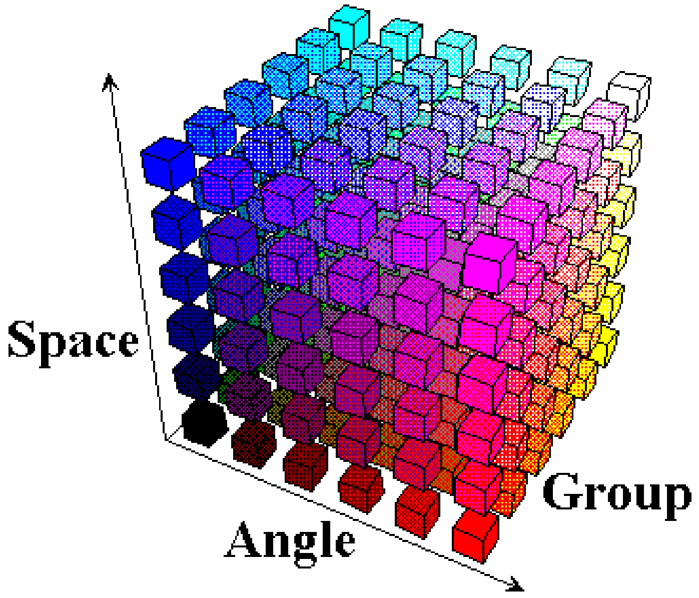
PENTRAN™

Parallel Environment Neutral-Particle TRANsport
(A Parallel 3-D Discrete Ordinates (Sn) Code)

Users Guide to Version 9.00



G. Sjoden and A. Haghighat
with contributions by V. Kucukboyaci

April 2000

H&S Advanced Computing Technologies Incorporated

Contents

Version 9.00 Release Notes

This version of *PENTRAN* includes several improvements to the code architecture. The changes include the addition of a restart capability, dynamically allocated arrays in F90, and fine-tuning of code parameters. Because the code runs in ANSI FORTRAN-90, there are no patches or code settings to be made; the same source code compiles and runs on any machine, and has been tested on *IBM*, *SUN*, and *SGI* parallel computers. Parallel operation on PCs under the *Linux* Operating System was also successful using the efficient VAST Linux F90 compiler.

LEGAL NOTICE: *PENTRAN* is marketed and limited-licensed by H&S Advanced Computing Technologies, Inc. (H&SACT), found on the web at http://www.hsact.com. Any use, application, or reference to any portion of the *PENTRAN* code system, User's Guide, magnetic media, or any other materials linked to H&SACT constitutes a full, implicit release of the code author(s) and collective research underwriters/sponsors from all liability. IN NO EVENT SHALL H&SACT BE LIABLE FOR ANY INDIRECT, INCIDENTAL, PUNITIVE, SPECIAL OR CONSEQUENTIAL DAMAGES, OR DAMAGES FOR LOSS OF PROFITS, REVENUE, OR USE INCURRED BY THE USER OR ANY THIRD PARTY, WHETHER IN AN ACTION IN CONTRACT, OR TORT, OR OTHERWISE EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. H&SACT'S LIABILITY FOR DAMAGES ARISING OUT OF OR IN CONNECTION WITH THIS SOFTWARE SHALL BE ZERO DOLLARS. THE PROVISIONS OF THIS AGREEMENT ALLOCATE THE RISKS BETWEEN H&SACT AND THE USER. H&SACT'S PRICING REFLECTS THIS ALLOCATION OF RISK AND BUT FOR THIS ALLOCATION AND LIMITATION OF LIABILITY, COMPANY WOULD NOT HAVE RELEASED THIS AGREEMENT.

DISTRIBUTION: Any user of the *PENTRAN* code, User's Guide, magnetic media, or any other materials linked to the *PENTRAN* Development Project implicitly agrees that the materials are not releaseable to other parties/individuals in accordance with a limited contract license agreement. Violators of the limited license agreement are subject to civil and criminal sanctions and will be prosecuted to include damages and legal expenses as awarded in accordance to the Confidentiality Laws of the State of Pennsylvania. The user agrees to take all reasonable steps to ensure that Confidential Information is not disclosed or distributed by its employees, representatives or agents in violation of the terms of this Agreement.

LICENSING: Parties interested in obtaining a licensed version of *PENTRAN* should contact Dr. Alireza Haghighat at H&S Advanced Computing Technologies, Inc, 430 Canterbury Dr, State College, Pennsylvania 16803, http:://www.hsact.com, telephone: (814) 231-8196, email: ali@hsact.com

About the Cover: A visualization of a virtual processor array with decomposition of the transport equation used in *PENTRAN* in the angular, energy, and spatial (*x y z* Cartesian) domains.

Foreword

The overall vision of solving bigger radiation transport problems better and faster than ever before was the fundamental reason for the development of the *PENTRAN* code. Continuing development and applicatiuon of *PENTRAN* has been a remarkable experience, and has provided great insight about discrete ordinates transport theory and parallel processing, although there is so much more to learn. Microsoft® Corporation's capable *FORTRAN PowerStation/Developer Studio 4.0* for the PC was used to develop the *PENTRAN* code (now *Compaq Visual FORTRAN*); these are superb programming and debugging vehicles, and allowed for easy portability of the code to most any platform, with strict enforcement of ANSI-FORTRAN. This User's Guide is intended to describe the features, input parameters, and general operating characteristics of *PENTRAN*. Undoubtedly, additions and amendments will be made to this document, as the code is under continuous development and testing.

Acknowledgments

For parallel supercomputing support, the authors thank the Cornell Theory Center for their access to the 512 node IBM -SP2. In particular, John Zollweg of the Theory Center's Technical Group provided timely advice on parallel data handling on the SP2. We also thank Kevin Morooney and Jeff Nucciarone of the Penn State Numerically Intensive Computing Group for technical suuport and access to the Penn State IBM-SP2.

Several individuals in the Penn State Transport Theory Group contributed to this effort in different capacities.

- Mr Vefa Kucukboyaci has been a major contributor to the code. He contributed significantly to the group window option, and on his own developed and integrated the Restart Capability for large multi-group problems. He also applied *PENTRAN* to real problems that identified key refinements needed to make the initial code release completely tractable for large problems, and greatly assisted in code benchmarking.
- Dr Melissa Hunter provided valuable experience and many crucial and timely *TWOTRAN-II* and *THREEDANT* solutions for benchmarking.
- Dr Bojan Petrovic provided *DORT/TORT* benchmarks and valuable technical advice on cross section formats.
- Dr John Wagner provided very useful advice on *IBM RISC-6000* and *SP2* operations.
- Mr Willie Stevenson provided constructive feedback and support.

The first author wishes to thank the following: I thank the US Air Force for the opportunity to return to college to pursue the PhD. Also, without the incredible patience demonstrated by my wife Patti, and 3 children, nothing at all would ever get done. Finally, and not the least, I thank God for the wisdom to develop this code. -*Glenn E. Sjoden, PhD, PE*

The second author wishes to thank the following: I thank my wife Mastaneh and my son Aarash for their faithful patience and support throughout my work. -*Alireza Haghighat, PhD*

I. Technical Notes

1.1 Goals and Objectives

The *PENTRAN (Parallel Environment Neutral-particle TRANsport)* code was initially developed with the following research goals and code development objectives in mind:

Research Goals: (1) To demonstrate the parallel scalability of a 3-D transport code designed from scratch for scalable parallel implementation with full variable decomposition on distributed memory and computing architectures, (2) To implement and test parallel algorithm phase space decomposition strategies based on problem physics and load balancing/parallel efficiency issues, (3) To provide for and investigate the benefits of adaptive discrete ordinates spatial differencing schemes as they relate to problem physics, decomposition, and load balancing, (4) To test the benefits of differencing and acceleration methods in conjunction with items (2) and (3) above, and (5) to provide higher order strategies for variable mesh coupling for increased accuracy and scalability in large transport problems.

Specific Objectives used for Code Development: The code should iteratively solve 3-D Cartesian, multigroup problems with anisotropic scattering via Legendre moments through P_7, with level symmetric angular quadratures through S_{20}. Industry standard *FIDO* input with vacuum and specular reflective boundaries should be allowed for. Sources should be definable as volumetric or plane surface incident fluxes, which may vary with space, angle, and energy. Further, a parallel memory structure should be used, where memory intensive arrays should be defined for local (as opposed to global) maximum dimensions to reduce storage extent and overhead. In addition, the code should permit varying mesh cells with a coarse grid topology. Also, the code should employ coarse mesh rebalance acceleration, as a minimum. Various differencing schemes should be readily selectable (e.g. linear diamond, directional theta-weighted, etc) via adaptive algorithm logic, and based on problem physics; "smart" ordering of scattering sweeps, such as Alternating Direction Sweeps, or *ADS* (Haghighat, 1992), and other tools, many readily extracted from the literature, should be implemented where practical, again according to problem physics. The code should be written in ANSI FORTRAN, and parallelized for message passing using the standardized MPI Message-Passing Interface library for portability to most any distributed memory parallel machine architecture (Gropp, et al, 1994).

Current Status: Coding of *PENTRAN* began in June 1995. A fully parallel two-grid (using "medium" and "fine" mesh transport grids) version of the code is in place in ANSI FORTRAN-90 and is currently 34,000+ lines. Scalable parallel process testing with complete angular, energy, and spatial domain decomposition, discontinuous meshing, multiple materials, first order zone coupling, fully definable sources, vacuum and reflective boundary conditions, multigroup, anisotropic scattering (benchmarked through P_5), multigrid coarse mesh zoned rebalancing, and selectable differencing for both forward and adjoint transport is complete. Test problems have demonstrated exact agreement (within the convergence criteria) with *TWOTRAN-II, THREEDANT, DORT,* and *TORT* production codes for all fixed source problems tested. Also, the code has been experimentally benchmarked in 3-D using models of the Venus-3 Reactor owned by SCK in Belgium and extensively tested (with excellent results) using the Kobayashi 3-D benchmark problems. Criticality eigenvalue problem results compared between *TWOTRAN-II* and *PENTRAN* also yielded excellent agreement, as did recent tests against *MCNP* in a variety of applications.

1.2 Unique *PENTRAN* Code Features

- The *PENTRAN* code has been designed from scratch in ANSI FORTRAN *and recently adapted to* ANSI FORTRAN-90 (to take advantage of dynamic array allocation) to be parallelized on a distributed memory, multiple instruction, multiple data (MIMD) machine architecture using the Message Passing Interface (MPI) message passing language. Any distributed memory MIMD parallel system running FORTRAN with MPI could be used to execute *PENTRAN* without modification. *PENTRAN* is also fully Y2K compliant.

- All Input/Output (I/O) is performed by each processor in parallel (as required) to the fullest extent possible. This includes input processing, initializations, and file output. Output files are written only for the local energy groups and spatial cells processed on a given processor (with local angles written in the case of angular fluxes), in accordance with local memory partitioning of the problem to *n* processors. A data management utility (*PENDATA*) is available to gather data automatically following a parallel *PENTRAN* run, and provides several options for the user in stripping results from parallel output files, including data extractions from binary file storage.

- Parallel memory utilization was a paramount design goal. All dimensions for memory intensive arrays (e.g. angular fluxes) are partitioned ***locally***. That is, these arrays only need be as large as required for the largest locally stored spatial grid, number of local energy groups, and local angular sweep octants based on the problem being solved. This is possible due to the independent memory of each processor on a distributed memory MIMD machine; in theory, if the problem becomes larger, one simply can add more processes (with further decomposition) to obtain a solution.

- Full phase space decomposition is available (space, angle, and energy), with fully automatic scaling of the problem to *n* processes (based on a user specified *decomposition weighting vector*). Also, a specific number of processes can be locked for each decomposition variable if desired; or, decomposition scaling in any one variable can be blocked (restricted to one processor). The automatic scaler/mapper will attempt to best adapt decomposition to the user's weighting vector and the number of processes assigned at execution. Following decomposition, *PENTRAN* includes automatic load balancing and red-black options to maximize efficiency.

- To further maximize parallel execution efficiency, communication among processors is, where practical, carried out only between processors specifically involved in a given task, e.g. for those processors all computing transport sweeps through particular coarse cells in the same energy group in a multigroup transport problem. This is accomplished by MPI process "communicators" constructed during problem initialization to exchange data between specific groups of processors. The number of communicators constructed is minimized to the number uniquely required to reduce network buffering overhead. The communicator structure is fully automatic and completely transparent to the user.

- Different differencing schemes can be assigned to different coarse meshes; therefore, differencing can be *adaptive* based on each coarse mesh. Linear Diamond Differencing without fixup (DD) and with set-to-zero fixup (DZ), Directional Theta-Weighted (DTW), Eponential-Directional Weighted (EDW), Exponential Directional Hybrid (EDH), and Exponential Directional Averaged (EDA) differencing schemes are independently selectable for each coarse mesh. The DD and EDA schemes are included only for test purposes and not intended for routine use. It is strongly recommended that DD and EDA *not* be used.

