects on the condensed history model.

11. TITLE

Syntax:
TITLE
Example:
TITLE
Adjoint Dose calculation in Al
box in
Satellite
GPS-4
Default:
no title

This keywordsignals that the next line of input contains a character string that is a title of up to 80 columns that will be written to the output file.

D
K
e
y
w
or
d
s
f
o
r
X
G
E
N

T
a
b
l
e
D
2
L
i
s
t
o
f
a
v
ai
l
a
b
l
e
e
l
e
m
e
n
t
s
a
n
d
<table>
<thead>
<tr>
<th>Z</th>
<th>Element</th>
<th>Symbol</th>
<th>Atomic Weight</th>
<th>Density (g/cm³)</th>
<th>State</th>
<th>Conductor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hydrogen</td>
<td>H</td>
<td>1.008</td>
<td>0.08988E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>2</td>
<td>Helium</td>
<td>He</td>
<td>4.0026</td>
<td>0.1785E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>3</td>
<td>Lithium</td>
<td>Li</td>
<td>6.94</td>
<td>0.53</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>4</td>
<td>Beryllium</td>
<td>Be</td>
<td>9.01218</td>
<td>1.85</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>5</td>
<td>Boron</td>
<td>B</td>
<td>10.81</td>
<td>2.34</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>6</td>
<td>Carbon</td>
<td>C</td>
<td>12.011</td>
<td>2.26</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>7</td>
<td>Nitrogen</td>
<td>N</td>
<td>14.0067</td>
<td>2.506E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>8</td>
<td>Oxygen</td>
<td>O</td>
<td>15.9994</td>
<td>1.429E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>9</td>
<td>Fluorine</td>
<td>F</td>
<td>18.9984</td>
<td>1.696E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>10</td>
<td>Neon</td>
<td>Ne</td>
<td>20.17</td>
<td>0.89990E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>11</td>
<td>Sodium</td>
<td>Na</td>
<td>22.9898</td>
<td>0.97</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>12</td>
<td>Magnesium</td>
<td>Mg</td>
<td>24.305</td>
<td>1.74</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>13</td>
<td>Aluminum</td>
<td>Al</td>
<td>26.9815</td>
<td>2.70</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>14</td>
<td>Silicon</td>
<td>Si</td>
<td>28.086</td>
<td>2.33</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>15</td>
<td>Phosphorus</td>
<td>P</td>
<td>30.9738</td>
<td>1.82</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>16</td>
<td>Sulfur</td>
<td>S</td>
<td>32.06</td>
<td>2.07</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>17</td>
<td>Chlorine</td>
<td>Cl</td>
<td>35.453</td>
<td>3.214E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>18</td>
<td>Argon</td>
<td>Ar</td>
<td>39.948</td>
<td>1.7837E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>19</td>
<td>Potassium</td>
<td>K</td>
<td>39.102</td>
<td>0.86</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>20</td>
<td>Calcium</td>
<td>Ca</td>
<td>40.08</td>
<td>1.55</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>21</td>
<td>Scandium</td>
<td>Sc</td>
<td>44.9559</td>
<td>3.00</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>22</td>
<td>Titanium</td>
<td>Ti</td>
<td>47.90</td>
<td>4.51</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>23</td>
<td>Vanadium</td>
<td>V</td>
<td>50.941</td>
<td>6.10</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>24</td>
<td>Chromium</td>
<td>Cr</td>
<td>51.996</td>
<td>7.19</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>25</td>
<td>Manganese</td>
<td>Mn</td>
<td>54.938</td>
<td>7.43</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>26</td>
<td>Iron</td>
<td>Fe</td>
<td>55.847</td>
<td>7.86</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>27</td>
<td>Cobalt</td>
<td>Co</td>
<td>58.9332</td>
<td>8.90</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>28</td>
<td>Nickel</td>
<td>Ni</td>
<td>58.71</td>
<td>8.90</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>Copper</td>
<td>Cu</td>
<td>63.546</td>
<td>8.96</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>30</td>
<td>Zinc</td>
<td>Zn</td>
<td>65.37</td>
<td>7.14</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>31</td>
<td>Gallium</td>
<td>Ga</td>
<td>69.72</td>
<td>5.91</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>32</td>
<td>Germanium</td>
<td>Ge</td>
<td>72.59</td>
<td>5.32</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>33</td>
<td>Arsenic</td>
<td>As</td>
<td>74.9216</td>
<td>5.72</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>34</td>
<td>Selenium</td>
<td>Se</td>
<td>78.96</td>
<td>4.79</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>35</td>
<td>Bromine</td>
<td>Br</td>
<td>79.904</td>
<td>7.59E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>36</td>
<td>Krypton</td>
<td>Kr</td>
<td>83.80</td>
<td>3.733E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>37</td>
<td>Rubidium</td>
<td>Rb</td>
<td>85.467</td>
<td>1.53</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>38</td>
<td>Strontium</td>
<td>Sr</td>
<td>87.62</td>
<td>2.60</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>39</td>
<td>Yttrium</td>
<td>Y</td>
<td>88.9059</td>
<td>4.47</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>40</td>
<td>Zirconium</td>
<td>Zr</td>
<td>91.22</td>
<td>6.49</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>Z</td>
<td>Element</td>
<td>Symbol</td>
<td>Atomic Weight</td>
<td>Density (g/cm³)</td>
<td>State</td>
<td>Conductor</td>
</tr>
<tr>
<td>---</td>
<td>--------------</td>
<td>--------</td>
<td>---------------</td>
<td>----------------</td>
<td>-------</td>
<td>-----------</td>
</tr>
<tr>
<td>41</td>
<td>Niobium</td>
<td>Nb</td>
<td>92.9064</td>
<td>8.40</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>42</td>
<td>Molybdenum</td>
<td>Mo</td>
<td>95.94</td>
<td>10.2</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>43</td>
<td>Technetium</td>
<td>Tc</td>
<td>98.9062</td>
<td>11.5</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>44</td>
<td>Ruthenium</td>
<td>Ru</td>
<td>101.07</td>
<td>12.2</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>45</td>
<td>Rhodium</td>
<td>Rh</td>
<td>102.9055</td>
<td>12.4</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>46</td>
<td>Palladium</td>
<td>Pd</td>
<td>106.4</td>
<td>12.0</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>47</td>
<td>Silver</td>
<td>Ag</td>
<td>107.868</td>
<td>10.5</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>48</td>
<td>Cadmium</td>
<td>Cd</td>
<td>112.4</td>
<td>8.65</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>49</td>
<td>Indium</td>
<td>In</td>
<td>114.82</td>
<td>7.31</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>50</td>
<td>Tin</td>
<td>Sn</td>
<td>118.69</td>
<td>7.30</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>51</td>
<td>Antimony</td>
<td>Sb</td>
<td>121.75</td>
<td>6.62</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>52</td>
<td>Tellurium</td>
<td>Te</td>
<td>127.6</td>
<td>6.24</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>53</td>
<td>Iodine</td>
<td>I</td>
<td>126.9045</td>
<td>4.94</td>
<td>S/L</td>
<td>N</td>
</tr>
<tr>
<td>54</td>
<td>Xenon</td>
<td>Xe</td>
<td>131.3</td>
<td>5.887E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>55</td>
<td>Cesium</td>
<td>Cs</td>
<td>132.9055</td>
<td>1.90</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>56</td>
<td>Barium</td>
<td>Ba</td>
<td>137.34</td>
<td>3.50</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>57</td>
<td>Lanthanum</td>
<td>La</td>
<td>138.9055</td>
<td>6.17</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>58</td>
<td>Cerium</td>
<td>Ce</td>
<td>140.12</td>
<td>6.67</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>59</td>
<td>Praseodymium</td>
<td>Pr</td>
<td>140.9077</td>
<td>6.77</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>60</td>
<td>Neodymium</td>
<td>Nd</td>
<td>144.24</td>
<td>7.00</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>61</td>
<td>Promethium</td>
<td>Pm</td>
<td>145.0</td>
<td>7.22</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>62</td>
<td>Samarium</td>
<td>Sm</td>
<td>150.4</td>
<td>7.54</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>63</td>
<td>Europium</td>
<td>Eu</td>
<td>151.96</td>
<td>5.26</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>64</td>
<td>Gadolinium</td>
<td>Gd</td>
<td>157.25</td>
<td>7.89</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>65</td>
<td>Terbium</td>
<td>Tb</td>
<td>158.9254</td>
<td>8.27</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>66</td>
<td>Dysprosium</td>
<td>Dy</td>
<td>162.50</td>
<td>8.54</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>67</td>
<td>Holmium</td>
<td>Ho</td>
<td>164.9303</td>
<td>8.80</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>68</td>
<td>Erbium</td>
<td>Er</td>
<td>167.26</td>
<td>9.05</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>69</td>
<td>Thulium</td>
<td>Tm</td>
<td>168.9342</td>
<td>9.33</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>70</td>
<td>Ytterbium</td>
<td>Yb</td>
<td>173.04</td>
<td>6.98</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>71</td>
<td>Lutetium</td>
<td>Lu</td>
<td>174.97</td>
<td>9.84</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>72</td>
<td>Hafnium</td>
<td>Hf</td>
<td>178.49</td>
<td>13.1</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>73</td>
<td>Tantalum</td>
<td>Ta</td>
<td>180.947</td>
<td>16.6</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>74</td>
<td>Tungsten</td>
<td>W</td>
<td>183.85</td>
<td>19.3</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>75</td>
<td>Rhenium</td>
<td>Re</td>
<td>186.2</td>
<td>21.0</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>76</td>
<td>Osmium</td>
<td>Os</td>
<td>190.2</td>
<td>22.6</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>77</td>
<td>Iridium</td>
<td>Ir</td>
<td>192.22</td>
<td>22.5</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>78</td>
<td>Platinum</td>
<td>Pt</td>
<td>195.09</td>
<td>21.4</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>79</td>
<td>Gold</td>
<td>Au</td>
<td>196.9665</td>
<td>19.3</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>80</td>
<td>Mercury</td>
<td>Hg</td>
<td>200.59</td>
<td>13.6</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>Z</td>
<td>Element</td>
<td>Symbol</td>
<td>Atomic Weight</td>
<td>Density (g/cm³)</td>
<td>State</td>
<td>Conductor</td>
</tr>
<tr>
<td>---</td>
<td>--------------</td>
<td>--------</td>
<td>---------------</td>
<td>----------------</td>
<td>-------</td>
<td>-----------</td>
</tr>
<tr>
<td>81</td>
<td>Thallium</td>
<td>Tl</td>
<td>204.37</td>
<td>11.85</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>82</td>
<td>Lead</td>
<td>Pb</td>
<td>207.2</td>
<td>11.4</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>83</td>
<td>Bismuth</td>
<td>Bi</td>
<td>208.9806</td>
<td>9.8</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>84</td>
<td>Polonium</td>
<td>Po</td>
<td>209.0</td>
<td>9.2</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>85</td>
<td>Astatine</td>
<td>At</td>
<td>210.0</td>
<td>No default</td>
<td>S/L</td>
<td>N</td>
</tr>
<tr>
<td>86</td>
<td>Radon</td>
<td>Rn</td>
<td>222.0</td>
<td>9.73E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>87</td>
<td>Francium</td>
<td>Fr</td>
<td>223.0</td>
<td>No default</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>88</td>
<td>Radium</td>
<td>Ra</td>
<td>226.0</td>
<td>5.00</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>89</td>
<td>Actinium</td>
<td>Ac</td>
<td>227.0</td>
<td>10.07</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>90</td>
<td>Thorium</td>
<td>Th</td>
<td>232.0381</td>
<td>11.7</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>91</td>
<td>Protactinium</td>
<td>Pa</td>
<td>231.0359</td>
<td>15.4</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>92</td>
<td>Uranium</td>
<td>U</td>
<td>238.029</td>
<td>19.07</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>93</td>
<td>Neptunium</td>
<td>Np</td>
<td>237.0482</td>
<td>19.5</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>94</td>
<td>Plutonium</td>
<td>Pu</td>
<td>244.0</td>
<td>19.84</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>95</td>
<td>Americium</td>
<td>Am</td>
<td>243.0</td>
<td>11.7</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>96</td>
<td>Curium</td>
<td>Cm</td>
<td>247.0</td>
<td>13.51</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>97</td>
<td>Berkelium</td>
<td>Bk</td>
<td>247.0</td>
<td>14.0</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>98</td>
<td>Californium</td>
<td>Cf</td>
<td>251.0</td>
<td>No default</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>99</td>
<td>Einsteinium</td>
<td>Es</td>
<td>254.0</td>
<td>No default</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>100</td>
<td>Fermium</td>
<td>Fm</td>
<td>257.0</td>
<td>No default</td>
<td>S/L</td>
<td>Y</td>
</tr>
</tbody>
</table>

E Running CEPXS

This document contains outlines for running the CEPXS code. Instructions are provided for running the code by using commands or by using scripts. Known platform dependencies of the code are discussed in the last section.

E.1 Running CEPXS without Scripts

CHECKOUT: Do a "cvs checkout -P cepxs" to acquire a copy of the code. If necessary, tar the directory and transfer it to the desired platform.
To run on any platform, either use the sendn script (described in the Running CEPXS with Scripts section) or build an executable using the following commands.

CONFIGURE: In the directory cepxs/Code, execute "./configure". If the platform and operating system are identified, but the proper config/mh-* and config/mt-* files are not present, then you must either identify the proper files to use or create the necessary files and alter the configure.in file accordingly. If the platform and operating system are not identified, the config.sub file also must be altered (search for tflops as an example).

MAKE: If configure functions properly, a working Makefile will be produced. Execute this with the "make" command to produce the CEPXS executable, xcepxs.

ATOMIC DATA: The code xcepxs will look for the atomic data files in the local directory. These files are in the cepxs directory under cepxs/Code/XSdata. Soft links may be created from the location of the atomic data files to one's working directory as:

```
ln -s $HOME/cepxs/Code/XSdata/elec3.dat elec3.dat
```

REGRESSION TESTS: Files for running regression tests are in cepxs/Tests/RegTests. The "runregtests" script will make an executable, run each problem, place output files in the directory NewOutput, and place results of diffs with the repository version of output files in the directory Diffs.

### E.2 Running CEPXS with Scripts

ACVS checkout may be performed on platforms that have direct access to the CVS repository. On all other platforms, the directory of files must be tarred and transferred, and the directory options in the sendn script must always be used instead of the CVS options.

For working on platforms that do not have direct access to the repository, it is recommended that two copies of the repository checkout be set up: one copy to be used as a working directory in which code modifications and builds can be performed, and one unaltered copy that can be compared with the maintained record of changes made to the working version. The unaltered copy should be give a name uniquely indicating the cvs version.

1. SCRIPTS: Use the command "cvs checkout -P cepxs" to acquire a copy of the code and all associated files.
(a) From the directory cepxs/Scripts, copy sendn and the ncCVS script to your $HOME/bin directory. Copy the files in the Subscripts directory to a $HOME/bin/Subscripts directory. sendn and the Subscripts are identical for ITS, XGEN, and CEPXS.

2. CUI FILE: Copy the cepxs/Scripts/cepxs.cui file to your working directory, and edit it for your specific problem. The sections of a cui file are:

   (a) DRIVER SCRIPT: The script ncCVS is available as a driver script. You may substitute a customized script by including it in the first section of the cui file. Sendn will position the cui file, including the sections of the driver script. The driver script contains the commands necessary to build and execute the program, clean up after itself, and produce relevant result information in a job file. (If necessary, it will also include in the job file information useful in determining the cause of a job failure).

   (b) DEFS: There are currently no preprocessor definitions to be set for CEPXS. There are several options for running the scripts that may be set in the defs section of the cui file. Compiler flags may be specified. The user may also request an interactive script. The interactive script should be used on systems that require job queuing or on systems that require cross compiling. The first half of the script will build the executable. The files for running the job will be located in $HOME/tmp/<jobname>. The executable is named "xcepxs". When the calculation is complete, the user may execute the "postproc" file in the $HOME/tmp/<jobname> directory. This will move the fort.11 cross section file to $HOME/crossm directory, rename it according to the response that was given to the sendn script, and produce a job file in the directory from which the job was originally submitted.

   (c) DIFFS: Patches to be applied to the code should be inserted in the "diffs" portion of the cepxs.cui file. These patches can be formatted as a cvs diff or as a directory diff. Diff can be lifted out of job files from previous calculations. Alternatively, one can checkout a copy of the code, make modifications, and then generate a diffs file. On a platform with access to the CVS repository, one can use "cvs diff > diffs" from within the cepxs/Code directory. On a platform without access to the CVS repository one can perform a directory diff, but this must be performed from within the cepxs/Code directory of the unaltered copy of the code, and it must use the -r and -b flags (in that order), such as "diff-r -b $HOME/cepxsaltered/Code> diffs". Then, the diffs file can be used in the cepxs.cui file. Multiple diff files (diffs from two different executions of the diff command) cannot be patched to a single cvs file. New files can be added to the build. Within the diffs section, a line of the following format denotes the start of a new file: New file: <Directory/Filename> Tobe included in the compiling and linking of the code, then new file must be referenced in the Makefile.
list of sources. This change in the Makefile.in can also be included in the diffs section by making the desired change in a copy of Makefile.in and using the cvs diff procedure described above.

(d) CEPINP: The CEPXS input should be included in the last section of the cui file. Instructions for CEPXS input are in Keywords for CEPXS.

3. SENDN: Submit the job using the command “sendn cepxs.cui <jobname>”.
   This script will prompt you for the following 3 pieces of information (and possibly the 4th depending upon your responses to the first 3):

   (a) CROSS SECTIONS: First, sendn will request the name of the cross section file <cross> to be created. Upon successful completion, the scripts will assign the given name to the fort.11 cross section file and place it at “$HOME/crossm/<cross>.11”.

   (b) LOCAL COMPILE: Second, sendn will request the location of a checked-out copy of cepxsthat can be used for making the executable. Thus, you may use a version of the code that you have checked out and modified. You must give a complete pathname for the base cepxs directory (e.g., /scratch/temporary/cepxs). No files will be deleted from the make directory as a result of the calculation, but some files may be modified if requested in the diffs section of the cui file. Any new files that have been included in the directory (other than through the diffs section of the cui file) will not appear in the job file, but they may be used if the Makefile.in has been so modified (in which case, the job file will show the reference to the new file as a difference in the Makefile.in). Be aware that any definitions specified in the “defs” section of the cui file and any modifications specified in the “diffs” section of the cui file will be applied to the files in this directory. Attempts to change the code definitions for a calculation using the same directory will not take effect unless “makeclean” has been executed. Attempts to apply the diffs in a directory where the diffs have already been applied for a previous calculation will result in an error.

   (c) DIRECTORYDIFF for LOCAL COMPILE: If your response to the
second request was a directory pathname, the script will request a local directory with which to perform a diff. This should be the unaltered copy that you made when you checked out. The diff command will be used to compare the two directories and all subdirectories. The job file will not record the version number of either directory, therefore the user may not have enough information in the job file to duplicate a calculation unless the diff directory has a name corresponding to the tag used in the cvs checkout of the code. The directory name does appear in the job file. VERSION for CVS COMPIL E: If your response to the second request was “none”, the script will request the version for a cvs checkout. The command will be issued as “cvs checkout <options> cepxs”. The response to this request will be used for the <options> and any valid cvs flags may be used that do not contain a slash, “/”. Examples of valid syntax for responses are: -D now, -D “March 28, 2001”, -D 4:00pm, -D “3 hours ago”, -D “2 fortnights ago”, -r 1, -r release-1.0, etc. Another valid response is to simply press “Enter” with no input, which will result in the checkout of the most recent version of the code.

(d) CVSDIFF for LOCAL COMPIL E: If you gave a pathname for making the executable but not for a directory diff, the script will request the version for a cvs diff. The syntax for responses to this request are the same as for specifying the cvs version for a checkout. The directory in which the executable is made will be compared with the repository using the cvs diff command. This version (that will appear in the job file) and the results of the cvs diff (that will also appear in the job file) can be used to reproduce a calculation. CVSUPDATE/DIFF for CVS COMPIL E: If you did not give a pathname for making the executable, the script will request a version for a cvs update and diff. In this case, the

script will request a local directory with which to perform a diff. The diff command will be used to compare the two directories and all subdirectories. The job file will not record the version number of either directory, therefore the user may not have enough information in the job file to duplicate a calculation unless the diff directory has a name corresponding to the tag used in the cvs checkout of the code. The directory name does appear in the job file. VERSION for CVS COMPIL E: If your response to the second request was “none”, the script will request the version for a cvs checkout. The command will be issued as “cvs checkout <options> cepxs”. The response to this request will be used for the <options> and any valid cvs flags may be used that do not contain a slash, “/”. Examples of valid syntax for responses are: -D now, -D “March 28, 2001”, -D 4:00pm, -D “3 hours ago”, -D “2 fortnights ago”, -r 1, -r release-1.0, etc. Another valid response is to simply press “Enter” with no input, which will result in the checkout of the most recent version of the code.