- An adaptive (DZ,DTW,EDW) scheme to automatically shift DZ to DTW (in the event a DZ negative flux fixup is detected), and shift from DTW to EDW (if a DTW angular flux weight factor that is too high is encountered) is available. Also, performance metrics are available for each differencing scheme, enabling the user to assess differencing performance in each coarse mesh for either the medium and/or fine mesh grids.

- Variable 3-D meshing (along each of x, y, and z) is available between different coarse meshes, with "medium" and "fine" multigrid meshing within each coarse cell. (Coarse cells are set to contain heterogeneous zones, where differencing is performed on medium and fine grids within each coarse mesh cell).

- As a consequence of variable 3-D meshing, Different spatial aspect ratios may be used along any set of axes in 3-D (e.g. boxoids) within each coarse mesh, and mesh grid distances need *not* be the same along adjoining surface boundaries between different coarse mesh cells. Mesh interpolations for angular fluxes between adjacent coarse cell surfaces (where coarse mesh cells are not necessarily on the same processor) are accomplished using Taylor Projection Mesh Coupling (TPMC). This is performed in all transport sweeps to increase accuracy and minimize the information loss when moderate and dense mesh grids are interfaced on a common boundary between two adjacent coarse cells. Note that particle conservation is strictly applied in this process. A simplified multigrid method that can provide speedup and conserves memory uses a single set of arrays for (local) angular fluxes on both the medium and fine mesh grids. The medium grid angular fluxes are relaxed to convergence using a tolerance less than that for the fine grid, whereupon they are projected onto the fine grid (overwriting the last medium grid values) using a new Taylor Projection Mesh Coupling (TPMC) scheme. The preconditioned fine grid values are then iterated to final convergence. This is a "simplified" multigrid method, since no residual corrections are projected with cycling between the medium and fine grids. Transport solution projections are only performed from medium to fine grids, and accelerate the solution by preconditioning the fine grid in a nested iteration. Varying speedups are possible with this simplified multigrid scheme, depending on the problem, with the advantage of conserving memory by not storing the medium grid explicitly.

- Standard coarse mesh rebalancing (CMR), implemented with restrictions and damping as Partial Current Rebalancing (PCR), and/or system rebalancing (SR) is available for acceleration, and can be used simultaneously with multigrid. There is no restriction on the differencing scheme that can be used with either CMR (PCR) or multigrid. Each processor independently performs rebalance using a direct solution (Cholesky factorization in the case of CMR (PCR)) over a zoned subset of coarse meshes (selected by the user) to obtain group rebalancing factors following transport sweeps. Further, rebalancing factors are used to scale only the *scalar* flux rather than the *angular* flux, as this prevents the need for message passing to complete additional angular quadratures prior to the next source iteration (if angular decomposition is invoked). Angular sweeps are ordered/decomposed in a sequence for Alternating Direction Sweeping (ADS). This can drastically increase convergence in some problems when used with rebalance (Haghighat, 1992).

- The spatial coarse mesh structure in *PENTRAN* fundamentally defines rebalance subdomains, parallel spatial decomposition subdomains, and adaptive Sn differencing subdomains. Although spatial decomposition is not required, more than a single coarse mesh must be defined to permit spatial decomposition on more than a single processor. Since a processor synchronization is performed following completion of a coarse mesh, a sufficient number of fine meshes should be contained within each coarse mesh to maintain computational load granularity and rebalance integrity. **One restriction is that there be an equal number of defined coarse cells, energy groups, and directions partitioned to each processor; this is needed for parallel synchronization to prevent "deadlocks." PENTRAN performs this automatically to the extent permissible, again based on the decomposition weight vector specified by the user.**

- Fixed sources can be defined as volumetric or as planar boundary fluxes; sources can have completely arbitrary spatial, angular, and/or energy distributions. Spatial and angular source distributions can be defined independently by energy group. Group dependent source scale factors can be specified. Criticality eigenvalue solutions are fully implemented and have been benchmarked using test problems. Criticality problems and problems that include upscatter can not be used with the group window option or the restart option; these features are currently reserved for downscatter-only fixed source problems.

- Industry standard, free field format *FIDO* input is used, with standardized order for cross sections. All cross sections are assumed to be blended and assembled outside of *PENTRAN*. Provisions are made for row, column, binary, and *ORNL-GIP*-binary formats, with and without Legendre coefficients multiplied in advance. Wall-clock time, maximum iterations, and convergence tolerance are all independent means of execution control, particularly useful for batch data processing.

- Planar images (*z*-levels) are available using the *PENMSH* utility. This utility generates a 3-D Cartesian mesh and can be used in conjunction with *PENINP* to automatically generate a *PENTRAN* input deck. Graphical schematics of 3-D problem geometry are automatically generated for use in the *Mathematica*^{TM} computer algebra package, where *PostScript* images are rendered; however, the 3-D rendering of images requires a significant effort in *Mathematica*. Use of *PENMSH/PENINP* reduces problem input preparation to a trivial process, even for large, intricate geometries; this is marketed as a separate package by H&S Advanced Computing Technologies.

- The code is written in ANSI FORTRAN-90; it has been implemented on a single-processor *Pentium*-PC, an *IBM RISC*-6000 Workstation, *SUN* Workstation, and in parallel on the *IBM Scalable PowerParallel System-2* (*IBM-SP2*) and *SGI Origin-2000* supercomputers, and on a *SPARKYcluster* from H&S Advanced Computing Technologies using *Linux* based PCs. Benchmark testing has demonstrated that *PENTRAN* is greater than 97% parallelizeable, resulting in excellent parallel speedups. Actual performance depends on the problem being solved, differencing and acceleration methods used, problem load balancing, red-black coloring, and applied decomposition strategy. Also, due to the inherent scalable parallel memory structure used, increasingly large 3-D problems that could not be solved on single processor platforms and/or within a reasonable time period can be solved in parallel using *PENTRAN* by adding processors.

2. Theory and Application

2.1 Multiprocessing Terminology

The idea of multiprocessing and the evolution of parallel computing began in the late 1950s and continues today. Due to the ten-fold increase in computational performance during each five year period since then, multiprocessing has made great progress. The reasons for attempting to solve any numerically intensive problem with multiprocessing are simple: to reduce execution time, obtain higher accuracy, and/or solve problems that are larger than can be solved using a traditional, single processor von Neumann architecture (Freeman and Phillips, 1992). It is useful to define some common parallel processing terms; they are only briefly mentioned here and will appear throughout this manual.

There are four classes of machines, as introduced by Flynn (1972):

- SISD - Single Instruction/Single Data Stream (traditional von Neumann machine)
- SIMD - Single Instruction/Multiple Data Stream (lock-step arrays, vector machines)
- MISD - Multiple Instruction/Single Data Stream (all tasks contribute to one data set)
- MIMD - Multiple Instruction/Multiple Data Stream (multiple independent tasking)

Shared memory MIMD systems are constructed so that each processor has global memory access and are typically limited to tens of processors due to the large number of physical connections to the memory map. *Distributed memory* MIMD parallel computers maintain completely independent memories, where processors exchange information by message passing over a high speed network; each processor independently executes code and can perform independent input/output (I/O) if allowed for in the parallel algorithm. Distributed memory processors, viewed as "nodes," are typically connected together using a variety of topologies, and can range in number into the thousands.

Parallel performance models are necessary for analyzing and quantifying parallel speedup and efficiency. Parallel *speedup* (S_P) measures the overall reduction in computing time to solve a problem. It is defined as the wall-clock time on a serial (single) processor divided by the wall-clock time on P processors:

(2.1) $S_P = T_S / T_P$

Parallel *efficiency* (E_P) measures the economic advantage of the parallelization by comparing the speedup factor to the allocated number of processors (Freeman and Phillips, 1992):

(2.2) $E_P = S_P / P$

It is assumed here that the parallel algorithm *overhead,* the extra executable code and storage required to expedite parallel execution, is negligible during single processor execution. This is a typical convention often adopted in measuring parallel speedup (Werner, 1981).

Also, an upper bound on anticipated parallel speedup can be determined by applying the *Amdahl's Law*, which states that given the fraction of a code that is parallelizeable: $0<f_P<1$, the maximum observed speedup for P processors with parallel communication time (T_C) is equal to:

(2.3) $$S_P = \frac{1}{(1-f_P)+f_P/P+T_C/T_S}$$

where in the limit of an infinite number of processes (assuming zero communication time):

(2.4) $$\lim_{p\to\infty} S_P \to \frac{1}{(1-f_P)}$$

Therefore, from equation (2.4), if the parallelizeable portion of a code is $f_P = 0.80$, the maximum *theoretically* observed speedup is 5.0 regardless of the number of additional processors added to the problem. In reality, due to increasing parallel instruction and communication overhead with the addition of more and more processors, there will be a point (depending on f_P, system architecture, and problem size) where adding more processors leads to extremely low efficiencies. This may be irrelevant if the code is scalable in memory (as in the case of *PENTRAN*), where, regardless of speed, the problem requires some number of processors to be solved *at all*.

Other important terms include *load balancing* and *granularity*. Load balancing involves distributing work to processors evenly to maximize parallel efficiency. Algorithm granularity is a qualitative term that refers to the number of process operations that can be executed by each processor before a synchronization (or communication) of the processors must be implemented. These synchronizations can be viewed as serial barriers that limit parallel performance. Conventional definitions of grain size are:

- fine grain - unit numbers of operations before synchronization
- medium grain - tens of operations before synchronization
- coarse grain - hundreds (or more) of operations before synchronization

The *computation/communication ratio* varies from machine to machine; this is the ratio of the CPU instruction speed in *flops* (*fl*oating *p*oint *o*peration*s* per *s*econd speed, often stated in *Mflops*, or *millions* of *flops*) attainable on the system to the relative speed of data transfer between processors. The speed of data transfer is related to communication *latency*, or the time required to send a zero byte-length message, and the communication *bandwidth*, in megabytes per second. The computation/ communication ratio is often more of an issue on distributed memory, message passing machines, as network communication data rates are typically orders of magnitude slower than typical *Mflop* rates (Gropp, et al, 1994).

For illustration, the *IBM SP2,* as of 1996-1997, has these characteristics:

- Processor performance of 266 Mflops/node (peak), minimum of 128 Mb/node
- Message latency of 500 nsec, bandwidth 35 Mb/s (peak)
- Computation/communication ratio of 7.6 Mflop/Mb-transfer (using peak values)
- Based on these values, a coarse grained code architecture should provide optimum results on the SP2.

Parallel machine availability is a practical performance issue. If processors are available to users in a *dedicated* mode, then during parallel execution, a single user has complete control of the processors, and parallel performance can be accurately determined. Alternatively, in a *non-dedicated* (interactive) mode, processors are simultaneously available to many users; *absolute* parallel performance may be difficult to verify in this case.

With regard to a working definition of parallel scalability, scalable algorithms maximize computation to communication ratio, minimize serial operations, maximize algorithm and data parallelism, and maximize efficiency for the architecture (shared versus distributed memory) (Gerner, 1995). While there is a great deal more that can be mentioned about multiprocessing fundamentals and terminology, more complete discussions can be found elsewhere (see Freeman and Phillips, 1992, and Gropp, et al, 1994, and Chandy and Misra, 1988).

2.2 Deterministic Transport Methods

Deterministic discrete ordinates approximations of the transport equation invoke a discretization of the energy, angle, and space variables. Discretization of the energy variable is accomplished by spectrally averaging over energy groups (*g=1,G*), from high energies to low energies, resulting in the multigroup transport formulation. In steady state, the multigroup transport equation is (Lewis and Miller, 1993):

(2.5) $$\hat{\Omega}\cdot\nabla\psi_g(\vec{r},\hat{\Omega})+\sigma_g(\vec{r})\psi_g(\vec{r},\hat{\Omega})=$$
$$\sum_{g'=1}^{g}\int_{4\pi}d\Omega'\sigma_{s\,g'\to g}(\vec{r},\hat{\Omega}'\cdot\hat{\Omega})\psi_g(\vec{r},\hat{\Omega}')+\frac{\chi_g}{k_o}\sum_{g'=1}^{G}\int_{4\pi}d\Omega'\nu\sigma_{f\,g'}(\vec{r})\psi_{g'}(\vec{r},\hat{\Omega})$$

Note that the angular variable is normalized on the unit sphere in the above formulation, so that integration over Ω is expressed in terms of the polar angle cosine μ and azimuthal angle φ as:

(2.6) $$\int_{4\pi}d\Omega=\int_{-1}^{1}\frac{d\mu}{2}\int_{0}^{2\pi}\frac{d\varphi}{2\pi}=1$$

Hereafter, this is implicitly assumed. The scattering term is then expanded using a truncated set of spherical (surface) harmonics, with $\hat{\Omega}\to<\theta,\varphi>,\ (\hat{\Omega}'\cdot\hat{\Omega})\to(\mu_o),\ \mu=(\cos\theta),\ \mu'=(\cos\theta')$:

(2.7) $$\sigma_{s\,g'\to g}(\vec{r},\mu_o)=\sum_{l=0}^{L}(2l+1)\sigma_{s\,g'\to g,l}(\vec{r})P_l(\mu_o)$$

(2.8) $$\sigma_{s\,g'\to g,l}(\vec{r})=\int_{-1}^{1}\frac{d\mu_o}{2}\sigma_{s\,g'\to g}(\vec{r},\mu_o)P_l(\mu_o)$$

(2.9) $$\mu_o=\mu\mu'+(1-\mu^2)^{1/2}(1-\mu'^2)^{1/2}\cos(\varphi-\varphi')$$

The Legendre polynomial $P_l(\mu_o)$, using the Legendre Addition Theorem, is:

(2.10) $$P_l(\mu_o)=\frac{1}{(2l+1)}\sum_{k=-l}^{l}Y_{l,k}^{*}(\theta',\varphi')Y_{l,k}(\theta,\varphi)$$

The spherical (surface) harmonics $Y_{l,k}$ and $Y_{l,k}^{*}$ are defined in terms of the Associated Legendre polynomials and an exponential term:

(2.11) $$Y_{l,k}(\theta,\varphi)=\sqrt{(2l+1)\frac{(l-k)!}{(l+k)!}}P_l^k(\mu)\exp(ik\varphi)$$

(2.12) $$Y_{l,-k}(\theta,\varphi)=(-1)^kY_{l,k}^{*}(\theta,\varphi)$$

Using equations (2.10), (2.11), and (2.12), $P_l(\mu_o)$ can be written:

(2.13) $$P_l(\mu_o) = P_l(\mu)P_l(\mu') + 2\sum_{k=1}^{l}\frac{(l-k)!}{(l+k)!}P_l^k(\mu)P_l^k(\mu')\cos(k(\varphi-\varphi'))$$

By trigonometric identity:

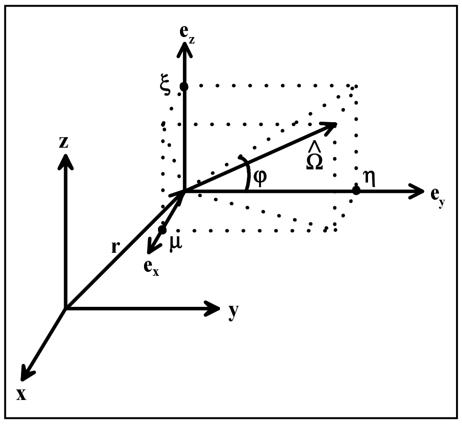
(2.14) $$\cos(k(\varphi-\varphi')) = \cos(k\varphi)\cos(k\varphi') + \sin(k\varphi)\sin(k\varphi')$$

The vectors $\hat{\Omega} \rightarrow <\theta,\varphi>$ on the unit sphere can be expressed as a set of direction cosines projected parallel to the *x, y,* and *z* axes, respectively, as $<\mu,\eta,\xi>$ Note that η and ξ can be expressed in terms of polar angle cosine μ and the azimuthal angle φ:

(2.15) $$\eta = \sqrt{1-\mu^2}\cos(\varphi)$$

(2.16) $$\xi = \sqrt{1-\mu^2}\sin(\varphi)$$

A 3-D Cartesian geometry (using a right handed coordinate system) is shown in Figure below.



3-D Cartesian Geometry

If the streaming operator $\hat{\Omega}\cdot\nabla$ in equation (2.5) is expanded in 3-D Cartesian coordinates, it becomes:

(2.17) $$\hat{\Omega}\cdot\nabla = \mu\frac{\partial}{\partial x} + \eta\frac{\partial}{\partial y} + \xi\frac{\partial}{\partial z}$$

Substituting equations (2.7), (2.13), (2.14), and (2.17) into equation (2.5), we obtain the Legendre expanded multigroup form of the transport equation in 3-D Cartesian geometry (Lewis and Miller, 1993, and Bell and Glasstone, 1985):

(2.18)
$$(\mu\frac{\partial}{\partial x}+\eta\frac{\partial}{\partial y}+\xi\frac{\partial}{\partial z})\psi_g(x,y,z,\mu,\varphi)+\sigma_g(x,y,z)\psi_g(x,y,z,\mu,\varphi)=$$
$$\sum_{g'=1}^{G}\sum_{l=0}^{L}(2l+1)\sigma_{s,g'\to g,l}(x,y,z)\{P_l(\mu)\phi_{g',l}(x,y,z)+2\sum_{k=1}^{l}\frac{(l-k)!}{(l+k)!}P_l^k(\mu)\cdot$$
$$[\phi^k_{C\,g',l}(x,y,z)\cos(k\varphi)+\phi^k_{S\,g',l}(x,y,z)\sin(k\varphi)]\}+\frac{\chi_g}{k_o}\sum_{g'=1}^{G}\nu\sigma_{f\,g'}(x,y,z)\phi_{g',0}(x,y,z)$$

where	$\mu =$ *x* direction cosine for angular ordinate
	$\eta =$ *y* direction cosine for angular ordinate
	$\xi =$ *z* direction cosine for angular ordinate
	$\psi_g =$ group *g* angular particle flux (for groups *g*=*1*,*G*)
	$\varphi =$ azimuthal angle constructed from $\arctan(\xi/\eta)$, with proper phase shift
	$\sigma_g =$ total group macroscopic cross section
	$l =$ Legendre expansion index ($l = 0, L$), *L*=0 or odd truncation
	$\sigma_{sg'\to g,l} = l$ th Legendre moment of the macroscopic differential scattering cross section from group $g' \to g$ (Equation (2.7))
	$P_l(\mu) =$ l th Legendre polynomial
	$\phi_{g',l} =$ l th Legendre scalar flux moment for group *g*
	$P_l^k(\mu) =$ l th , k th Associated Legendre polynomial
	$\phi^k_{C\,g',l} =$ l th , k th Cosine Associated Legendre scalar flux moment for group *g*
	$\phi^k_{S\,g',l} = l$ th , k th Sine Associated Legendre scalar flux moment for group *g*
	$\chi_g =$ group fission distribution constant (neutrons)
	$k_o =$ criticality eigenvalue (neutrons)
	$\nu\sigma_{f\,g} =$ group fission production (neutrons)

The flux moments, $\phi_{g',l}$, $\phi^k_{C\,g',l}$ and $\phi^k_{S\,g',l}$ are defined in terms of μ' and φ' as:

(2.19) $$\phi_{g',l}(x,y,z)=\int_{-1}^{1}\frac{d\mu}{2}P_l(\mu')\int_0^{2\pi}\frac{d\varphi'}{2\pi}\psi_{g'}(x,y,z,\mu',\varphi')$$

(2.20) $$\phi^k_{C\,g',l}(x,y,z)=\int_{-1}^{1}\frac{d\mu}{2}P_l^k(\mu')\int_0^{2\pi}\frac{d\varphi'}{2\pi}\cos(k\varphi')\psi_{g'}(x,y,z,\mu',\varphi')$$

(2.21) $$\phi^k_{S\,g',l}(x,y,z)=\int_{-1}^{1}\frac{d\mu}{2}P_l^k(\mu')\int_0^{2\pi}\frac{d\varphi'}{2\pi}\sin(k\varphi')\psi_{g'}(x,y,z,\mu',\varphi')$$

To solve for the adjoint function using the adjoint transport equation, the forward transport equation (2.18) can be used if all angles $\hat{\Omega}$ are taken to be $-\hat{\Omega}$, with angular and energy group indexes transposed (since the transport operator is not self adjoint). Any forward transport algorithm can be used to solve adjoint transport problems if the cross sections and sources are transposed and group re-ordered from group G to 1, with angles implicitly defined in opposite directions (Bell and Glasstone, 1985). Once solved for, the adjoint function provides the neutron or photon importance throughout the problem phase space relative to a particular response, defined by the adjoint source. The adjoint function can then be used in several ways. One use of the adjoint function is to determine the regions/energies that most affect the response to help determine limiting mesh intervals in the geometry. Further, a deterministic adjoint solution may be used to assign importances for variance reduction in non-analog Monte-Carlo applications, which can add extreme efficiency to such calculations (Wagner and Haghighat, 1996). *PENTRAN* can be used to solve for the adjoint function; transposition of all cross sections, etc is performed internally by the code, and the user is reponsible only for properly defining the transposed adjoint source, and noting that groups and directions are reported implicitly reversed.

2.3 Discrete Ordinates and Quadrature

In the discrete ordinates approximation, it is assumed that the transport equation only holds for a set of *M* distinct angular directions $\hat{\Omega} \rightarrow < \mu, \eta, \xi >$ on the unit sphere. In standard S_N calculations, a numerical quadrature is used to integrate the discrete ordinate angular fluxes to obtain flux moments. In practice, quadrature sets must have the following properties:

(2.22) $$\sum_{m=1}^{M} w_m = 1.0$$

(2.23) $$\sum_{m=1}^{M} w_m \mu_m^n = \sum_{m=1}^{M} w_m \eta_m^n = \sum_{m=1}^{M} w_m \xi_m^n = 0 \quad \text{for } n \text{ odd, since} \quad J_{net} = \sum_{m=1}^{M} w_m \Omega_m \psi_m$$

(2.24) $$\sum_{m=1}^{M} w_m \mu_m^n = \sum_{m=1}^{M} w_m \eta_m^n = \sum_{m=1}^{M} w_m \xi_m^n = \frac{1}{n+1} \quad \text{for } n \text{ even}$$

Because net current J_{net} in an isotropic flux is *zero,* this requires that equations (22) and (23) hold true. Equations (2.23) are known as the *odd moment conditions,* and because they must be satisfied, the quadrature set must be symmetric on the unit sphere for each set of directions, invariant with respect to 90-degree axis rotations. Equations (2.24) are known as the *even moment conditions,* required to insure proper integration of the Legendre functions. The Legendre polynomials must be represented due to the expansion in the scattering term, so that:

(2.25) $$\frac{\delta_{l\,l'}}{2l+1} = \int_{-1}^{1} \frac{d\mu}{2} P_l(\mu') P_{l'}(\mu')$$

and for Associated Legendre Polynomials:

(2.26) $$\frac{\delta_{l\,l'}\delta_{k\,k'}}{2l+1}\frac{(l+k)!}{(l-k)!} = \int_{-1}^{1} \frac{d\mu}{2} P_l^k(\mu') P_{l'}^{k'}(\mu')$$

For example, from the first even moment condition, with *n*=2, the quadrature set should satisfy equations (2.25) and (2.26) for the P_l (first order only) Legendre Polynomials. Since, for any unit angular direction vector $\hat{\Omega}$, the ordinate must lie on the unit sphere:

(2.27) $$\mu_i^{\ 2} + \eta_j^{\ 2} + \xi_k^{\ 2} = 1 \qquad \text{where} \quad i \in [1, M], j \in [1, M], k \in [1, M]$$

and due to the required symmetries in three dimensions, the following recursion relationship holds for any given octant:

(2.28) $$\mu_i^{\ 2} = \mu_1^2 + C(i-1) \qquad \text{where} \quad C = \frac{2}{N-2}(1-3\mu_1^2) \quad \text{and} \quad 2 \le i \le \frac{N}{2}$$

The *N* in 3-D S_N corresponds to the number of levels from ***each*** direction cosine on the unit sphere, and there are *M=N(N+2)* ordinates on the unit sphere, with M_{oct}=*N(N+2)/8* in each octant, with *N/2* distinct direction cosine values (Stamm'ler and Abbate, 1983). For example, to derive an S_6 level symmetric quadrature set using the first octant, six equations with six unknowns must be solved for to provide unique, symmetric direction cosines and corresponding level weights. The following equations must be solved simultaneously using equations (2.22), (2.24), and (2.28) (equations (2.23) are then satisfied implicitly):

(2.29) **Symmetry conditions:**

$w_1 + w_2 + w_3 = 1$ (weights will be multiplied by 1/8 for all octants, *M* ordinates)

$\mu_2{}^2 = \mu_1^2 + C(2-1)$ (with $C = \frac{2}{6-2}(1-3\mu_1^2)$)

$\mu_3{}^2 = \mu_1^2 + C(3-1)$ (again with $C = \frac{2}{6-2}(1-3\mu_1^2)$)