(d) CVSDIFF for LOCAL COMPIL E: If you gave a pathname for making the executable but not for a directory diff, the script will request the version for a cvs diff. The syntax for responses to this request are the same as for specifying the cvs version for a checkout. The directory in which the executable is made will be compared with the repository using the cvs diff command. This version (that will appear in the job file) and the results of the cvs diff (that will also appear in the job file) can be used to reproduce a calculation. CVSUPDATE/DIFF for CVS COMPIL E: If you did not give a pathname for making the executable, the script will request a version for a cvs update and diff. In this case, the

script will request a local directory with which to perform a diff. The diff command will be used to compare the two directories and all subdirectories. The job file will not record the version number of either directory, therefore the user may not have enough information in the job file to duplicate a calculation unless the diff directory has a name corresponding to the tag used in the cvs checkout of the code. The directory name does appear in the job file. VERSION for CVS COMPIL E: If your response to the second request was “none”, the script will request the version for a cvs checkout. The command will be issued as “cvs checkout <options> cepxs”. The response to this request will be used for the <options> and any valid cvs flags may be used that do not contain a slash, “/”. Examples of valid syntax for responses are: -D now, -D “March 28, 2001”, -D 4:00pm, -D “3 hours ago”, -D “2 fortnights ago”, -r 1, -r release-1.0, etc. Another valid response is to simply press “Enter” with no input, which will result in the checkout of the most recent version of the code.

(d) CVSDIFF for LOCAL COMPIL E: If you gave a pathname for making the executable but not for a directory diff, the script will request the version for a cvs diff. The syntax for responses to this request are the same as for specifying the cvs version for a checkout. The directory in which the executable is made will be compared with the repository using the cvs diff command. This version (that will appear in the job file) and the results of the cvs diff (that will also appear in the job file) can be used to reproduce a calculation. CVSUPDATE/DIFF for CVS COMPIL E: If you did not give a pathname for making the executable, the script will request a version for a cvs update and diff. In this case, the

script will request a local directory with which to perform a diff. The diff command will be used to compare the two directories and all subdirectories. The job file will not record the version number of either directory, therefore the user may not have enough information in the job file to duplicate a calculation unless the diff directory has a name corresponding to the tag used in the cvs checkout of the code. The directory name does appear in the job file. VERSION for CVS COMPIL E: If your response to the second request was “none”, the script will request the version for a cvs checkout. The command will be issued as “cvs checkout <options> cepxs”. The response to this request will be used for the <options> and any valid cvs flags may be used that do not contain a slash, “/”. Examples of valid syntax for responses are: -D now, -D “March 28, 2001”, -D 4:00pm, -D “3 hours ago”, -D “2 fortnights ago”, -r 1, -r release-1.0, etc. Another valid response is to simply press “Enter” with no input, which will result in the checkout of the most recent version of the code.
appear in the job file) can be used to reproduce a calculation. For either of these options, if no option is specified for the cvs diff, the option "-D now" will be used. The date and time of the calculation, which are recorded in the job file, and the result of the cvs diff can be used to reproduce the calculation at a later date.

E.3 Platform Dependencies

The following is a list of platforms on which CEPXS has been successfully built and tested. Following the name of the platform is the system type and operating system.

- Crater (AMD Athlon, Linux)
- Pegasus (64-bit Intel, Linux)
- PC (Microsoft Visual Studio)

See the Installation section for guidance on installing CEPXS on a PC.

Summary of CEPXS Keywords

Last Modified Date: 2008/04/17 21:39:25

F Summary of CEPXS Keywords

Table F.3 contains a listing of keywords for the CEPXS code for generating cross sections to be used with the MITS code. The keywords are listed approximately in order of importance. In addition, for each keyword the default code behavior is listed. The default behavior will be employed by the code if the keyword is not found in the input deck. More detailed descriptions of the syntax, sub-keywords, and use of these keywords are contained in the CEPXS Keywords section.

F.
Summary of CEPXS Keywords

Table F.3: Summary of CEPXS keywords (for use with MITS) and default settings

<table>
<thead>
<tr>
<th>KEYWORD</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>MITS</td>
<td>ONELD</td>
</tr>
<tr>
<td>**** MATERIALS ****</td>
<td></td>
</tr>
<tr>
<td>MATERIAL or MATNAM</td>
<td>required (repeated for each material)</td>
</tr>
<tr>
<td>**** ENERGY RANGE ****</td>
<td></td>
</tr>
<tr>
<td>ENERGY</td>
<td>1.0 MeV</td>
</tr>
<tr>
<td><strong>CUTOFF</strong></td>
<td>1keV for photon sources 1% of source energy for electron sources</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>**** SPECIES/COUPLING ****</td>
<td></td>
</tr>
<tr>
<td>ELECTRON-SOURCE or PHOTON-SOURCE</td>
<td>electron source, full-coupling with photons</td>
</tr>
<tr>
<td>NO-POSITRONS</td>
<td>positrons automatic if energy extends above 1.03 MeV and full-coupling</td>
</tr>
<tr>
<td>**** GROUP STRUCTURE ****</td>
<td></td>
</tr>
<tr>
<td>ELECTRONS (or EGROUP)</td>
<td>50 logarithmic groups</td>
</tr>
<tr>
<td>PHOTONS (or PGROUP)</td>
<td>50 logarithmic groups</td>
</tr>
<tr>
<td>ANNIHILATION-LINE</td>
<td>automatic line if energy extends above 1.03 MeV</td>
</tr>
<tr>
<td>NO-LINES</td>
<td>annihilation line is automatic</td>
</tr>
<tr>
<td>LINES</td>
<td>relaxation lines are mixed with continuum</td>
</tr>
<tr>
<td>MONO-PHOTON</td>
<td>no source line groups</td>
</tr>
<tr>
<td>**** ALTERNATIVE PHYSICS MODELS ****</td>
<td></td>
</tr>
<tr>
<td>ITS2P1</td>
<td>cross sections correspond to ITS 3.0</td>
</tr>
<tr>
<td>USCATand/or USCAT-ITS</td>
<td>Fokker-Planck scattering cosine = 0.95</td>
</tr>
<tr>
<td>NO-PCODE</td>
<td>ITS-PCODES ionization/relaxation physics</td>
</tr>
<tr>
<td>NO-SEC-ELEC-GLOBAL</td>
<td>secondary electrons generated</td>
</tr>
<tr>
<td>NO-COHERENT</td>
<td>coherent photon scattering included</td>
</tr>
<tr>
<td>NO-INCOH-BINDING</td>
<td>incoherent scattering includes binding effects</td>
</tr>
<tr>
<td>CSDA</td>
<td>restricted CSDA</td>
</tr>
<tr>
<td>KNOCKONS-WITH-PRIMARIES</td>
<td>secondaries w/o corresp. primary downscatter</td>
</tr>
<tr>
<td>LEGENDRE</td>
<td>Legendre order is 15</td>
</tr>
<tr>
<td>ELASTIC-LEGENDRE</td>
<td>Legendre order is 15</td>
</tr>
<tr>
<td>**** OTHER OPTIONS ****</td>
<td></td>
</tr>
<tr>
<td>TITLE</td>
<td>no title</td>
</tr>
<tr>
<td>PRINT</td>
<td>cross section matrices not printed</td>
</tr>
<tr>
<td>PRINT-ALL</td>
<td>no &quot;additional&quot; information is printed</td>
</tr>
<tr>
<td>INCLUDE-FILE</td>
<td>all data in single input file</td>
</tr>
<tr>
<td>CONTRAST</td>
<td>compact fort.11 file</td>
</tr>
<tr>
<td>CEPXS-INFO-ONLY</td>
<td>cross sections are generated</td>
</tr>
</tbody>
</table>
G Keywords for CEPXS

The input keywords must be specified in either the CUI file (if using the scripts to execute the code) or in the input file (if executing the code manually). Refer to the documentation on Running CEPXS for more details on specifying the input file in the CUI file or otherwise. An overview of the keywords is available in the Summary of CEPXS Keywords section.

G.1 Input Notation

Most keywords should be used once and not repeated in an input file. Exceptions to this are the MATERIAL, MATNAM, and MONO-PHOTON keywords. Most subkeywords should be used once per use of their primary keyword.

Comments may be inserted in the input deck. Lines with an asterisk in the first column will be ignored by the code. It is important to note that asterisks inserted anywhere other than the first column may not cause words that follow to be ignored by the code.

Input is not case sensitive, with one exception. Character input provided with the TITLE keyword will appear in the output file exactly as provided, but the case does not affect the code in any way.

G.2 Keywords

Transport Code Keywords

1. ONELD

Syntax: ONELD
Default: ONELD

This keyword specifies that the cross sections from CEPXS will be formatted for ONELD.

2. MITS

Syntax: MITS
Default: ONELD

This keyword specifies that the cross sections from CEPXS will be formatted for MITS.

3. BFP (Not MITS, Not ONELD)
Syntax: BFP
Default: The Boltzmann-Fokker-Planck approximation is not used.

This keyword specifies that the Boltzmann-Fokker-Planck approximation is to be used for electron cross sections. This approximation is automatically used if the MITS keyword is used.

G. Keywords for CEPXs

Additional Keywords

1. ANNIHILATION-LINE

   Syntax: ANNIHILATION-LINE
   Default: Annihilation line is included if energy grid extends above 1.03 MeV, unless keyword NO-LINES is used. (See the LINES keyword for explanation of line groups.)

   Specifies that the annihilation line will be separated from the continuum, if 0.511 MeV is within the energy range. This keyword overrides the NO-LINES keyword and overrides the requirement that the energy grid extend above 1.03 MeV.

   Note: This keyword is incompatible with PGROUP.

2. BCD (NOT CURRENTLY FUNCTIONAL)

   Syntax: BCD

   This keyword specifies the format needed for MCNP.

3. CEPXS-INFO-ONLY
Syntax: CEPXS-INFO-ONLY

Default: Cross sections are calculated.

This keywords specifies that only range, mfp, and other such information is to be generated, but cross sections will not be calculated.

4. **CONTRAST** *(MITS)*

   Syntax: CONTRAST
   Default: A compact fort.11 file is produced for MITS. (No fort.11 file is produced unless MITS is specified.)

   An expanded fort.11 file is produced that can be compared with fort.11 files from other versions of CEPXS using the contrast program. This fort.11 file is still functional for MITS, but it has larger memory requirements.

5. **CONDENSED-HISTORY** *(Not MITS)*

   Syntax: CONDENSED-HISTORY
   Default: A diamond energy-differencing scheme is used for the CSDA operator.

   A second order backward differencing of the CSDA operator is used, and a condensed-history based approximation is used.

6. **CSDA**

   Syntax: CSDA
   Default: Restricted CSDA is used.

The Continuous-Slowing-Down Approximation (CSDA) is used to characterize all electron energy loss.
7. **CSDLD (Not MITS)**

Syntax: CSDLD  
Default: A diamond energy-differencing scheme is used for the CSDA operator.

8. **CUTOFF**

Syntax: CUTOFF [parameter(1)] Example: CUTOFF 0.01  
Default: 1 keV for photon sources. One percent of the source energy for electron sources.

Cross sections extend to the cutoff energy. This energy is the lower bound of the lowest energy group for both electrons and photons. [parameter(1)] is the cutoff energy in MeV, i.e., it specifies the lower energy bound of the lowest-energy group to be generated. The cutoff energy cannot be less than 1 keV.

9. **EGROUP**

Syntax: EGROUP  
Default: Fifty logarithmic electron groups.

The electron group structure is obtained from the ASCII file `egroup`. An `egroup` file is generated on each CEPXS run and may be used on a subsequent CEPXS run. The structure of the `egroup` file is not important unless the user wishes to create this file from scratch. On the first line of the file, the number of electron groups is specified. Each subsequent line of the file is associated with a group (in descending order of energy.) Each line specifies, in order, the top energy of the group, the mid-point energy of the group, and the bottom energy of the group, all in MeV. The code will function provided that the midpoint energies are within the energy group bounds, even if they are not exact mid-point values.

10. **ELASTIC-LEGENDRE**

Syntax: ELASTIC-LEGENDRE [parameter(1)]  
Example: ELASTIC-LEGENDRE5  
Default: 15, or Legendre order of cross sections set with LEGENDRE keyword.
This keyword specifies the Legendre order of electron elastic cross sections as \( \text{[parameter(1)]} \). This order is used if between 0 and the Legendre order of the cross sections. Otherwise, it is reset to the Legendre order of the cross sections.

Note: This keyword should be used with caution. For ONELD, the extended transport correction is applied, and it is important to consider the quadrature set with which the cross sections will be used.

11. ELECTRONS

Syntax: ELECTRONS Default:
For MITS, 50 logarithmic electron groups. For other codes, up to 50 automatic-structure electron groups.

If this primary keyword is used, one of the secondary keywords must be used to specify the electron group structure.

Note: This keyword is incompatible with EGROUP.

(a) LINEAR
Syntax: LINEAR [parameter(1)]
Example: LINEAR 40

A linear electron group structure is created where \([\text{parameter(1)}]\) is the number of electron groups.

(b) LOG
Syntax: LOG [parameter(1)]
Example: LOG 40

A logarithmic electron group structure is created where \([\text{parameter(1)}]\) is the number of electron groups.

(c) USER

Syntax: USER [parameter(1)]
[parameter(2)] ...
[parameter(parameter(1)+1)]

Example: USER 10
1.0 0.8 0.6 0.5 0.4 0.35 0.3 0.28 0.26 0.25

A user-defined electron group structure is created where [parameter(1)] is the number of electron groups. The string that follows is composed of the lower boundary energies of each group arranged in decreasing order. The last energy must be the same as the cutoff energy, i.e., [parameter(parameter(1)+1)] must equal CUTOFF [parameter(1)].

12. ELECTRON-SOURCE

Syntax: ELECTRON-SOURCE
Default: For MITS, a fully-coupled electron source. A source must be specified for other codes.

The following sub-keywords are used to specify the coupling scheme for photon and electron cross sections.

(a) NO-COUPLING

Syntax: NO-COUPLING

Default: Full-coupling. This keyword indicates that photons will not be included in the calculation.

(b) PARTIAL-COUPLING

Syntax: PARTIAL-COUPLING
Default: Full-coupling.

This keyword indicates that electrons can produce photons but photons cannot produce electrons.
(c) **FULL-COUPLING**

Syntax: FULL-COUPLING  
Default: Full-coupling.

This keyword indicates that electrons can produce photons and photons can produce electrons. Also, if the upper bound of the energy grid extends above 1.03 MeV, positrons will be included in the cross sections with full coupling to photons and partial coupling to electrons.

13. **ENERGY**

Syntax: ENERGY parameter(1)  
Example: ENERGY 0.665  
Default: 1 MeV

**parameter(1)** specifies the midpoint energy in MeV of the highest-energy group to be generated (but see description of energy grid generation for caveats). This energy cannot exceed 100 MeV.

14. **FIRST-ORDER** *(Not MITS)*

Syntax: FIRST-ORDER  
Default: A diamond energy-differencing scheme is used for the CSDA operator.

Cross sections associated with the CSD operator will be equivalent to a first-order differencing in energy.

15. **INCOH-BINDING**

Syntax: INCOH-BINDING  
Example: INCOH-BINDING  
Default: No incoherent binding for ONELD. Incoherent binding for MITS.

This keyword specifies that incoherent binding effects will be included. Either of the keywords ITS2P1 or NO-INCOH-BINDING specifies that incoherent binding effects will not be included.

16. **INCLUDE-FILE**

S  
y  
n  
t
ax:

INCLUDE

FILE

Example:

INCLUDE

FILE
Default: All input is read from a single file.

This allows the user to include input parameters from other files.

17. **ITS2P1**

Syntax: ITS2P1

Default: ITS version 3.0 physics.

Cross sections corresponding to ITS version 2.1 are used.

18. **KNOCKONS-WITH-PRIMARIES**

Syntax: KNOCKONS-WITH-PRIMARIES

Default: Knockon electrons are generated without corresponding primary down-scatter.

Knockon electron production is associated with primary downscatter.
19. LEGENDRE

Syntax: LEGENDRE [parameter(1)]
Example: LEGENDRE7
Default: The Legendre order of the cross sections is 15.

[parameter(1)] is the Legendre order of the cross sections. If parameter(1) is zero or omitted, the default will be used.

20. LINES

Syntax: LINES [parameter(1)] [parameter(2)] ...
Example: LINES AL BE
Default: photon relaxation lines are mixed with the continuum.

Specifies that separate groups are generated for the photon relaxation radiation of the elements specified by the parameters. The binding energies of these elements will also fall on the photon group boundaries (the group structure is adjusted accordingly.) If no parameters are specified, relaxation line groups will be created for all elements in the problem.

Line groups have assigned energy group widths corresponding to the continuum group with which they would otherwise be mixed and mid-point energies equal to the line energy. The cross sections for the line groups are calculated at the line energies.

Note: This keyword is incompatible with PGROUP.

21. MATERIAL

Syntax:
MATERIAL
[parameter(1)]
parameter(2) ...
Example:
MATERIAL H .111 .8889
Keywords for CEPXS

Default: None. At least one MATERIAL or MATNAM entry is required.

Specifies material composition. The mandatory string specifies either the name of a hard-wired material or the chemical names of the constituent elements and their weight fractions. In the latter case, the first parameter in the string is mandatory and is the chemical name of an element in the material. The second parameter is the weight fraction of the element in the material. For single element materials, the weight fraction is not needed (and will be in error if not equal to 1.0).

Additional parameters may specify other elements and their weight fractions. If necessary, this parameter list may be extended to other lines by terminating a line with a dash. The weight fractions of elements in a material must sum to 1.0.

The default properties of elements are provided in Table G.4. The hardwired materials available in CEPXS and the default properties of those materials are provided in Table G.5.

(a) **CONDUCTOR** Syntax: CONDUCTOR Default: Default values are provided for single element materials. A compound is set as a conductor unless any one of its constituent elements is a non-conductor. The only collective effect in the CEPXS model is the density-effect correction to the electronic stopping power. The value of this correction can depend on whether a material is a conductor or a non-conductor. Note: If a compound is specified as a conductor, but none of its constituent elements are conductors, then the material is automatically reset to a non-conductor.

Note: This keyword is incompatible with ITS2P1.

(b) **DENSITY** Syntax: DENSITY [parameter(1)] Example: DENSITY 1.001 Default: For single element materials and hard-wired materials, defaults are provided. For a user-defined composition, this keyword is required.

The density of the material is set to [parameter(1)] in g/cm³.
(c) **GAS**

Syntax: GAS

Default: Solid or Liquid.

This keyword specifies that the material is a gas.

(d) **NON-CONDUCTOR** Syntax: NON-CONDUCTOR

Default: Default values are provided for single element materials. A compound is set as a conductor unless any one of its constituent elements is a non-conductor. The only collective effect in the CEPXS model is the density-effect correction to the electronic stopping power. The value of this correction can depend on whether a material is a conductor or a non-conductor.

Note: This keyword is incompatible with ITS2P1.

(e) **NO-SEC-ELEC**

Syntax: NO-SEC-ELEC

Default: Secondary electrons are generated. This keyword specifies that secondary electrons are not generated in this material.
22. **MATNAM**

Syntax: MATNAM [parameter(1)]

[parameter(2)] parameter(3) ...

Example: MATNAMWATER

H0.1111O0.8889

Default: None. At least one MATERIAL or MATNAM entry is required.

This keyword is an alternative to the MATERIAL keyword. It allows the user to assign a material name to any material whether an element, a hard-wired material, or a user-defined composition. The code will use the first six characters of the user-assigned material name, [parameter(1)]. This user-assigned name will only appear in the CEPXS output file, cepout, and is not available for use in the generated cross section files.

Unlike the MATERIAL keyword, the material composition is specified on the following line. However, all of the same secondary keywords apply.

23. **MONO-PHOTON**

Syntax: MONO-PHOTON [parameter(1)]

Example: MONO-PHOTON 1.311

Default: No source line groups.

A mono-energetic source line group is created at the energy specified by [parameter(1)] in MeV. This keyword must be repeated for each source line needed. Upto 28 lines are allowed.

If the source line is meant to be the highest energy in the calculation, it should be set exactly equal to the energy specified with the ENERGY keyword. This will result in the source line being the highest energy group, with the second highest energy group spanning the energy range between it and the third highest energy group. (ENERGY specified the mid-point energy of the highest energy group. This is still the case, but now the highest energy group is a line group.) If the source line is otherwise specified to fall within the highest energy group, it will still be the first group, but the second group will extend in energy both above and below the line source group and therefore, may not yield the desired results.

If the energy specified for the source line is higher or lower than the extent of the energy grid, it will be omitted.

Note: In MITS, source lines are not allowed in adjoint mode.

Note: For ONELD, PHOTON-SOURCE must be specified to use MONO-
24. **NO-COHERENT**

Syntax: NO-COHERENT

Default: Coherent photon scattering will be included in the calculation.

This keyword deactivates the simulation of coherent photon scattering. This provides a functionality equivalent to the NO-COHERENT keyword in ITS.

25. **NO-INCOH-BINDING**

Syntax: NO-INCOH-BINDING

Default: Incoherent photon scattering will include binding effects.