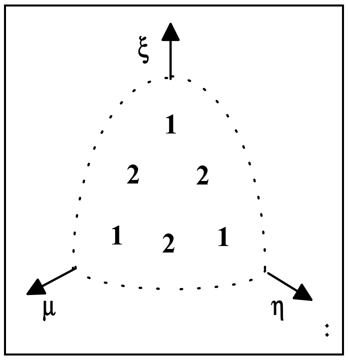
(2.30) **Even Moment Conditions:**

$$w_1\mu_1^2 + w_2\mu_2^2 + w_3\mu_3^2 = \frac{1}{2+1} \qquad w_1\mu_1^4 + w_2\mu_2^4 + w_3\mu_3^4 = \frac{1}{4+1}$$

$$w_1\mu_1^6 + w_2\mu_2^6 + w_3\mu_3^6 = \frac{1}{6+1}$$

Since the weights w_i in equations (2.29) and (2.30) are *level weights* (note there are 6/2=3 *levels* in an octant for S_6), another set of equations is required to obtain *point weights* $w_{m\,i}$; these can be derived from the ordinate pattern in the first octant (see the Figure at right depicting the point weight pattern within an octant):

(2.31) $2w_{m1} + w_{m2} = w_1$; $2w_{m2} = w_2$; $1w_{m1} = w_3$



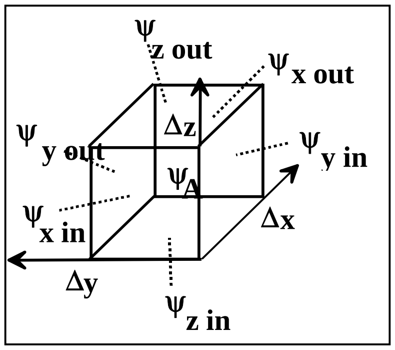
Point Weight Pattern for S_6

Using Equations (2.29) to (2.31), the quadrature set for S_6 can be solved for. Note that from the even moment conditions in equation (2.30), this S_6 quadrature set will properly integrate Legendre moments through P_3 . All quadratures from S_2 through S_{20} were derived for use in *PENTRAN* using equations (2.22), (2.24), and (2.28), along with equations similar to (2.31). No level symmetric quadratures satisfying Equations (2.25) and (2.26) are available beyond S_{20} due to the appearance of unphysical negative weights.

2.4 Differencing Schemes

At this point, a spatial approximation to equation (2.18) is required. As mentioned earlier, several approaches can be made to formulate a discrete ordinates spatial differencing scheme. Zeroth *spatial* Legendre Functions are:

(2.32) $P_0(x) = 1 \qquad P_0(y) = 1 \qquad P_0(z) = 1$



Cell Volume Element

To derive the zeroth spatial moment balance equation, equation (2.18), multiplied by equations (2.32), is integrated over a local cell volume and divided by the integral of the product of equations (2.32), also integrated over the cell volume. For our purposes, a cell volume has parallelepiped dimensions $(\Delta x, \Delta y, \Delta z)$. Assuming that particles are traveling in a positive direction, edge and center flux integrals are represented by integral averages. If we consider that directions traveled could be negative, then $\mu_m \to |\mu_m|$, $\eta_m \to |\eta_m|$, and $\xi_m \to |\xi_m|$ for a positive sense in the equations, which always occurs if the equations are derived *in the direction of particle motion*. The *zeroth spatial moment balance equation* (omitting the group *g* subscript for brevity) is:

(2.33)
$$\frac{|\mu_m|}{\Delta x}(\psi_{\text{out}x} - \psi_{\text{in}x}) + \frac{|\eta_m|}{\Delta y}(\psi_{\text{out}y} - \psi_{\text{in}y}) + \frac{|\xi_m|}{\Delta z}(\psi_{\text{out}z} - \psi_{\text{in}z}) + \sigma\psi_A = q_A$$

For a positive angular vector, where $\{\mu_m, \eta_m, \xi_m\} > 0,$ the entering and exiting surface averaged angular fluxes normal to the *x*-axis are given by:

(2.34)
$$\psi_{\text{in}\,x} = \frac{1}{\Delta y \Delta z}\int_0^{\Delta y}\int_0^{\Delta z} \psi_m(0,y,z)\, P_0(y)\, P_0(z)\, dy\, dz$$

$$\psi_{\text{out}\,x} = \frac{1}{\Delta y \Delta z}\int_0^{\Delta y}\int_0^{\Delta z} \psi_m(\Delta x,y,z)\, P_0(y)\, P_0(z)\, dy\, dz$$

Surface terms normal to the *y*- and *z*- dimensions are defined in a similar manner. The cell volume-averaged angular flux is given by:

(2.35)
$$\psi_A = \frac{1}{\Delta x \Delta y \Delta z}\int_0^{\Delta x}\int_0^{\Delta y}\int_0^{\Delta z} \psi_m(x,y,z)\, P_0(x)\, P_0(y)\, P_0(z)\, dx\, dy\, dz$$

The volume-averaged source term q_A is defined in a similar manner. Note that we refer to the surface averaged terms that enter and leave the cell as the "in" and "out" subscripts, respectively, while the "*A*" subscripts denote cell average quantities.

Equation (2.33) is exact, but contains seven unknowns. We can consider the three entrant values ("in") are known from boundary values, and that the collective cell averaged volumetric source q_A is assumed to be known from a previous source iteration (in the standard S_N source iteration scheme). Therefore, only the cell average angular flux ψ_A and the exiting ("out") surface values are unknowns, where these latter values are obtained using a set of *auxiliary equations*. Auxiliary equations amount to "fitting functions" that resemble the behavior of the angular flux across the spatial cell, and they establish the accuracy of the differencing method (Lewis and Miller, 1993).

For weighted spatial differencing schemes, the following auxiliary equations are assumed to hold between cell average and boundary angular fluxes:

(2.36)
$$\psi_{\text{out } x} = 1/a(\psi_A + \psi_{\text{in } x}(a-1))$$
$$\psi_{\text{out } y} = 1/b(\psi_A + \psi_{\text{in } y}(b-1))$$
$$\psi_{\text{out } z} = 1/c(\psi_A + \psi_{\text{in } z}(c-1))$$

Note that the standard Diamond Differencing (DD) scheme results when *a*=½, *b*=½, and *c*=½ in equations (2.36); the DD scheme is second order accurate, but may lead to negative solutions. (Lewis and Miller, 1993). In such situations, a "negative flux set to zero fixup" of the Diamond scheme is commonly used. In this paper, we denote the Diamond scheme with the zero fixup as Diamond-Zero (DZ). Furthermore, it is worth noting that the negative flux fixup has also demonstrated to be the source of load imbalance in parallel processing solutions (Haghighat, Hunter, and Mattis, 1995). To overcome the inherent difficulties of the Diamond scheme, Rhoades and Engle (1977) developed the Theta-Weighted (TW) scheme that is always positive.

Recently, Petrovic and Haghighat (1996) have shown that in multidimensional geometries, non-physical oscillations occur because of the "mismatch" between the direction of particles (along a characteristic) and the spatial axis where the differencing is carried out, even with very high mesh refinement. The oscillations are attributed to a "forced" relationship of the average angular flux to the boundary fluxes (depending on the auxiliary equation), where no directionally dependent boundary contribution relative to each axis is taken into account (Petrovic and Haghighat, 1996). The non-physical oscillations inherent in solutions rendered by the Diamond scheme only add to the previously mentioned difficulties with positivity.

To remedy this, Petrovic and Haghighat developed the Directional Theta-Weighted scheme (Petrovic and Haghighat, 1996) that is an extension of TW. To derive a Cartesian form of the TW scheme for weight factor *a* (for $\psi_{\text{out}x}$), equations (2.36) are placed into equation (2.33). Solving for ψ_A, and again using equations (2.36), an expression for $\psi_{\text{out } x}$ is obtained. The Diamond relations are then assumed to hold for the *y*- and *z*- directions (with *b*=½ and *c*=½).

To force positivity, arbitrary fixed theta-weighting parameters (θ) are introduced into the formulation; $|\mu|/(a\Delta x)$ is dropped from the denominator, and *a* is assimilated into the parameters. Assuming the lower bound of $\psi_{\text{out}x}$ is zero, we obtain an equation for the "*a*" weight:

(2.37)

$$a = 1 - \frac{q_A + \frac{|\mu_m|}{\Delta x}\psi_{\text{in }x} + \theta\left(\frac{|\eta_m|}{\Delta y}\psi_{\text{in }y} + \frac{|\xi_m|}{\Delta z}\psi_{\text{in }z}\right)}{\left(2\frac{|\eta_m|}{\Delta y} + 2\frac{|\xi_m|}{\Delta z} + \sigma\right)\psi_{\text{in }x}}$$

Using a similar procedure along the *y*- and *z*-axes to yield weights for "*b*" and "*c*," respectively:

(2.38)

$$b = 1 - \frac{q_A + \frac{|\eta_m|}{\Delta y}\psi_{\text{in }y} + \theta\left(\frac{|\mu_m|}{\Delta x}\psi_{\text{in }x} + \frac{|\xi_m|}{\Delta z}\psi_{\text{in }z}\right)}{\left(2\frac{|\mu_m|}{\Delta x} + 2\frac{|\xi_m|}{\Delta z} + \sigma\right)\psi_{\text{in }y}}$$

(2.39)

$$c = 1 - \frac{q_A + \frac{|\xi_m|}{\Delta z}\psi_{\text{in }z} + \theta\left(\frac{|\mu_m|}{\Delta x}\psi_{\text{in }x} + \frac{|\eta_m|}{\Delta y}\psi_{\text{in }y}\right)}{\left(2\frac{|\mu_m|}{\Delta x}\psi_{\text{in }x} + 2\frac{|\eta_m|}{\Delta y} + \sigma\right)\psi_{\text{in }z}}$$

Petrovic and Haghighat specified that for maximum smoothing in directions perpendicular to respective characteristics (to minimize oscillations), the fixed theta parameters in the TW scheme could be modified to allow variable theta parameters that are dependent on the characteristic of the incident radiation. As a result, the Directional Theta Weighted (DTW) scheme employs the 3-D theta parameters given in equation (2.40) substituted for each θ parameter in equations (2.37), (2.38), and (2.39).

(2.40)

$$\theta(\mu_m) = \mu_m^2 \quad ; \quad \theta(\eta_m) = \eta_m^2 \quad ; \quad \theta(\xi_m) = \xi_m^2$$

where μ, η, ξ are the direction cosines along the *x*-, *y*-, and *z*- axes, respectively.

The angular flux weighting factors are subsequently used to obtain the DTW average cell angular flux, given by:

$$\psi_A = \frac{q_A + \frac{|\mu_m|}{a\Delta x}\psi_{\text{in}\,x} + \frac{|\eta_m|}{b\Delta y}\psi_{\text{in}\,y} + \frac{|\xi_m|}{c\Delta z}\psi_{\text{in}\,z}}{\frac{|\mu_m|}{a\Delta x} + \frac{|\eta_m|}{b\Delta y} + \frac{|\xi_m|}{c\Delta z} + \sigma}$$

(2.41)

Therefore, using equations (2.40) in equations (2.37), (2.38), and (2.39), respectively, the DTW scheme uses direction-based parameters to obtain angular flux weighting factors, from which average and inherently positive exiting angular fluxes are derived using equation (2.36). The DTW scheme is clearly non-linear in the way the angular flux weights (*a,b,c*) are derived from directionally dependent parameters, incident fluxes, and volumetric sources. To be consistent, these weights are restricted to the range between ½ and 1, with accuracy approaching second order truncation when all weights are ½ (equivalent to the Diamond scheme). This truncation error is evident if equations (2.36) are substituted into (2.33). If that result is subtracted from equation (2.33), and Taylor's series expansions are applied about ψ_A, the trucation error of the DTW formulation is:

$$\in_{DTW} = (1-\frac{1}{2a})\frac{\Delta x}{\rho_x}\frac{\partial\psi}{\partial x}|_A + \frac{\Delta x^2}{8a\rho_x}\frac{\partial^2\psi}{\partial x^2}|_A +$$

(2.42)

$$(1-\frac{1}{2b})\frac{\Delta y}{\rho_y}\frac{\partial\psi}{\partial y}|_A + \frac{\Delta y^2}{8b\rho_y}\frac{\partial^2\psi}{\partial y^2}|_A + (1-\frac{1}{2c})\frac{\Delta z}{\rho_z}\frac{\partial\psi}{\partial z}|_A + \frac{\Delta z^2}{8c\rho_z}\frac{\partial^2\psi}{\partial z^2}|_A + O(\Delta^3)$$

where $\rho_u = \frac{\sigma\Delta u}{|\tau\ |}$ for $u \in \{x,y,z\}$ and $\tau_m \in \{\mu_m,\eta_m,\xi_m\}$

A *positive* truncation error $\in_{DTW}$ indicates that DTW will *underestimate* the solution, while a *negative* truncation error indicates DTW will *overestimate* the solution. When DTW weights are all ½, the truncation error is identical to that of the Diamond scheme, influenced only by second partial derivatives. Note that when DTW weights are greater than ½, the truncation error is influenced by *both* first *and* second partial derivatives of the angular flux (Sjoden, 1997). Because of the directional weighting of DTW, a set of angular fluxes along different paths will contain both over- *and* underestimated angular fluxes. In addition to being positive and free of oscillations from the directional weighting, the DTW scheme can be significantly more accurate than the Diamond scheme, mainly because we typically are interested only in the *scalar flux* (integrated over all directions). The *combined* effects of over- and underestimates of the angular flux among different directions with DTW often cancel during integration (quadrature), resulting in more accurate *scalar fluxes*. In the special case where the flux is relatively flat (such as in the middle of a reactor core), DTW weights will be near unity, and the truncation error will be very small due to small flux gradients.