This keyword causes incoherent photon scattering to be simulated in the Klein-Nishina or free-electron approximation. This provides a functionality equivalent to the NO-INCOH-BINDING keyword in ITS.

26. **NO-LINES**

Syntax: NO-LINES

Default: An annihilation line is included if the energy grid extends above 1.03 MeV.

This keyword specifies that no line groups will be included. The LINES keyword will override this keyword and cause photon relaxation line groups to be included. The ANNIHILATIONLINE keyword will override this keyword and cause an annihilation line group to be included.

27. **NO-PCODE**

Syntax: NO-PCODE

Default: CEPXS essentially duplicates the ionization/relaxation physics of the ITS PCODES.
This keyword is used to select the ionization/relaxation physics of the standard (non-PCODES) ITS codes. This option allows only the K-shell to have non-zero binding energy. A single “average” energy for the fluorescence photons and Auger electrons is used. This “average” energy is less than the K-shell binding energy.

28. **NO-POSITRONS**

Syntax: NO-POSITRONS  
Default: Positrons are generated if the upper limit of the energy grid exceeds 1.03 MeV and cross sections are fully-coupled.

Positrons are not generated. Annihilation quanta are generated at the site of the pair interaction. The approximate treatment for positrons must be specified by one of the secondary keywords.

(a) **PEQE**

Syntax: PEQE  
Default: None.

Pair secondaries are reproduced and transported as electrons. That is, annihilation quanta are produced at the beginning rather than end of the positron path. This option provides a reasonable estimate for dose but an incorrect charge prediction.

(b) **NO-PAIR**

Syntax: NO-PAIR  
Default: None.

No pair secondaries are reproduced. Energy and charge are redeposited locally. This option provides a reasonable estimate for charge but an incorrect dose prediction.
29. **NO-SEC-ELEC-GLOBAL**

Syntax: NO-SEC-ELEC-GLOBAL

Default: Secondary electrons are generated in all materials.

Neither photon-produced nor electron-produced secondary electrons are generated in any material.

30. **PGROUP**

Syntax: PGROUP

Default: Fifty logarithmic photon groups.

The photon group structure is obtained from the ASCII file, "pgroup". A “pgroup” file is generated on each CEPXS run and may be used on a subsequent CEPXS run. The structure of the "pgroup" file is not important unless the user wishes to create this file from scratch. On the first line of the file, the number of photon groups is specified. For MITS, the first line also contains the following (in order): the value of USCAT, a logical flag indicating whether positrons are present, and a logical flag indicating whether the NO-PAIR keywords are used. Each subsequent line of the file is associated with a group (in descending order of energy.) Each line specifies, in order, the top energy of the group in MeV, the midpoint energy of the group in MeV, the bottom energy of the group in MeV, and an integer that is zero for a continuum group or one for a line group (relaxation, source, or annihilation). The code will function provided that the mid-point energies are within the energy group bounds, even if they are not exact mid-point values. For line groups, the energy bounds should be the energy bounds of the group in which the line would otherwise be included, and the mid-point energy should be the line energy.

The line in the “pgroup” file following the energy group structure specifies the number of relaxation lines (not including annihilation and source lines) and the group that contains the annihilation line (zero if not applicable). On the following lines, one line gives the energy of an relaxation line followed by the Z value of the associated element and the next line contains three character fields (a3, a4, a4) that describe the relaxation line type.

31. **PHOTONS**

Syntax: PHOTONS

Default: For MITS, 50 logarithmic photon groups. For other codes, up to 50 automatic-structure photon groups.
If this primary keyword is used, one of the secondary keywords must be used to specify the photon group structure.
Note: This keyword is incompatible with PGROUP.

(a) LINEAR
Syntax: LINEAR [parameter(1)]
Example: LINEAR 40

A linear photon group structure is created where [parameter(1)] is the number of photon groups.

(b) LOG
Syntax: LOG [parameter(1)]
Example: LOG 40

A logarithmic photon group structure is created where [parameter(1)] is the number of photon groups.

(c) USER

Syntax: USER [parameter(1)]

[parameter(2)] ...
[parameter(parameter(1)+1)]
Example: USER 10

1.0 0.8 0.6 0.5 0.4 0.35 0.3 0.28 0.26 0.25

A user-defined photon group structure is created where [parameter(1)] is the number of photon groups. The string that follows is composed of the lower boundary energies of each group arranged in decreasing order. The last energy should be the same as the cutoff energy, i.e., [parameter(parameter(1)+1)] must equal CUTOFF [parameter(1)].

32. PHOTON-SOURCE

Syntax: PHOTON-SOURCE Default:
For MITS only, transport is independent of the source specified and the cross sections are fully-coupled by default. A source must be specified for other codes.

The following sub-keywords are used to specify the coupling scheme for photon and electron cross sections.

(a) **NO-COUPLING**
**Syntax:** NO-COUPLING

Default: Full-coupling. This keyword indicates that electrons will not be included in the calculation.

(b) **PARTIAL-COUPLING**
**Syntax:** PARTIAL-COUPLING
**Default:** Full-coupling.

This keyword indicates that photons can produce electrons but electrons cannot produce photons.

(c) **FULL-COUPLING**
**Syntax:** FULL-COUPLING
**Default:** Full-coupling.

This keyword indicates that photons can produce electrons and electrons can produce photons. Also, if the upper bound of the energy grid extends above 1.03 MeV, positrons will be included in the cross sections with full coupling to photons and partial coupling to electrons.

33. **PRINT**

**Syntax:** PRINT
**Default:** Multigroup Legendre cross section matrices are not
This keywords specifies that the multigroup Legendre cross section matrices are to be printed to the CEPSX output file, CEPOUT.

(a) **LEG**

Syntax: LEG [parameter(1)]

Example: LEG3

Default: Only the zero Legendre order cross sections are printed.

This sub-keyword specifies that multigroup Legendre cross sections are to be printed from Legendre order zero to Legendre order [parameter(1)].

(b) **ROWS**

Syntax: ROWS

Default: All cross section rows are printed.

This sub-keyword specifies that multigroup Legendre cross sections are to be printed for only selected rows of cross sections. More specifically, the total cross section, the energy deposition cross section, etc., will be printed, but group-to-group scattering cross sections will not be printed except for self-scatter cross sections.

34. **PRINT-ALL**

Syntax: PRINT-ALL

Default: No “additional” information will be printed.

This keyword requests that additional information be printed out for each material. This information includes:

- The collisional and radiative stopping powers and the density effect correction.
- The collisional stopping power due to large-energy losses.
- The exponent for the power-law extrapolation of the collisional stopping power below 10 keV.

35. **SECOND-ORDER** *(Not MITS)*

G. K ey w
or
ds
for
C
E
P
X
S

Syntax: SECOND-ORDER Example: SECOND-ORDER
Default: A diamond energy-differencing scheme is used for the
CSDA operator.

A second order backward differencing of the CSDA operator is used.

36. **TITLE**

    Sy
    nt
    ax:
    TI
    TL
    E
    pa
    ra
    m
    et
    er(1)
    Ex
    a
    m
    ple:
    TI
    TL
    E
    3
    M
    eV
    ele
This keywords specifies the title of the calculation that will appear in output files. The title will be read from the following parameter(1) lines. If parameter(1) is zero or omitted, the following (one) line will be read as the title. The title card may contain any alphanumeric information in columns 1 to 72 with which the user wishes to identify the calculation.

37. USCAT (MITS)

Syntax: USCAT[parameter(1)]

Example: USCAT0.99

Default: The cosine of the Fokker-Planck scattering angle is 0.95 for all energies and materials.

This keyword sets the cosine of the Fokker-Planck scattering angle to [parameter(1)].

38. USCAT-ITS (MITS)

Syntax: USCAT-ITS

Default: The cosine of the Fokker-Planck scattering angle is 0.95 for all energies and materials.

Energy-and material-dependent values of the Fokker-Planck scattering angle are internally calculated such that the electron mean free path is equal to the electron sub-step size in ITS. However, the automatic generation of Fokker-Planck scattering angles is over-ridden in cases where the scattering cosine would be less than the constant “uscat” parameter (which has a default value of 0.95 but may be changed with the
Table G. 4. List of available elements and default properties in CEPX

<table>
<thead>
<tr>
<th>Z</th>
<th>Element</th>
<th>Symbol</th>
<th>Atomic Weight</th>
<th>Density (g/cm³)</th>
<th>State</th>
<th>Conductor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hydrogen</td>
<td>H</td>
<td>1.00794</td>
<td>0.08375E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Symbol</td>
<td>Atomic Number</td>
<td>First Ionization Energy</td>
<td>State</td>
<td>Reactivity</td>
</tr>
<tr>
<td>---</td>
<td>-------------</td>
<td>--------</td>
<td>---------------</td>
<td>-------------------------</td>
<td>-------</td>
<td>------------</td>
</tr>
<tr>
<td>1</td>
<td>Helium</td>
<td>He</td>
<td>2</td>
<td>24.57</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>2</td>
<td>Lithium</td>
<td>Li</td>
<td>3</td>
<td>5.38</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>3</td>
<td>Beryllium</td>
<td>Be</td>
<td>4</td>
<td>9.01</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>4</td>
<td>Boron</td>
<td>B</td>
<td>5</td>
<td>10.81</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>5</td>
<td>Carbon</td>
<td>C</td>
<td>6</td>
<td>12.01</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>6</td>
<td>Nitrogen</td>
<td>N</td>
<td>7</td>
<td>14.01</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>7</td>
<td>Oxygen</td>
<td>O</td>
<td>8</td>
<td>15.99</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>8</td>
<td>Fluorine</td>
<td>F</td>
<td>9</td>
<td>18.99</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>9</td>
<td>Neon</td>
<td>Ne</td>
<td>10</td>
<td>20.18</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>10</td>
<td>Sodium</td>
<td>Na</td>
<td>11</td>
<td>22.99</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>11</td>
<td>Magnesium</td>
<td>Mg</td>
<td>12</td>
<td>24.31</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>12</td>
<td>Aluminum</td>
<td>Al</td>
<td>13</td>
<td>26.98</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>13</td>
<td>Silicon</td>
<td>Si</td>
<td>14</td>
<td>28.09</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>14</td>
<td>Phosphorus</td>
<td>P</td>
<td>15</td>
<td>30.97</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>15</td>
<td>Sulfur</td>
<td>S</td>
<td>16</td>
<td>32.06</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>16</td>
<td>Chlorine</td>
<td>Cl</td>
<td>17</td>
<td>35.45</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>17</td>
<td>Argon</td>
<td>Ar</td>
<td>18</td>
<td>39.95</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>18</td>
<td>Potassium</td>
<td>K</td>
<td>19</td>
<td>39.09</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>19</td>
<td>Calcium</td>
<td>Ca</td>
<td>20</td>
<td>40.08</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>20</td>
<td>Scandium</td>
<td>Sc</td>
<td>21</td>
<td>44.96</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>21</td>
<td>Titanium</td>
<td>Ti</td>
<td>22</td>
<td>47.87</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>22</td>
<td>Vanadium</td>
<td>V</td>
<td>23</td>
<td>50.94</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>23</td>
<td>Chromium</td>
<td>Cr</td>
<td>24</td>
<td>51.99</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>24</td>
<td>Manganese</td>
<td>Mn</td>
<td>25</td>
<td>54.94</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>25</td>
<td>Iron</td>
<td>Fe</td>
<td>26</td>
<td>55.85</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>26</td>
<td>Cobalt</td>
<td>Co</td>
<td>27</td>
<td>58.93</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>27</td>
<td>Nickel</td>
<td>Ni</td>
<td>28</td>
<td>58.69</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>28</td>
<td>Copper</td>
<td>Cu</td>
<td>29</td>
<td>63.54</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>29</td>
<td>Zinc</td>
<td>Zn</td>
<td>30</td>
<td>65.39</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>30</td>
<td>Gallium</td>
<td>Ga</td>
<td>31</td>
<td>69.72</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>31</td>
<td>Germanium</td>
<td>Ge</td>
<td>32</td>
<td>72.64</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>32</td>
<td>Arsenic</td>
<td>As</td>
<td>33</td>
<td>74.92</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>33</td>
<td>Selenium</td>
<td>Se</td>
<td>34</td>
<td>78.96</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>34</td>
<td>Bromine</td>
<td>Br</td>
<td>35</td>
<td>79.90</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>35</td>
<td>Krypton</td>
<td>Kr</td>
<td>36</td>
<td>83.80</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>36</td>
<td>Rubidium</td>
<td>Rb</td>
<td>37</td>
<td>85.47</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>37</td>
<td>Strontium</td>
<td>Sr</td>
<td>38</td>
<td>87.62</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>38</td>
<td>Yttrium</td>
<td>Y</td>
<td>39</td>
<td>88.91</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>39</td>
<td>Zirconium</td>
<td>Zr</td>
<td>40</td>
<td>91.22</td>
<td>S/L</td>
<td>Y</td>
</tr>
</tbody>
</table>
Table G.4 (continued).