While DTW may not be a highly accurate scheme in all situations, it behaves reliably in general situations (positivity, stability, with derivatives having proper signs, etc). Therefore, a *predictor- corrector* exponential scheme that uses DTW to *predict* a solution that is then *corrected* by an exponential fit should be stable and more accurate than DTW alone. Using this approach, the following inherently positive (provided the coefficient a_0 is positive) exponential auxiliary equation is proposed:

(2.43) $$\psi_m(x,y,z) = a_o \, exp(\lambda_i \, P_1(x)/|\mu_m|) \, exp(\lambda_j \, P_1(y)/|\eta_m|) \, exp(\lambda_k \, P_1(z)/|\xi_m|)$$

First order spatial Legendre functions, orthogonal to equations (2.32) over the widths of a single cell, are:

(2.44) $$P_1(u) = \frac{2u}{\Delta u} - 1 \quad \text{where} \quad 0 \le u \le \Delta u \quad \text{and} \quad u \in \{x,y,z\}$$

The exponential coefficients (λ) define the overall profile of equation (2.43); these coefficients are normally obtained by root solving 1st moment transcendental conservation equations (Mathews, Sjoden, and Minor, 1994), (Walters, Wareing, and D. Marr, 1995), and (Wareing and Alcouffe, 1995). To avoid the computational overhead of conserving the often ill-conditioned 1st moment balance equations, we can use a DTW solution to provide good estimates of these coefficients in a *predictor* step in the following manner. Consider a single cell of dimensions $(\Delta x, \Delta y, \Delta z)$. By taking first partial derivatives of equation (2.43) with respect to *x*-, *y*-, and *z*- axes, and assuming that in the limit as cell dimensions approach zero, $\psi(x,y,z) \to \psi_A$, then λ_i, λ_j, and λ_k can be separated as follows:

(2.45) $$\frac{1}{\psi_A}\frac{\partial \psi}{\partial x}|_A = \frac{2\lambda_i}{\Delta x|\mu_m|} \qquad \frac{1}{\psi_A}\frac{\partial \psi}{\partial y}|_A = \frac{2\lambda_j}{\Delta y|\eta_m|} \qquad \frac{1}{\psi_A}\frac{\partial \psi}{\partial z}|_A = \frac{2\lambda_k}{\Delta z|\xi_m|}$$

Then, the first partial derivatives in equation (2.45) can be approximated, again using "out" and "in" cell surface references, using a standard finite difference formulation, where $u \in \{x, y, z\}$:

(2.46) $$\frac{\partial \psi}{\partial u}|_A = \frac{(\psi_{\text{out}\,u} - \psi_{\text{in}\,u})}{\Delta u} - \frac{\Delta u^2}{24}\frac{\partial^3 \psi}{\partial u^3}|_A + O(\Delta u^3)$$

Then, using the *predicted* angular fluxes initially calculated using DTW (where DTW predicted angular fluxes are denoted by $\tilde{\psi}$) we can obtain explicit estimates (after algebraic simplification) for λ_i, λ_j, and λ_k. For the *x*-, *y*-, and *z*- dimensions, respectively:

(2.47) $$\lambda_i \approx \frac{(\tilde{\psi}_{\text{out}x} - \tilde{\psi}_{\text{in}\,x})|\mu_m|}{2\tilde{\psi}_A} \qquad \lambda_j \approx \frac{(\tilde{\psi}_{\text{out}y} - \tilde{\psi}_{\text{in}\,y})|\eta_m|}{2\tilde{\psi}} \qquad \lambda_k \approx \frac{(\tilde{\psi}_{\text{out}z} - \tilde{\psi}_{\text{in}\,z})|\xi_m|}{2\tilde{\psi}_A}$$

Note that use of equation (2.46) inherently assumes that the average of the first partial derivative of the angular flux is assumed to be at the cell center. However, note further that this equation only needs to estimate the derivative of the angular flux as opposed to the angular flux itself. Furthermore, the formulations cast in equation (2.47) demonstrate that λ_i, λ_j, and λ_k are based on a dimensionless ratio of the DTW predicted angular fluxes, thus reducing the sensitivity of computing a precise derivative. Since the DTW scheme provides estimates for the exponential constants, we must solve for the coefficient a_o in equation (2.43) using the zeroth moment balance defined in equation (2.33). To do this, we first perform the integrations for the *outbound* surface and cell volume angular fluxes as in equations (2.34) and (2.35) defined using the exponential auxiliary equation (2.43). (Note that *inbound* surface averaged fluxes and the cell volumetric source terms are assumed to be known). Placing the resulting formulations into the balance equation (2.33), we can obtain a solution (albeit rather cumbersome) for a_o.

Substituting the resulting expression for a_o back into the formulations for the outbound surface and volume averaged angular fluxes, we can obtain ψ_A (after more algebraic simplification):

$$\psi_A = \left(\exp(\frac{2\lambda_i}{|\mu_m|}) - 1\right)\left(\exp(\frac{2\lambda_j}{|\eta_m|}) - 1\right)\left(\exp(\frac{2\lambda_k}{|\xi_m|}) - 1\right) \cdot \frac{1}{\beta}\left(q_A + \frac{|\mu_m|}{\Delta x}\psi_{\text{in}\,x} + \frac{|\eta_m|}{\Delta y}\psi_{\text{in}\,y} + \frac{|\xi_m|}{\Delta z}\psi_{\text{in}\,z}\right) \tag{2.48}$$

where β in equation (2.48) is defined by:

$$\begin{aligned}\beta = {} & \frac{2\lambda_i}{\Delta x}\left(\exp(\frac{2\lambda_i}{|\mu_m|})\right)\left(\exp(\frac{2\lambda_j}{|\eta_m|}) - 1\right)\left(\exp(\frac{2\lambda_k}{|\xi_m|}) - 1\right) + \\ & \frac{2\lambda_j}{\Delta y}\left(\exp(\frac{2\lambda_j}{|\eta_m|})\right)\left(\exp(\frac{2\lambda_i}{|\mu_m|}) - 1\right)\left(\exp(\frac{2\lambda_k}{|\xi_m|}) - 1\right) + \\ & \frac{2\lambda_k}{\Delta z}\left(\exp(\frac{2\lambda_k}{|\xi_m|})\right)\left(\exp(\frac{2\lambda_i}{|\mu_m|}) - 1\right)\left(\exp(\frac{2\lambda_j}{|\eta_m|}) - 1\right) + \\ & \sigma\left(\exp(\frac{2\lambda_i}{|\mu_m|}) - 1\right)\left(\exp(\frac{2\lambda_j}{|\eta_m|}) - 1\right)\left(\exp(\frac{2\lambda_k}{|\xi_m|}) - 1\right)\end{aligned} \tag{2.49}$$

The outbound cell fluxes can be defined in terms of the cell average angular flux:

$$\begin{aligned}\psi_{\text{out}\;x} &= \psi_A \frac{2\lambda_i}{|\mu_m|}\left(1 - \exp(\frac{-2\lambda_i}{|\mu_m|})\right)^{-1} \\ \psi_{\text{out}\;y} &= \psi_A \frac{2\lambda_j}{|\eta_m|}\left(1 - \exp(\frac{-2\lambda_j}{|\eta_m|})\right)^{-1} \\ \psi_{\text{out}\;z} &= \psi_A \frac{2\lambda_k}{|\xi\ \ |}\left(1 - \exp(\frac{-2\lambda_k}{|\xi\ \ |})\right)^{-1}\end{aligned} \tag{2.50}$$

Equations (2.48) through (2.50) therefore provide a *correction* to the initial DTW *predicted* angular fluxes using exponential functions based on the auxiliary equation (2.43). This methodology allows the exponential coefficients (λ) to be predicted at *very low cost* using the DTW scheme (Sjoden and Haghighat, 1997). This is the Exponential Directional Weighted (EDW) method; it is absolutely positive, stable, directionally weighted, and is significantly more accurate than the DTW scheme in streaming problems with relaxed cell intervals. Demonstrations as to the effectiveness of EDW in streaming problems can be found in the literature.

To obtain an expression for truncation error of the EDW scheme, we again make the assumption that in the limit of small cells, the DTW and EDW solutions are identical. Expanding the exponential arguments using a Taylor's Series truncated to third order, substituting into equation (2.33), and then subtracting that result from equation (2.33) yields an expression for the truncation error of the EDW scheme (again after additional algebraic simplification):

$$\in_{EDW} = \frac{\Delta x}{2\rho_x}\frac{\partial\psi}{\partial x}|_A + \frac{\Delta x^2}{8\rho_x}\frac{\partial^2\psi}{\partial x^2}|_A + \tag{2.51}$$

$$\frac{\psi_A}{\rho_x}\left(1 - \frac{\psi_A}{\psi_A + (-\Delta x/2)\ \partial\psi/\partial x|_A + (\Delta x^2/(6\psi_A))\ (\partial\psi/\partial x|_A)^2}\right) + \frac{\Delta y}{2\rho_y}\frac{\partial\psi}{\partial y}|_A + \frac{\Delta y^2}{8\rho_y}\frac{\partial^2\psi}{\partial y^2}|_A +$$

$$\frac{\psi_A}{\rho_y}\left(1 - \frac{\psi_A}{\psi_A + (-\Delta y/2)\ \partial\psi/\partial y|_A + (\Delta y^2/(6\psi_A))\ (\partial\psi/\partial y|_A)^2}\right) + \frac{\Delta z}{2\rho_z}\frac{\partial\psi}{\partial z}|_A + \frac{\Delta z^2}{8\rho_z}\frac{\partial^2\psi}{\partial z^2}|_A +$$

$$\frac{\psi_A}{\rho_z}\left(1 - \frac{\psi_A}{\psi_A + (-\Delta z/2)\ \partial\psi/\partial z|_A + (\Delta z^2/(6\psi_A))\ (\partial\psi/\partial z|_A)^2}\right) + O(\Delta^3)$$

From equation (2.51), note that the truncation error (and therefore the accuracy) of the EDW scheme is dependent on both the slope and concavity (for each partial derivative) of the angular flux, with a strong influence from first partial derivatives. At this point, we have presented complete formulations for development of the DTW and EDW schemes in Cartesian geometry. In sections that follow, we focus on how each scheme is applied in *PENTRAN* in an adaptive differencing strategy.

The EDH differencing scheme is a completely adaptive combination of DTW and EDW. The application of either scheme is determined based on the flux gradient selectively for each ordiante, making this the most adaptive scheme selectable. The EDH scheme has been shown to be quite robust in difficult streaming problems, although further testing is warranted before specific conclusions can be reached.

The EDA scheme is an early hybrid scheme used in developmental testing and analysis, and is <u>not</u> recommended for use, and therefore should be avoided.

2.5 Metrics and Adaptive Differencing

The DD, DZ, DTW, EDW, EDH, and EDA differencing schemes are fully implemented into *PENTRAN*. The DD and EDA schemes are available not recommended and are only included for test purposes! Differencing metrics reported for each coarse mesh are as follows (note: /sweep = "per angular flux sweep"):

- DD: no metric is used (not recommended)
- DZ: the average number of fixups/sweep (**adaptive level 1**)
- DTW: the maximum average weighting factor /sweep (**adaptive level 2**)
- EDW: the number of applications of DTW/sweep alone due to under/over-flow (**adaptive level 3**)
- EDH: the number of applications of DTW/sweep, **selectable & adaptive for each ordinate**
- EDA: the number of applications of DTW/sweep due to zero incident fluxes (**not recommended**)

The differencing metrics provide the user with useful information about the relative accuracy of the differencing in each coarse mesh cell. *PENTRAN* allows the user to take advantage of an adaptive differencing capability using DZ, DTW, or EDW to remove some of the difficulty in determining the appropriate scheme within a particular coarse mesh. Clearly, zones containing strong sources will not likely require many (if any) DZ fixups, although this depends on the cross sections, mesh size, and parallel decomposition strategy used. While *PENTRAN* allows the user to restrict ("lock-in") any differencing scheme in a particular coarse mesh, and the use of a specific algorithm is not a strict requirement, *adaptive differencing* can be selected for any coarse mesh.

The *adaptive differencing* strategy in *PENTRAN* works in the following manner: assume (for illustration) that the DZ scheme is initially assigned (but not locked) in each coarse mesh. In *PENTRAN*, an automatic transfer from DZ to DTW takes place if a negative flux fixup is encountered, followed later by another transfer to EDW if a maximum weight factor beyond a user specified maximum weight (typically set to 0.96) is detected for DTW within a coarse mesh. A shift from DTW to EDW can occur in a strong source region if the flux is relatively flat (low gradient), where a step scheme would be ideal (causing higher weights in DTW). Since EDW would not be practical in these situations, forcing DTW in regions with low flux gradients may be warranted.

If parallel decomposition is used with adaptive differencing, a synchronization is made among the processors working on a particular coarse mesh to upgrade to the *same* adaptive differencing scheme, even if an upgrade is not required by *all* processors. This lends the adaptive procedure to a degree of numerical consistency, so that the user is certain of the differencing algorithm rendering a solution in a particular coarse mesh.

Note that in the case of a fuel pin bundle immersed in water, the flux gradients could be steep, and allowing omega-selected EDH differencing may be best. This is because the differencing (DTW or EDW) is adaptively selected for each discrete ordinate, and with no restrictions or processor synchronizations of the differencing algorithm. We note that the EDH algorithm is still being tested, although to date, results have been promising given that this scheme implements a truly adaptive approach for each direction.

2.6 Angular Flux Moments and Boundary Conditions

Regardless of the differencing method used, following an update of the group source and angular flux transport sweep through each cell, cell group scalar, cosine, and sine flux moments are updated using the latest cell average iterates. Therefore, equations (2.52), are based on the cell averaged angular flux in equation (2.33), and are updated using the following quadrature expressions (with an implicit group *g* subscript):

(2.52) $$\phi_l = \frac{1}{8}\sum_{m=1}^{M} w_m \psi_{A\,m} P_l(\mu_m)$$

$$\phi^k_{C\,l} = \frac{1}{8}\sum_{m=1}^{M} w_m \psi_{A\,m} P^k_l(\mu_m)\cos(k\varphi_m)$$

$$\phi^k_{S\,l} = \frac{1}{8}\sum_{m=1}^{M} w_m \psi_{A\,m} P^k_l(\mu_m)\sin(k\varphi_m)$$

Note that these expressions include the 1/8 term from the criteria that quadrature weights were derived summing to one in each octant. Azimuthal angles are determined from:

(2.53) $$\varphi_{m0} = \arctan\left(\frac{\xi_m}{\eta_m}\right)$$

where care must be taken to obtain the correct phase of the angle on the unit sphere. If $\xi_m < 0,\ \eta_m < 0,$ then $\varphi_m = (\varphi_{m0} - \pi)$. If $\xi_m > 0,\ \eta_m < 0,$ then $\varphi_m = (\varphi_{m0} + \pi),$ otherwise, $\varphi_m = \varphi_{m0}$.

Standard boundary conditions include specular reflective, albedo, and vacuum (zero return current) boundaries. For albedo boundaries, entering angular fluxes are reflected from surfaces with normal vectors parallel to the *x* axis, are, with $\widehat{\Omega}_m = \langle \mu_m, \eta_m, \xi_m \rangle$:

(2.54) $$\psi_{xBdy}(\widehat{\Omega}_m) = a\ \psi_{xBdy}(\widehat{\Omega}'_m)$$ where $\widehat{\Omega}_m \cdot \hat{i} = -\widehat{\Omega}'_m \cdot \hat{i}$ and $\widehat{\Omega}'_m = \langle -\mu_m, \eta_m, \xi_m \rangle$

Similar formulations hold for angular fluxes reflected from a surface normal to the *y*-axis with $\widehat{\Omega}'_m = \langle \mu_m, -\eta_m, \xi_m \rangle$ and normal to the *z*-axis with $\widehat{\Omega}'_m = \langle \mu_m, \eta_m, -\xi_m \rangle$. In each case, the albedo factor a can be energy group dependent, and is equal to unity if the boundary is fully reflective, or equal to zero if the boundary is a vacuum.

2.7 Acceleration Schemes: Rebalancing

PENTRAN contains options for standard rebalance methods, as well as for a simplified multigrid acceleration scheme. Each iterative acceleration schemes is compatible with any of the available differencing schemes with no special treatment, and they can be used simultaneously to accelerate the iteration process. Coarse mesh rebalancing involves requiring the integral balance equation to hold for each energy group for each coarse mesh (Reed, 1971). The integral balance equation for each group, after applying Gauss' theorem to the streaming divergence, is

$$\int_A J \cdot dA + \int_V (\sigma - \sigma_s)\phi_0 \, dV = \int_V (Q_{S\,(D,U)} + \frac{Q_{fiss}}{k_o} \tag{2.55}$$

where J = leakage current across surfaces *A* bounding volume *V*
$\sigma - \sigma_s$ = within group removal cross section
ϕ_o = scalar flux
$Q_{S\,(D,U)}$ = down- and up-scattering source
$\frac{Q_{fiss}}{k_o}$ = volumetric fission source term
Q_{Ext} = volumetric external source term

Introducing rebalance factors f_{ijk} for each coarse mesh of volume V_{ijk} in an *x-y-z* gridspace for each energy group, and using partial currents normal to each respective surface, equation (2.56) is a general expression for rebalance for each coarse mesh. Note that surfaces are labeled with respect to each local coarse mesh cell:

$$\begin{aligned}
&f_{ijk}\,((Jx^-_{ijk}|A_{x\ In}) + (Jx^+_{ijk}|A_{x\ Out}) + (Jy^-_{ijk}|A_{y\ Right}) + (Jy^+_{ijk}|A_{y\ Left}) + \\
&(Jz^-_{ijk}|A_{z\ Bottom}) + (Jz^+_{ijk}|A_{z\ Top}) + (\sigma - \sigma_s)\phi_{o\ ijk}V_{ijk}) - f_{i-1,jk}(Jx^+_{i-1,jk}|A_{x\ Out}) - f_{i+1,jk}(Jx^-_{i+1,jk}|A_{x\ In}) \\
&- f_{i,\,j-1,k}(Jy^+_{i,\,j-1,k}|A_{y\ Left}) - f_{i,\,j+1,k}(Jy^-_{i,\,j+1,k}|A_{y\ Right}) - f_{ij,k-1}(Jz^+_{ij,k-1}|A_{z\ Top}) - f_{ij,k+1}(Jz^-_{ij,k+1}|A_{z\ Bottom}) \\
&= (Qs_{(D,U)\ ijk} + \frac{Q_{fiss}}{k_o}{}_{ijk} + Q_{Ext\ ijk})V_{ijk}
\end{aligned} \tag{2.56}$$

Simultaneous solution of the system of equations that result from Equation (2.56) for each coarse mesh in each energy group is required; this is performed in *PENTRAN* using a direct Cholesky-LU factorization algorithm. All rebalance factors are damped using restrictions equivalent to Partial Current Rebalance (Rhoades, 1981). Maximum rebalance factors are set by the worst imbalance, with a damping factor applied in further rebalance operations. To retain scalability and bound the memory required for this direct solution method, zoned rebalancing over subsets of coarse meshes is required if the number of coarse meshes exceeds the maximum rebalance matrix size, or if the user specifies a subset of contiguous zones over which to constrain particle balance following each source iteration. If all of the coarse cells in a zone are not local to the processor, rebalance for that zone is by-passed. This direct, zoned solution scheme for rebalance is necessary (as opposed to an iterative technique) to achieve adequate processor synchronization and minimize rebalance overhead.

2.8 Acceleration Schemes: Multigrid

Multigrid spatial acceleration methods essentially use coarse grid iterates projected to correct fine grid iterates, and are best described by transforming the spatial domain into the frequency domain (with units of inverse length) using the Fourier transform. High frequency errors present in successive fine grid iterates are readily reduced by each fine grid transport source iteration. However, the low frequency persistent errors present in the fine grid iterates become increasingly difficult to reduce with mesh grid refinement. In comparing a fine grid to a coarse grid formulation of a finite difference problem, the number of possible Fourier frequencies correspond to the number of equations in the linear system. Since this depends on the stepsize used in the problem, a larger mesh step size will limit the number of possible Fourier modes (frequencies). The highest frequency error components at a point on a coarse mesh will typically correspond to low or mid-range frequency error components for that same point on a fine mesh. Therefore, if a value from one iteration on the coarse grid were projected to correct the fine grid solution, the correction should be greater (and most often is significantly greater) than a single fine grid iteration by itself. This is because the low frequency errors in a fine grid problem are most effectively reduced by using a correction from a coarse grid solution. Therefore, multigrid methods effectively increase the rate of convergence in the iterative scheme. Because the asymptotic rate of convergence is the negative logarithm of the spectral radius of the linear system, use of a multigrid procedure effectively reduces the spectral radius in comparison to the spectral radius of the single fine grid system. This is the basis of the multigrid (also called multi-level) approach.

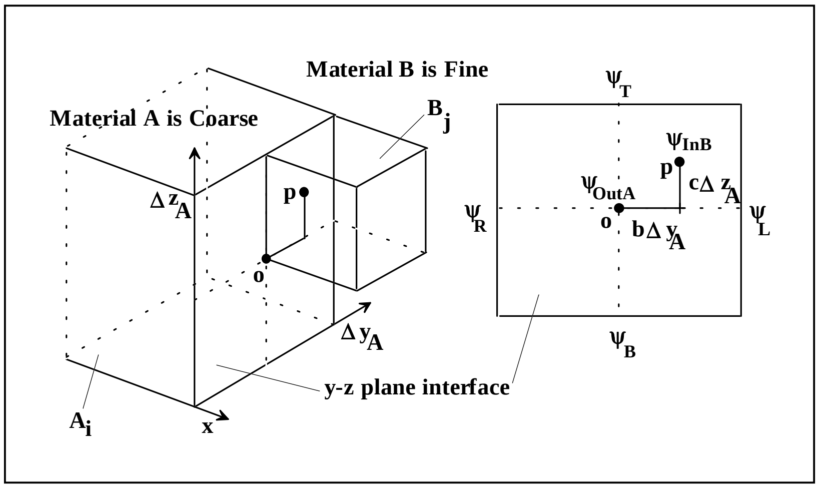
For transport applications, Nowak, Larsen, and Martin (1987) demonstrated via Fourier analysis that if a mesh size is doubled (for computing on coarser mesh with a high scattering ratio), the number of iterations required to reduce the *high frequency error* by a factor of 10 is ***doubled*** (demonstrated using a weighted diamond scheme). At the same time, in three dimensions, the ***number of cells is reduced by 1/8*** (a factor of two along each axis). ***Therefore, the overall work required on the coarser mesh is 25% of that on the fine mesh.*** Nowak, et. al. also observed that if the scattering ratio is small, multigrid methods will not be as beneficial, since the transport operator effectively reduces the iterative error. This is expected, since the spectral radius of the difference equation is directly proportional to the scattering ratio. From this, it is readily inferred that problems that are less elliptic (with lower scattering ratios) are dominated by errors that require a high frequency ***local*** iterative correction, and are therefore solved readily by the transport operator on the fine grid. Alternatively, problems with high scattering ratios are elliptic in nature and are dominated by a ***global*** low frequency error; these problems benefit the most from a low frequency iterative error correction provided by a multigrid solution.

Overall, the key to obtaining accelerated convergence in a multigrid scheme is to use a powerful (yet computationally inexpensive) equation for a coarse grid iteration step. Yet due to the immense memory requirements demanded by multigroup parallel transport methods, there is a need to conserve the memory demanded per processor. To capture some of the beneficial acceleration effects of a multigrid scheme and still conserve memory, the *PENTRAN* code iterates to a prescribed tolerance on the medium grid, and then overwrites medium grid angular flux values as they are projected onto the fine grid. In this way, the same arrays are used to store both grids. Recall that angular fluxes are partitioned in memory local to each processor in *PENTRAN*, so that only the portion of the phase space assigned to each processor is stored. In spite of this, use of ***two*** transport grids would add to memory overhead, and increase the minimum processor pool required to solve a large problem.

Therefore, the medium grid angular fluxes are lost after they are converged to a suitable tolerance and projected onto the fine grid using 3-D Taylor Projection Mesh Coupling (*TPMC*; this is presented in the next section). Using this procedure, the low frequency error on the fine grid can be largely eliminated, causing the solution to be dramatically accelerated. This scheme of projecting converged coarse grid values to the fine grid with TPMC is defined here as a "Simplified" Multigrid approach, and differs from more conventional multigrid methods that use residual error corrections between grids. Using this multigrid scheme in *PENTRAN*, test problems with high scattering ratios have demonstrated convergence with as much as an order of magnitude fewer iterations compared to the same problem solved using a single grid iteration sequence (Sjoden and Haghighat, 1996).