<table>
<thead>
<tr>
<th>Z</th>
<th>Element</th>
<th>Symbol</th>
<th>Atomic Weight</th>
<th>Density (g/cm³)</th>
<th>State</th>
<th>Conductor</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>Niobium</td>
<td>Nb</td>
<td>92.90638</td>
<td>8.57</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>42</td>
<td>Molybdenum</td>
<td>Mo</td>
<td>95.94</td>
<td>10.22</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>43</td>
<td>Technetium</td>
<td>Tc</td>
<td>98.0</td>
<td>11.5</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>44</td>
<td>Ruthenium</td>
<td>Ru</td>
<td>101.07</td>
<td>12.41</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>45</td>
<td>Rhodium</td>
<td>Rh</td>
<td>102.9055</td>
<td>12.41</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>46</td>
<td>Palladium</td>
<td>Pd</td>
<td>106.42</td>
<td>12.02</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>47</td>
<td>Silver</td>
<td>Ag</td>
<td>107.8682</td>
<td>10.5</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>48</td>
<td>Cadmium</td>
<td>Cd</td>
<td>112.411</td>
<td>8.65</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>49</td>
<td>Indium</td>
<td>In</td>
<td>114.818</td>
<td>7.31</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>50</td>
<td>Tin</td>
<td>Sn</td>
<td>118.71</td>
<td>7.31</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>51</td>
<td>Antimony</td>
<td>Sb</td>
<td>121.76</td>
<td>6.691</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>52</td>
<td>Tellurium</td>
<td>Te</td>
<td>127.60</td>
<td>6.24</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>53</td>
<td>Iodine</td>
<td>I</td>
<td>126.90447</td>
<td>4.93</td>
<td>S/L</td>
<td>N</td>
</tr>
<tr>
<td>54</td>
<td>Xenon</td>
<td>Xe</td>
<td>131.29</td>
<td>5.485E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>55</td>
<td>Cesium</td>
<td>Cs</td>
<td>132.90545</td>
<td>1.873</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>56</td>
<td>Barium</td>
<td>Ba</td>
<td>137.327</td>
<td>3.50</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>57</td>
<td>Lanthanum</td>
<td>La</td>
<td>138.9055</td>
<td>6.154</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>58</td>
<td>Cerium</td>
<td>Ce</td>
<td>140.116</td>
<td>6.657</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>59</td>
<td>Praeseodymium</td>
<td>Pr</td>
<td>140.90765</td>
<td>6.71</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>60</td>
<td>Neodymium</td>
<td>Nd</td>
<td>144.24</td>
<td>6.90</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>61</td>
<td>Promethium</td>
<td>Pm</td>
<td>145.0</td>
<td>7.22</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>62</td>
<td>Samarium</td>
<td>Sm</td>
<td>150.36</td>
<td>7.46</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>63</td>
<td>Europium</td>
<td>Eu</td>
<td>151.964</td>
<td>5.243</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>Element</td>
<td>Symbol</td>
<td>Atomic Mass</td>
<td>Electronegativity</td>
<td>Solubility</td>
<td>Reactivity</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
<td>-------------</td>
<td>-------------------</td>
<td>------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>Gadolinium</td>
<td>Gd</td>
<td>157.25</td>
<td>7.90</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Terbium</td>
<td>Tb</td>
<td>158.92534</td>
<td>8.229</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Dysprosium</td>
<td>Dy</td>
<td>162.50</td>
<td>8.55</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Holmium</td>
<td>Ho</td>
<td>164.93032</td>
<td>8.795</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Erbium</td>
<td>Er</td>
<td>167.26</td>
<td>9.066</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Thulium</td>
<td>Tm</td>
<td>168.9342</td>
<td>9.321</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Ytterbium</td>
<td>Yb</td>
<td>173.04</td>
<td>6.73</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Lutetium</td>
<td>Lu</td>
<td>174.967</td>
<td>9.84</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Hafnium</td>
<td>Hf</td>
<td>178.49</td>
<td>13.31</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Tantalum</td>
<td>Ta</td>
<td>180.9479</td>
<td>16.65</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Tungsten</td>
<td>W</td>
<td>183.84</td>
<td>19.3</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Rhenium</td>
<td>Re</td>
<td>186.207</td>
<td>21.02</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Osmium</td>
<td>Os</td>
<td>190.23</td>
<td>22.57</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Iridium</td>
<td>Ir</td>
<td>192.217</td>
<td>22.42</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Platinum</td>
<td>Pt</td>
<td>195.078</td>
<td>21.02</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Gold</td>
<td>Au</td>
<td>196.96655</td>
<td>19.32</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Mercury</td>
<td>Hg</td>
<td>200.59</td>
<td>13.55</td>
<td>S/L</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Z</td>
<td>Element</td>
<td>Symbol</td>
<td>Atomic Weight</td>
<td>Density (g/cm³)</td>
<td>State</td>
<td>Conductor</td>
</tr>
<tr>
<td>----</td>
<td>-------------</td>
<td>--------</td>
<td>---------------</td>
<td>-----------------</td>
<td>-------</td>
<td>-----------</td>
</tr>
<tr>
<td>81</td>
<td>Thallium</td>
<td>Tl</td>
<td>204.3833</td>
<td>11.72</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>82</td>
<td>Lead</td>
<td>Pb</td>
<td>207.2</td>
<td>11.35</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>83</td>
<td>Bismuth</td>
<td>Bi</td>
<td>208.98038</td>
<td>9.747</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>84</td>
<td>Polonium</td>
<td>Po</td>
<td>209.0</td>
<td>9.32</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>85</td>
<td>Astatine</td>
<td>At</td>
<td>210.0</td>
<td>No default</td>
<td>S/L</td>
<td>N</td>
</tr>
<tr>
<td>86</td>
<td>Radon</td>
<td>Rn</td>
<td>222.0</td>
<td>9.066E-3</td>
<td>Gas</td>
<td>N</td>
</tr>
<tr>
<td>87</td>
<td>Francium</td>
<td>Fr</td>
<td>223.0</td>
<td>No default</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>88</td>
<td>Radium</td>
<td>Ra</td>
<td>226.0</td>
<td>5.00</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>89</td>
<td>Actinium</td>
<td>Ac</td>
<td>227.0</td>
<td>10.07</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>90</td>
<td>Thorium</td>
<td>Th</td>
<td>232.0381</td>
<td>11.72</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>91</td>
<td>Protactinium</td>
<td>Pa</td>
<td>231.03588</td>
<td>15.37</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>92</td>
<td>Uranium</td>
<td>U</td>
<td>238.0289</td>
<td>18.95</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>93</td>
<td>Neptunium</td>
<td>Np</td>
<td>237.0381</td>
<td>20.45</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>94</td>
<td>Plutonium</td>
<td>Pu</td>
<td>244.0</td>
<td>19.82</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>95</td>
<td>Americium</td>
<td>Am</td>
<td>243.0</td>
<td>13.671</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>96</td>
<td>Curium</td>
<td>Cm</td>
<td>247.0</td>
<td>13.51</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>97</td>
<td>Berkelium</td>
<td>Bk</td>
<td>247.0</td>
<td>14.78</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>98</td>
<td>Californium</td>
<td>Cf</td>
<td>251.0</td>
<td>No default</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>99</td>
<td>Einsteinium</td>
<td>Es</td>
<td>252.0</td>
<td>No default</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>100</td>
<td>Fermium</td>
<td>Fm</td>
<td>257.0</td>
<td>No default</td>
<td>S/L</td>
<td>Y</td>
</tr>
<tr>
<td>Material</td>
<td>Keyword</td>
<td>Composition</td>
<td>Density</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------</td>
<td>--------------------------------</td>
<td>----------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alumina</td>
<td>AL203</td>
<td>O.4708 AL .5292</td>
<td>3.99</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aluminum 6061</td>
<td>AL6061</td>
<td>MG .0080 AL .9725 SI .0040 TI .0015 CR .0015 MN .0015 FE .0070 CU .0015 ZN .0025</td>
<td>2.69</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aluminum 7075</td>
<td>AL7075</td>
<td>MG .0250 AL .9000 CR .0030 SI .0016 ZN .0060</td>
<td>2.80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bakelite Phenolic</td>
<td>BAKELI</td>
<td>H.0645 C.7656 O.1699</td>
<td>1.45</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Brass</td>
<td>BRASS</td>
<td>FE .0020 CU .6150 ZN .3520 PB .0310</td>
<td>8.40</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bronze</td>
<td>BRONZE</td>
<td>P.0030 CU .9470 SN .0050</td>
<td>8.86</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CaF TLD</td>
<td>CAFTLD</td>
<td>F.4668 CA .5132 MN .0198</td>
<td>3.18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kapton</td>
<td>KAPTON</td>
<td>H.0100 C.5600 N.1300 O.3000</td>
<td>1.42</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kevlar</td>
<td>KEVLAR</td>
<td>H.0428 C.6958 N.1166 O.1448</td>
<td>1.31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kennertium</td>
<td>KENNER</td>
<td>CU .0200 W.9800</td>
<td>18.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kovar</td>
<td>KOVAR</td>
<td>MG .0030 FE .5370 CO .1700 NI .2900</td>
<td>7.90</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lithium Fluoride</td>
<td>LIF</td>
<td>LI .2675 F.7325</td>
<td>2.635</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mica</td>
<td>MICA</td>
<td>H.0051 O.4820 AL .2032 SI .2115 K.0982</td>
<td>2.80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mylar</td>
<td>MYLAR</td>
<td>H.0519 C.6186 O.3295</td>
<td>1.38</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nylon</td>
<td>NYLON</td>
<td>H.0980 C.6368 N.1238 O.1414</td>
<td>1.13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phenolic Linen</td>
<td>PHENOL</td>
<td>H.0800 C.7100 N.0600 O.1500</td>
<td>1.33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polyethylene</td>
<td>POLYE</td>
<td>H.1437 C.8563</td>
<td>0.92</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polyimide PCB</td>
<td>POLYIM</td>
<td>H.0150 B.0120 C.2800 N.0350 O.3370 NA .0030 MG .0020 AL .0590 SI .1460 CA .1110</td>
<td>1.85</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polyurethane</td>
<td>POLYU</td>
<td>H.0402 C.5913 N.1398 O.2287</td>
<td>0.192</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Silicone Dioxide</td>
<td>SIO2</td>
<td>O.5326 SI .4674</td>
<td>2.20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Silica Glass</td>
<td>GLASS</td>
<td>O.5257 MG .0006 AL .0048 SI .4475 K.0008 CA .0164 FE .0042</td>
<td>2.18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solder</td>
<td>SOLDIER</td>
<td>SN .6000 PB .4000</td>
<td>8.67</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stainless Steel 304</td>
<td>SS304</td>
<td>C.0008 SI .0100 P.0005 S.0003 CR .1900 MN .0200 FE .6784 NI .1000</td>
<td>8.03</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stainless Steel 410</td>
<td>SS410</td>
<td>C.0015 SI .0100 P.0004 S.0003 CR .1200 MN .0100 FE .8578</td>
<td>7.76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTV630</td>
<td>RTV630</td>
<td>H.0490 C.2100 O.3250 AL .0010 SI .4130 CU .0020</td>
<td>1.28</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Teflon</td>
<td>TEFLON</td>
<td>C.2401 F.7599</td>
<td>2.15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Water</td>
<td>WATER</td>
<td>H.1100 O.8900</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Glossary of Terms