2.9 Taylor Projection Mesh Coupling

In 3-D discrete ordinates (S_N) transport methods, it is desirable (and at times necessary) to use discontinuous spatial grids in coarse cells or at material interfaces. On a single processor machine, variable meshing permits high definition in regions of interest, with coarse grids used in less important regions. In parallel processing, variable meshing can be used to reduce load imbalances in problems spatially decomposed over large processor arrays.



Trasnfer from "A" Coarse to "B" Fine Meshes, where *I*=1 and *J*=4

In a typical S_N iterative solution scheme, a source iteration is performed, followed by sweeps in angular flux for each discrete ordinate using a spatial differencing scheme. As the sweeps progress through the various spatial grids in a problem, particle conservation must be maintained at any discontinuous grid interfaces. Typically, this is accomplished using a simple balance of particles streaming across a boundary surface, resulting in a zeroth order approximation. This procedure works well in the case of sweeps progressing from fine grids to coarse grids; however, a loss of information occurs as angular sweeps progress from coarse grids to fine grids.

PENTRAN uses a new *Taylor Projection Mesh Coupling* (*TPMC*) scheme for an *x-y-z* S_N method that attempts to mitigate the loss of angular flux information as sweeps are made from coarse to fine grid interfaces. Note that the relationship between surface and cell averaged angular fluxes is determined using a spatial differencing scheme, as already discussed. In the figure above, consider a transfer of angular fluxes from the "A" boundary to the "B" boundary, crossing the (*y-z*) plane in the +*x* sweep direction. The mismatch between the cells results in a discontinuity that must be resolved while sweeping along angles.

To reduce the loss of information that occurs when a transfer of boundary fluxes is made from a coarser -to-finer grid pitch, a Taylor Projected Mesh Coupling of surface angular fluxes is proposed. This *TPMC* scheme amounts to using the coarser A surface angular fluxes to approximate partial derivatives in a truncated Taylor series expansion; the expansion is used to project surface angular fluxes exiting A into B. Taylor interpolation methods have been employed in computational fluids and heat applications, but often with difficulty in maintaining strict conservation (Kallinderis, 1992, and Phillips and Schmidt, 1984).

The first step in the TPMC scheme involves the direct transfer of the angular fluxes exiting cell A at point *o*, ψ_{OutA}, to the angular flux entering a B cell at point *p*, which is ψ_{InB}. Using a truncated Taylor expansion from surface centers connected between A and B:

$$\psi_{\text{inB}} = \psi_{\text{outA}} + b\,\Delta y_A \frac{\partial \psi}{\partial y}|_A + c\Delta z_A \frac{\partial \psi}{\partial z}|_A + O(\Delta^2) \tag{2.57}$$

With partial derivatives approximated by central differences computed from A boundary values:

(2.58) $$\left.\frac{\partial\psi}{\partial y}\right|_A = \frac{(\psi_L - \psi_R)}{\Delta y_A} + O(\Delta^2) \quad \text{and} \quad \left.\frac{\partial\psi}{\partial z}\right|_A = \frac{(\psi_T - \psi_B)}{\Delta z_A} + O(\Delta^2)$$

Note that the scale factors *b* and *c* are the relative fractions of cell A width connecting surface centers along the *y* and *z* axes, respectively. Combining equations (71) and (72) results in

(2.59) $$\psi_{InB} = \psi_{OutA} + b(\psi_L - \psi_R) + c(\psi_T - \psi_B)$$

Equation (2.59) permits a simple first order projection of angular boundary fluxes exiting from the coarser A cell entering into the finer B cells. During a transport sweep for a given angular ordinate, this process is repeated for all B cell boundaries. If a flux projection yields negative values, the *absolute value* of the *most negative* angular flux projected among the B surfaces is then added into each B surface flux, causing the minimum projected flux to be zero. Because particle conservation during the transfers from *I* cells of A to *J* cells of B must be conserved, normalized *projection fractions* f_j for each B surface angular flux are then computed by normalizing all values projected (in Figure 4, *I*=1 and *J*=4):

(2.60) $$f_j = \left(\frac{\psi_{InB_j}}{\sum_{j=1}^{J} \psi_{InB_j}} \right)$$

Particle balance is achieved by first computing the particle outflow from the surface of A:

(2.61) $$flow_A = \sum_{i=1}^{I} (\psi_{OutA_i})\, \mu_m\, (\Delta y_{A_i}\, \Delta z_{A_i})$$

Then, using equations (2.60) and (2.61), the *final TPMC* projected angular fluxes are obtained from

(2.62) $$\psi_{InB_j} = \frac{f_j \cdot flow_A}{\mu_m\, \Delta y_{B_j}\, \Delta z_{B_j}}$$

Note that a traditional, *zeroth order* projection is obtained by setting the scaling factors *b* and *c* to *zero*. In the event that grids do not directly "match," nearest neighboring cell centers are used to project angular fluxes. Based on test problems, the *TPMC* scheme provides fine grid fluxes that are 3 times or more accurate than traditional zeroth-order methods in cases where a finer meshed region of interest (ROI) is surrounded by coarser meshes. Therefore, *TPMC* as a minimum mitigates some effects of using coarser grids in cells surrounding a ROI. In more realistic problems with steep gradients, the differences between *TPMC* and zeroth-order schemes are more pronounced. Accuracy of fine mesh fluxes have shown a factor of three improvement in the Kobayashi problems using TPMC as opposed to zeroth order coupling in problems using discontinuous grids (Sjoden and Haghighat, 2000).

2.10 Criticality Eigenvalue Determination

An algorithm to determine the criticality eigenvalue of a system, consistent with complete parallel phase space decomposition, is in place in *PENTRAN*. Several benchmarks have been performed for various small multigroup problems to verify the iteration procedure. The existence of the criticality eigenvector is mathematically derived from Perron's theorem, which states that a positive matrix has a unique positive eigenvector with a single positive eigenvalue that is greater than the modulus of any other eigenvalue for the matrix. In reactor physics applications, the criticality eigenvalue represents the fundamental effective multiplication factor k_o that dominates the the nuclear system after all higher harmonic transients have died away.

In the code, the outer iteration fission source is based on the previously converged inner iteration flux estimate. The fundamental (criticality) eigenvalue k_o is determined from a variation of the Damped Power-Aitken Wielandt eigenvalue iteration, where a *correction* to the eigenvalue δk_o is made based on an Aitken extrapolation (used successively after each 3rd iteration) and the relative difference between successive eigenvalue iterates (Nakamura, 1977). Note that due to their added parallel storage and synchronization requirements, Chebyshev acceleration methods were not considered. In any one outer iteration in *PENTRAN*, the criticality eigenvalue estimate for iteration *t* is updated using the following procedure on each processor, with the most recent fission source estimate Q_{fiss}^t (based on the most recent scalar flux) used as the arbitrary power iteration weighting vector:

(2.63) $$\tilde{k}_o^{\,t} = k_o^{\,t-1}\frac{\left\langle Q_{fiss}^t, Q_{fiss}^t\right\rangle}{\left\langle Q_{fiss}^{t-1}, Q_{fiss}^t\right\rangle}$$

The Aitken extrapolation is then performed after at least 3 successive iterates are available, where

(2.64) $$\tilde{k}_o^{\,t} = \tilde{k}_o^{\,t-1} - \frac{(\tilde{k}_o^{\,t-1} - \tilde{k}_o^{\,t-2})^2}{(\tilde{k}_o^{\,t-3} - 2\tilde{k}_o^{\,t-2} + \tilde{k}_o^{\,t-1})}$$

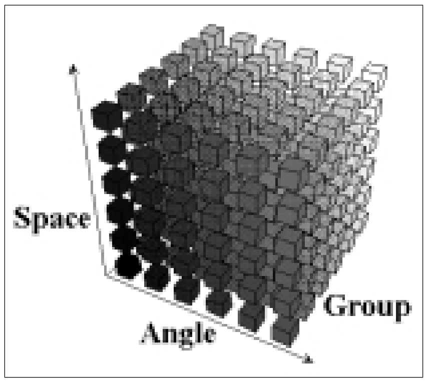
(2.65) $$k_o^t = k_o^{t-1} + \delta k_o^{t-1} \quad \text{where} \quad \delta k_o^{t-1} = C(\tilde{k}_o^{\,t} - k_o^{t-1})$$

The *C* in equation (2.65) is a factor restricted to [0.5,1], and is based on the relative error between successive eigenvalue iterates, increasing to unity as the relative error between iterates decreases below specific settings. This scheme is effectively a Damped Power-Aitken -Wielandt scheme, allowing convergence to the correct criticality eigenvalue (based on numerical testing), even if the initial guess is poor.

2.11 Automated Phase Space Decomposition Scheduling

The key to the parallel scalability of the *PENTRAN* code lies in its internal structure. A transport problem input deck is read and processed by each processor. Following several initialization sequences, the angles, energy groups, and spatial cells in the problem are automatically decomposed according to a user-specified *decomposition weighting vector*. This decomposition vector contains arbitrary weighting factors for angular, group, and spatial decomposition. This vector allows a user to "prioritize" the decomposition strategy used in a parallel execution of the problem *without* specifically assigning an exact number of processors (a user can also block decomposition in a particular variable, or lock-in any specific number of processors, if desired).

Therefore, the *PENTRAN* code auto-schedules the decomposition of the problem for the user at execution time onto *n* processors, *and self adjusts* the number of processes assigned for the problem being solved. However, this is performed within the restrictions allowed by the user's decomposition weights. If the weighting scheme specified by the user leads to an *odd number of processes* scheduled on an *even number of processors*, a halt statement for "processor utilization below 100%" is issued, with a message to the user to change the decomposition weights or the allocated number of processors. The number of *processes* must be greater than or equal to and divisible by the number of *processors*. Ultimately, the auto scheduling in *PENTRAN* leads to a 3-D Cartesian, virtual processor array topology, with angular, group, and spatial decomposition axes. The phase space of the problem is essentially projected onto this virtual processor topology during parallel execution.



3-D Cartesian Virtual Processor Array

2.12 Advanced Processor Communications

Efficient communications over an arbitrary domain decomposition are obtained from the use of complex inter-process communicator buffers (or *communicators* as they are called in *MPI*). These communicators are generated on each process during the distribution of problem data over the virtual processor array completely transparent to the user. They enable communications among specific subgroups of processors in the virtual topology. For example, consider a scenario where angles, groups, and sweeps are fully decomposed on a processor array. If a scalar flux is required for all groups in a particular coarse mesh cell, the reduction operation to collect all of the angular fluxes for quadrature in their respective groups uses a communicator that links processors "owning" the groups and angles for that specific coarse cell. *PENTRAN* builds communicators to selectively communicate with selected processors containing:

- All angles for a specific coarse cell and energy group
- All energy groups and angles for a specific coarse cell
- All coarse cells and angles for a specific energy group

On the *SP2*, an upper limit of 2000 communicators exists in the 1996 IBM MPI-product implementation. Other architectures have various limits on the number of possible unique communicators. To retain scalability, *PENTRAN* will automatically minimize the number of unique communicators "built" during each problem execution, re-using communicator buffers if they contact the same set of processors. Any excess communicator buffers that are not useable by the processor (null value) are immediately released back to the available *MPI* communicator buffer pool. Communications overhead is further minimized in *PENTRAN* by array packing routines. All array data transferred among processors is packed by the sending processor, then unpacked by the receiving processor after communications are completed.

2.13 Process Mapping Arrays and Dynamic Memory Allocation

Following the assignment of the processors and communicators in the virtual processor array, phase space variables assigned to a particular processor are tracked and computed independently. Local variable dimensions (when practical) and *process mapping arrays* are used on each processor. This allows all memory to be completely partitioned on each processor for all medium (and fine) mesh variables, including memory intensive angular, scalar flux, and current (*J*) variables. Therefore, *process mapping arrays* contain the global coarse mesh cell, energy group, angular sweep octant, and octant ordinate ($\hat{\Omega}$) indexes that are locally processed in locally dimensioned arrays.

A global *processor* ***reference*** *mapping array* (kpmap) is also maintained by each processor (allocated identically on each processor during problem setup), and permits access to the *processor number* containing any coarse cell, group, sweep octant, and octant ordinate. This permits "send" and "receive" processors to be readily identified if point to point message passing is required, as occurs in the case of reflective boundaries with angular decomposition, or with spatial decomposition. When message passing is performed, global index references are used during communications among processors. After messages are received, global variable array indexes are translated back to local indexes for storage. Further, messages and various communications are continuously cross-checked using the (kpmap) *processor* ***reference*** *mapping array,* the *process mapping arrays,* and the translation routine to verify data integrity. Because the *process mapping arrays* store global angular, energy, and spatial variable indexes for locally stored variables, any schemes that require specific ordering can be readily accomplished. Therefore, rearranging the order that global variable indexes are assigned and stored in the *process mapping arrays* alters *where and in what order* they are locally processed.