H Glossary of Terms

ACCEPT: the three dimensional geometry capability of the ITS codes.

ACCEPTM: Historically, the MCODES version of ACCEPT.

ACCEPTP: the PCODES version of ACCEPT. ACIS: a particular format of B-rep geometry developed by SpatialTechnology, Inc. Adjoint: the solution of the adjoint transport problem. This simulation method is similar to the physical processes going backwards in time. Particles begin at a radiation detector, and tallies are made at radiation source locations. Biasing: a distortion of natural analog processes to achieve variance reduction in certain desired output quantities.
**Body**: one of the geometric entities used to construct a zone. In CG, these are geometric primitives such as spheres, boxes, etc. In CAD, a body is not a geometric primitive, and a body and a zone refer to the same entity. **B-rep**: the boundary representation geometry description typically employed by CAD packages. **CAD**: computer aided design, also used to refer to the boundary representation (B-rep) geometries created by CAD. **CEPXS**: the Coupled Electron-Photon X-Section generation code used to create multi-group cross sections for ITS. **CG**: abbreviation for Combinatorial Geometry. The method of combining simple geometric entities, such as tori and spheres, into more complicated geometry descriptions. This method is also know as Constructive Solid Geometry (or CSG). **Code Options**: the set of valid preprocessor definitions used to select a version of the ITS codes to be compiled. **Code Zone**: unions in the CG description of an input zone are maintained internally as separate code zones for particle tracking purposes.
**Collision Forcing**: a biasing technique used for photons. Photons entering a zone are forced to interact with a specified probability, which may be larger or smaller than the natural interaction probability. The weight of the photon is modified accordingly.

**Combinatorial Geometry**: see CG. **Constructive Solid Geometry**: see CG. 

**CG**: see CG. **CSG**: see CG.

**CUI**: the Combined User Inputs employed when running ITS with shell scripts.

---

**CutoffEnergy**: the energy below which particles are not simulated in detail. The cutoff energies may be different for electrons and photons.

**CYLTRAN**: the axisymmetric cylindrical material geometry ITS codes, with fully three dimensional descriptions of particle trajectories.

**CYLTRANM**: Historically, the MCODES version of CYLTRAN.

**CYLTRANP**: the PCODES version of CYLTRAN.
**Dose**: energy absorbed by a material per unit mass, often used interchangeably with the term "energy deposition", which is not normalized per unit mass.

**ElectronTrapping**: see Trapping.

**ETRAN**: the one dimensional, single material electron/photon transport code developed by the National Institute of Standards and Technology from which the ITS codes were developed.

**Forward**: the solution of the forward transport problem. This simulation method is similar to the physical processes going forward in time. Particles begin at a radiation source, and tallies are made at radiation detector locations.

**Group**: a span of the particle energy domain within the multigroup approximation over which particles are assumed to interact with the same probabilities.

**HYBRID Geometry**: the geometry may be described by a combination of CAD and CG zones.

**Input Body**: see Body.

**Input Zone**: see Zone.

**ITS**: the Integrated TIGER Series codes, sometimes referring to only the continuous-energy versions of the codes and not the MITS codes.

**ITS-CAD**: the ITS and MITS codes capable of tracking particles in three dimensional CAD geometries.

**KERMA**: Kinetic Energy Released to Material is the energy deposition calculated in the absence of electron transport with the assumption of electronic equilibrium. Various assumptions can be made regarding the radiative energy that would be produced by electrons. CEPXS assumes that radiative energy is not lost from the system and is locally deposited.

**Keywords**: the set of words that may be used in the input deck that the code keys upon to activate and deactivate code features.

**MCODES**: The ITS codes that were historically used for simulations involving electric and magnetic fields. Fields are now integrated into the standard codes and activated with the EBFIELDS keyword.

**MITS**: the Multigroup Integrated TIGER Series codes.
**Multigroup**: an approximation involving the discretization of the particle energy domain into groups. The approximation is only accurate if the cross sections do not vary significantly over each group.

**Next Event Estimator**: a biasing technique used for photons. The probability that a photon will escape without interacting is used to record an escape tally, with the weight of the tally modified accordingly, regardless of whether the photon actually escapes in the particular simulation.

**Overlay**: a tally structure superimposed upon a zone for calculating the subzone distribution of energy and charge deposition and/or flux.

**PCODES**: the more elaborate ionization/relaxation model adapted from the SANDYL code.

**Preprocessor**: the compiler preprocessor used to apply the selection of the code version and include the necessary common block statements into the source code. The cpp program is used for preprocessing ITS.

**Prmfile**: the file containing CAD parameter input.

**RNG**: random number generator.

**Russian Roulette**: a biasing technique in which particles may be eliminated with some probability. If the particle is not eliminated from the simulation, the weight of the particle is modified accordingly.

**Satfile**: the file containing the CAD geometry in ACIS text format.

**Scaling**: a biasing technique based on scaling material cross sections (larger or smaller) and correspondingly modifying the weighting of any particle exposed to the scaled cross sections.

**Sub-keyword**: a secondary keyword that will not be recognized without the presence of the associated primary keyword.
Subzone: a geometry entity, usually a small fraction of a zone, in which energy and charge deposition and/or flux are tallied.

TIGER: the one-dimensional geometry capability of the ITS codes.

TIGERP: the PCODES version of TIGER.

Trapping: a biasing technique used for electrons. Because electrons have a finite range within material, it may not be possible for an electron to escape the zone or subzone it is in. If it cannot escape, it is “trapped.” Trapping may neglect effects of secondary photons that can escape from the zone.

XGEN: the coupled electron-photon X-section GENERation code used to create continuous-energy cross sections for ITS.

Zone: a geometry entity that consists of a single material and density.

References


Index

A1GRID, 76 A2GRID, 76 ACCEPT, 11, 34 ACCEPTM, 11 ACCEPTP, 12
ACIS Modeler, 89, 116 ACIS12, 34 ACIS BEGIN, 89 ACIS END, 89
ADJOINT, 41, 154, 155 adjoint, 13, 154, 155 ADJOINT-SPECTRA, 83
annihilation, 124, 125, 128, 133, 135 ANNIHILATION-LINE, 181
ANNIHILATION-LINE-TALLY-..., 42 ANNULUS, 75 AREAL-DENSITY-FILE,
63 AREAL-DENSITY-OUTPUT, 42 Auger, 11, 127, 140 AZ, 57

BATCHES, 42, 119, 124, 147 BELOW-1KEV, 42, 167 BIAS-GLOBAL, 18
BIAS-SHELL, 43 BIAS-ZONE, 18 BIASING, 43, 120, 143 biasing, 120, 124,
143 BINT, 51 BODY, 56, 62, 77, 82 body, see GEOMETRY, body BREAK-
INDICATOR, 73

CAD, 13, 34, 86, 87, 137 CAD-VOLUMES, 114 CAD ONLY, 88, 116–118
CEPXs, 16, 19, 123, 174, 178, 180 CEPXS-INFO-ONLY, 181 CG ONLY, 87,
88, 116, 117 CHARGE, 49, 65, 68, 69, 136 CHOLLA, 34 Cholla, 27, 29
CHOLLA BEGIN, 90 CHOLLA END, 90 CODEZONES-PER-PATH, 47
COLLISION-FORCING, 44, 144 comments, 41, 167, 180 configure, 19, 24,
25, 161, 174 CONNECTIVITY-DUMP-FILE, 63 CONNECTIVITY-READ-FILE,
63 CONTRAST, 181 COSINE-LAW, 52, 61 cross sections, 122 CSDA, 181
CUI, 27, 28, 162, 175 CUSTOM-RR, 45 CUSTOM-TRAP, 47 CUTOFF, 182
CUTOFF-PHOTONS-ESCAPE, 48, 131 CUTOFFS, 48, 119, 130, 143 CVS,
21, 30, 31, 163, 164, 176 CYLTRAN, 11, 34 CYLTRANM, 11 CYLTRANP, 11
defined void, 116 DELTA0-AVE, 61, 71 DEPOSITION-
UNITS, 41, 49, 132 DETAIL-IONIZE, 18 DETECTOR-
RESPONSE, 49, 124, 136 detour, 122, 123 DIRECTION,
51 DIRECTION-SPACE, 59, 61, 71 DISK, 74
documentation manual, 14, 15 other, 15 DOPPLER, 52,
167 DOSE, 50, 65, 68, 69, 136 DUMP, 53, 126 DUMP-
CONNECTIVITY, 53 DUMP-FILE, 64 DYNAMIC-MPI, 53,
125

EBFIELDS, 17, 53, 141, 142 ECHO, 41, 56, 167 EGROUP, 182 ELASTIC-
LEGENDRE, 182 ELECTRAN, 44, 48 electric fields, see EBFIELDS

203

ELECTRON-ANGLE-BINS, 168 ELECTRON-EMISSION, 56, 124, 134
ELECTRON-ESCAPE, 58, 124, 135 ELECTRON-FLUX, 60, 124, 133
ELECTRON-GRID-LENGTH, 168 ELECTRON-RR, 44, 144 ELECTRON-
SOURCE, 183 ELECTRON-SURFACE-SOURCE, 60, 81, 124 ELECTRON-
VOLUME-SOURCE, 61, 124 ELECTRONS, 56, 183 ENERGY, 62, 168, 184
error, 121, 124, 125 ESCAPE, 50

ELECTRONS, 50, 136
PHOTONS, 50, 136 escape zone, 112, 116, 117, 135, 136 ESCAPE-
SURFACES, 41, 62, 91, 95, 112, 130,
135 ETRAN, 11 EZTRAN, 11
EZTRAN2, 11
facet, 27, 29 facet geometry, 90 fields, 141, 142 FILE-NAMES, 41, 63 FINITE-ELEMENT-FILE, 64 FINITE-ELEMENT-FORMAT, 65, 68, 70, 114 fluorescence, 11, 123, 125, 127, 128, 140 Fokker-Planck, 124, 129, 192 FULL-COUPLING, 184, 191

GEOMETRY, 41, 66 ACCEPT, 96, 113 body, 96, 101 CAD, 116 CYLTRAN, 93, 95, 124 facet, 90 material, 112, 113, 115, 122–124 subzoning, see subzoning TIGER, 91, 124 volume, 111, 112, 114, 121, 133, 150, 151 negative, 121 zone, 102, 104 GROUP, 50, 62

HISTORIES, 66, 119, 124 HISTORIES-PER-BATCH, 66, 119, 124 HYBRID, 88, 116–118 INCLUDE-FILE, 66, 168, 184 INCOH-BINDING, 184 input body, see GEOMETRY, body input zone, see GEOMETRY, zone INPUT FILE, 87 installation, see testing, installation INTERMEDIATE-FILE, 64 ISOTROPIC, 52, 61 ITS, 16, 19 ITS2P1, 185 ITS BEGIN, 87 ITS END, 87, 89

KD TREE PARAMS, 88 KERMA, 51, 136 kicking, 127 KNOCKONS-WITH-PRIMARIES, 185

LEGENDRE, 185 LINE, 74 line radiation, 134, 135, 185, 187 LINE-TALLY-WITH-CONTINUUM, 67 LINES, 185 LIST, 65 LIST-ZONE, 65 LOCATION, 50, 51

magnetic fields, see EBFIELDS MAGNETIC-TRAP-LIMITS, 67 Makefile, 19, 24, 25, 34, 161, 174 MASS, 49 MATERIAL, 50, 51, 167, 169, 180, 185

CONDUCTOR, 169, 186 DENSITY, 169, 186 DENSITY-RATIO, 170

GAS, 169, 186 NO-SEC-ELEC, 187 NON-CONDUCTOR, 169, 186

SUBSTEP, 170

material, see GEOMETRY, material MATNAM, 180, 187 MCODES, see EBFIELDS MersenneTwister, 34, 152 MICRO, 67, 131 MIRROR CG, 88, 116, 117 MITS, 13, 34, 180 MODE, 88 MONO-PHOTON, 180, 187 MORSE, 11

MPI, 34 dynamic, 125, 126 static, 125, 126

multigroup, 13

NBINE, 59, 60, 78, 84 NBINP, 59, 61 NBINT, 59, 61 ncCVS, 175 NEW-DATA-SET, 41, 67, 126 NEXT-EVENT-ESCAPE, 45, 129, 144 nm6CVS, 27 NO-BANK, 45 NO-COHERENT, 48, 68, 128, 187 NO-COUPLING, 183, 190 NO-DEPOSITION, 68, 131 NO-DEPOSITION-OUTPUT, 68, 70, 131 NO-DETAILED-DEPOSITION, 68, 131 NO-GEOMETRY-TABLE, 69, 124 NO-INCOH-BINDING, 69, 188 NO-INTERMEDIATE-OUTPUT, 69 NO-KICKING, 69, 131 NO-KNOCKONS, 70, 127 NO-LINES, 181, 188 NO-OVERLAP-OUTPUT, 70 NO-PCODE, 188 NO-POSITRONS, 188 NO-PAIR, 189 PEQE, 188 NO-SEC-ELEC-GLOBAL, 189 NO-SIGMA, 65 NO-STRAGGLING, 70, 127 NO-SZDEPOSITION-OUTPUT, 70, 131 non-conformal, see subzoning, overlays NORMAL, 76 NUMBER-PER-BIN, 82, 83

NUMBER-PER-BIN-PER-MEV, 82, 83 ncCVS, 162
ORBITS, 73 OUTPUT FILE, 87 OUTWARD, 77 OVERLAP-OUTPUT-MAX, 70
overlay, see subzoning, overlays

PAGE-HEADER, 73 parallel, 34, 53, 84, 119, 125 PARTIAL-COUPLING, 184,
190 PCODES, 34, 130, 133, 135, 140, 162
PFF, 34 PFF-FILE, 64 PFF-FORMAT, 71 PGROUP,
189 PHOTON-ESCAPE, 71, 124, 129, 135, 144
PHOTON-FLUX, 71, 124, 133 PHOTON-SOURCE,
190 PHOTON-SURFACE-SOURCE, 71, 81, 124,
129 PHOTON-VOLUME-SOURCE, 72, 124
PHOTONS, 71, 189 PHOTRAN, 45, 145 PLOT-3DAXIS, 73 PLOT-FILE, 64 PLOTS, 72 PLOTS-
OVAL-RESOLUTION, 72 POINT, 50, 51, 74
POSITION, 74 positrons, 122, 124, 125, 135, 188
postproc, 27, 31, 162, 175 preprocess definitions,
34, 121, 165 PRINT, 191 PRINT-ALL, 78, 126, 170,
191 prmfle, 27, 29, 121 processors, see TASKS
PROFILE, 75 PULSE-HEIGHT, 78, 124, 134

RADIAL-BIASING, 75 RADIUS, 75 RANDOM-NUMBER, 79, 126 RANMAR,
34, 152 RAYPRINT, 71 RAYTRACE, 71 READ-CONNECTIVITY, 79
RECTANGLE, 75 REFLECTION-ZONE, 80 RESTART, 80, 126 RESTART-
FILE, 65 RESTART-HISTORY, 79, 81 REVERSE, 76 RMAX, 62, 76, 81
RNG1, 34, 79, 152, 153 RNG2, 34, 79, 152, 153 RNG3, 34, 79, 152, 153
Russian Roulette, see ELECTRON-RR

SANDYL, 12, 140 satfile, 27, 29 SCALE, 49 SCALE-BREMS, 46, 144, 145
SCALE-EP, 46, 144, 145 SCALE-IMPACT, 46, 145 SCALE-PE, 47, 146
scaling, 125 sendn, 27, 30, 162, 163, 175 SHELL, 78 sidestep, 137, 138
SIMPLE-BREMS, 81 SOURCE-SURFACES, 41, 81, 91, 95, 112 Spatial
Corporation, 89, 116 SPECTRUM, 82, 83, 130 SPHEM, 11 SPHERE, 11
statistical uncertainty, 126, 147 STEP, 170 steps, 127, 170 substeps, 122–
124, 149, 170 subsurfacing, 57 SUBZONE-ONLY, 41, 114 subzoning, 105,
107, 109, 110, 148–150
CYLTRAN, 93 non-conformal, see subzoning, overlays overlays, 110,
133, 150, 151 TIGER, 91
SURFACE, 56, 62, 76, 81 SURFACE-SOURCE, 84
SURFACES, 56

TASKS, 84, 119, 125
testing, 156, 159 installation, 19, 20, 156, 159, 160 regression
CEPXS, 174
XGEN, 161 TIGER, 11, 34 TIGERP, 11 timing, 125, 126, 138, 139
TITLE, 41, 84, 167, 170, 180, 192 TRAJECTORY-POINTS, 85 TRAP-
eLECTRONS, 47, 146 trapping, 120, 121

uncertainty, see statistical uncertainty undefined void, 116 UNIFORM-
ISOTROPIC-FLUX, 77 USCAT, 192 USCAT-ITS, 192
variancereduction, see biasing VOID, 114 VOLUME, 49–51, 78
warning, 121, 124
XGEN, 16, 19, 122, 161, 165–167 XSECTION-FILE, 65
ZMAX, 62, 76, 81 ZMIN, 62, 76, 81 ZONE, 78, 126 zone, see GEOMETRY, zone

**DISTRIBUTION:**

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Code</th>
<th>Name</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MS 1179</td>
<td>M. J. Crawford</td>
<td>01341</td>
</tr>
<tr>
<td>1</td>
<td>MS 1179</td>
<td>B. C. Franke</td>
<td>01341</td>
</tr>
<tr>
<td>1</td>
<td>MS 1179</td>
<td>R.P. Kensek</td>
<td>01341</td>
</tr>
<tr>
<td>1</td>
<td>MS 1179</td>
<td>T.W. Laub</td>
<td>01341</td>
</tr>
<tr>
<td>1</td>
<td>MS 1179</td>
<td>L. J. Lorence</td>
<td>01341</td>
</tr>
<tr>
<td>1</td>
<td>MS 0899</td>
<td>Technical Library</td>
<td>04536</td>
</tr>
</tbody>
</table>