The bulk of memory (per processor) in *PENTRAN* is consumed by storing the cell centered and surface angular fluxes. Grid arrays only need be as large as required for the largest spatial grid in a single coarse mesh cell, the total number of *local* energy groups, and the total number of *local* angular sweep octants for the problem being solved. Therefore, larger problems can be solved directly in *PENTRAN* by adding more processors with increased decomposition. The amount of memory, in bytes, consumed by each processor, is roughly approximated by

(2.66) Memory Bytes = λ maxcmc maxfmc maxglc maxswp maxqdm

where the parameters represent a scaling constant λ, the maximum local coarse meshes, fine meshes per local coarse mesh, local groups, local octant sweeps, and angles per octant (bound by angular quadrature method), respectively, for a processor. From experience, 1-2% of processor memory should be reserved for system level processes, caching, etc, depending on the machine; therefore, the left hand side of equation (2.66) should be regarded as “free” memory per processor. The constant λ is typically between 50 and 60 for the single precision version of *PENTRAN,* depending on the compiler. Note Equation (2.66) assumes that the group window or restart options (which can provide a significant reduction of memory requirements) are *not used.*

The amount of memory required for a specific calculation is accurately computed by the code when the parameters at the top of an input deck are provided. The memory required and is compared to the maxmem parameter to determine if the memory required exceeds the available memory. Therefore, knowing the memory bound per node, one can set parameters to cover a general problem type, and determine how many processors are required for a problem, depending on the decomposition approach used.

NOTE: The user should avoid setting some parameters to values significantly greater than required for a problem, since parallel efficiency can be directly affected. For example, the following parameters directly affect message length (and therefore message passing time): maxleg, maxgrp, maxfmc, and maxqdm. The first two parameters are the maximum Legendre moment required and the maximum global number of energy groups required, respectively. Since it is conceivable that a heterogeneous parallel cluster could be used to run *PENTRAN* under MPI, parameters should be set to allow the code to fit within the total memory on the *minimum capacity machine*, since the same code must run on all machines simultaneously, albeit independently. Note that all important parameter settings and total memory demand are also provided in the run logfile for inspection by the user.

Some variables in *PENTRAN* are dimensioned using an integral memory approach for efficiency, as they would provide a small memory savings, yet add appreciable message passing overhead. For example, group coarse grid data and coarse cell spatial mapping data are stored using global problem dimensions on each processor. Partitioned memory practices are applied when the memory savings potential is large, as in the case of medium/fine mesh scalar and angular fluxes. Overall, due to a partitioned memory structure, the *PENTRAN* code has true data parallelism for memory intensive arrays (the angular and scalar fluxes), and is therefore a fully scalable code. Larger problems require more processors, although a capability of complete phase space decomposition in the angular, energy, and spatial variables with discontinuous meshing offers great flexibility in how subdomains are created to render a parallel transport solution. The most efficient parallel decomposition scheme is, however, different for each problem, and is difficult to anticipate. The user should pay attention to the following rules in solving a problem with *PENTRAN* in parallel:

- Adjust parameters to determine how many processors are needed as a minimum to solve a problem based on the total amount of required memory per processor. (Setting maxmem=1 (Mb) and executing will report the total per processor memory required, and then halt the code if the demands exceed 1 Mb. This is a convenient approach the user may want to take when investigating the memory required for different decomposition strategies, and can be performed running any version of the code, including single processor versions).

- Be aware of parameter limits. For example, if always using 2 processors for angular decomposition (which is good to do since it does not degrade convergence in Cartesian geometry), the maxswp parameter can be set to 4 rather than 8, since a maximum of 4 sweep octants will be processed on each machine. Also note that the number of medium meshes (maxmmc) should always be set equal to the number of fine meshes (maxfmc) per coarse mesh, as currently required for the Simpified Multigrid method.

- Note that coarse mesh boundaries define subdomains for parallel spatial decomposition, acceleration methods (rebalance and multigrid), and assignment of the numerical differencing scheme used.

- When possible, avoid using small numbers of meshes inside each coarse mesh, since a processor synchronization is required after each processor completes a coarse mesh (if reflective boundaries and/or spatial decomposition is used). Use at least 25 fine meshes/coarse mesh; however, more is better--this makess the calculation increasingly coarse grained.

- TPMC is used on each surface of each coarse mesh connected to other coarse meshes. It is advisable to use discontinuous meshing where needed, since TPMC is always being paid for.

- Use of angular and spatial (or both) decomposition can be beneficial to rebalance acceleration, which can offer a significant increase in convergence (where it would not on *one* processor). This is due to ADS sweep ordering/partitioning, and will often offset the convergence penalty incurred from spatial decomposition. DZ may incur more negative fluxes with increased spatial decomposition, and can exhibit non-convergence in parallel executions. In recent studies with optically thick problems, better performance was obtained by starting with DTW differencing as a first scheme in the adaptive algorithm, followed by application of EDW (by *PENTRAN*) as needed.

• Avoid setting the maximum allowable DTW weight (using the dtwmxw setting in Block 4) too low--a premature shift to EDW could degrade convergence in source regions, since the EDW scheme performs best in streaming cases. Still, if the maximum weight is still high and a shift is made, it is clear that EDW is required. A shift from DTW to EDW can occur in a strong source region if the flux is relatively flat (low gradient), where a step scheme would be ideal (causing the higher weights in DTW). Since EDW would not be practical, forcing DTW (by using a -2 rather than a 2 in **ndmeth**) in these regions, or a specification of EDH may be warranted. Don't forget that meshing can be locked in *or* upgradable within a coarse mesh, depending on the scheme used.

• If not using group decomposition, use the standard multigroup iteration method by setting **methit=1**. This method allows for the most efficient use of particle downscattering by continuously accumulating the scattering source through groups *g-1* during convergence of group *g*, which can be a great savings when performing a P_3 calculation. With complete group decomposition, the Hiromoto-Wienke 1-level scheme may be best, selected with **methit=2.**

• Be careful in using group decomposition on *few group* problems if the scattering ratios vary greatly. Since few group problems are strongly coupled, one processor could tie up the rest with many iterations in a group with a high scattering ratio. This effect may be ameliorated by activating automatic load balancing and/or multigrid acceleration, although at the same time, load balancing might inhibit convergence.

• Use of widely varying medium and fine grids can hinder the multigrid effect--a ratio of two fine/medium mesh along each axis is generally recommended. In cases where multigrid is not as effective (e.g. low scattering ratios), use a low medium grid tolerance just to obtain a good first guess on the fine grid. Also, medium grid tolerance beyond the truncation error of the differencing will not contribute to convergence.

• In fixed source problems with multigrid, the user is responsible for properly defining the source, including the total number of source particles, on each grid.

• Be aware of disk storage requirements. Scratch files for FIDO input and material mesh specification will typically require 5 bytes of disk space per mesh per processor; typically, this can be stored in /tmp space on each processor. A typical run with binary scalar flux outputs can require disk space for $O\{(\text{total \# of meshes}) * \text{maxgrp} * 60 \text{ Bytes}\}$ bytes.

• Also, exercise caution when selecting high scattering moment dumps or angular flux binary dumps--file space could be quickly saturated.

• *PENTRAN* should be executed in parallel to the greatest extent possible. Restricting the use of the code to a single machine (when not already constrained by memory due to storage requirements) forfeits the effort required to perform all of the complex mechanics necessary for complete phase space parallelization. Some of these tasks include problem decomposition and distribution routines, partitioned memory mapping, range checking, global-local and local-global mapping, and the overall coarse grid based, discontinuous mesh code structure. Although there are some logical by-passes, many tasks required for parallel execution in *PENTRAN* are also carried out in serial as well, and amount to what is considered *parallel overhead*. If parallel execution is not possible, it is likely more practical to solve the problem with another S_N code that is optimized for a single CPU. Alternatively, *PENTRAN* was developed with the intent of solving *very large* problems on a distributed parallel cluster *that could not be solved in a practical time or fit under the memory and/or disk constraints of a single CPU.*

.

2.14 Source Iteration Schemes

Two source iteration schemes are available: a "multigroup" source iteration (the GOFMGM routine) and a Hiromoto-Wienke source iteration (the GOFCHM routine).

The GOFMGM "multigroup" routine iterates to convergence sequentially through each group, from group g_i to g_j, where $(i<j)$, with an added convergence confirmation of the local scattering source if upscattering or group decomposition is indicated. Progression to the next (local) group is not performed until convergence in the current group is obtained. The convergence of the scattering source is computed using a norm based on the relative change of the integral sum of the angular scattering source for all locally processed coarse mesh cells, groups and angles. A tolerance of one order of magnitude greater than the pointwise flux convergence tolerance is used to determine convergence on the scattering source (in the event of energy group decomposition). Convergence on the scattering source is checked following reported convergence in all groups. Parallel execution with any level of decomposition using the "multigroup" source iteration scheme is available without restriction.

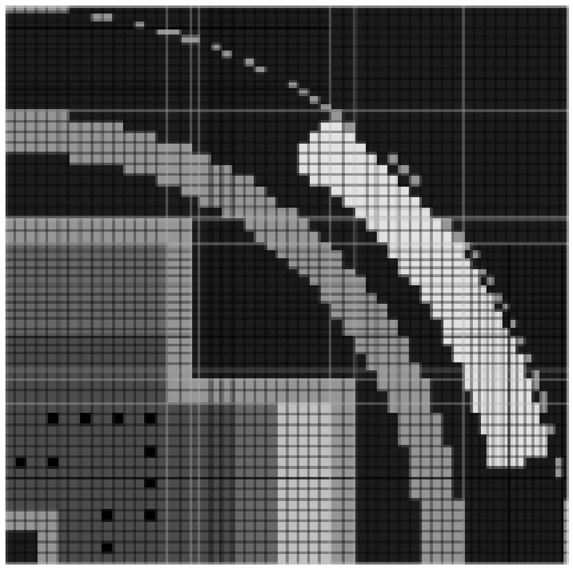
The GOFCHM routine is consistent with the one-level TPCS chaotic scheme of Hiromoto and Wienke (1989), and can be used with any phase space decomposition strategy in *PENTRAN*. Excellent parallel speedups have been demonstrated using various combinations of decomposition in the angular, energy, and spatial variables using this scheme (Sjoden and Haghighat, 1996). In this Hiromoto-Wienke scheme, each processor completes a source iteration in a *single pass* through a number of *locally* processed groups. Group flux moments are updated by a summing reduction among participating processors for the next scattering source calculation, but convergence is tested only on locally processed groups; this is followed by a new iteration, if needed. On a *shared memory* machine, this iteration procedure is *truly* chaotic, as group flux moments are updated for all processors instantaneously, resulting in chaotic convergence among the various groups scheduled on different processors. That said, on a *distributed* memory architecture, like the *SP2*, this iteration scheme is "chaotic" *in name only* since the group flux moment reduction is required following a sweep, which effectively acts as a processor synchronization. However, since convergence is tested based on locally processed groups, the user is able to identify the relative speed of convergence in each group, as *PENTRAN* reports when convergence is achieved on each processor. Computation for all processors must continue, however, until all tasks are converged, as message deadlocks will occur if one participating processor is deliberately stalled.

In general, if group decomposition is *not* used, the multigroup scheme will provide for faster convergence to a solution, with some exceptions. First, if there is upscattering, tests have demonstrated that the Hiromoto-Wienke scheme works best, since all local groups are iterated through sequentially, updating the upscatter components continuously. Also, faster convergence with the Hiromoto-Wienke scheme resulted when the simplified multigrid acceleration was used for any type of problem (Sjoden and Haghighat, 1996). As more users apply the code to a variety of test problems, this contributes to the experience and bounds the efficiencies of the code.

2.15 Benchmarking and Parallel Performance

Several problem benchmarks computed for one, two, and three dimensional, multigroup, multi-region, anisotropic problems have been tested using *PENTRAN* with DZ and DTW differencing and parallel domain decomposition. In particular, exact agreement (within the convergence tolerance) was obtained between solutions from the *TWOTRAN-II* , *TWODANT*, *DORT/TORT*, and *THREEDANT* single CPU production codes and parallel *PENTRAN* solutions. Similar agreement was demonstrated for criticality and adjoint test problems compared with *TWOTRAN-II*. An independent criticality benchamark using Hansen-Roach cross sections was performed with PENTRAN against MCNP in multigroup mode. The criticality eigenvalues rendered by each were the same within the limits of the cross sections and problem data. When comparing with the two dimensional codes tested, *PENTRAN* calculations were performed in 3-D with reflective boundary conditions along one of the three (*x-y-z*) axes. With single variable and hybrid phase space decomposition strategies, 3-D test problems were solved using dedicated timing benchmarks using the Cornell Theory Center IBM-SP2. Significant speedups were demonstrated, with high estimates of parallel code fraction (between 95% and 98%, based on Amdahl's Law).

An experimental *PENTRAN* benchmark modeling the complete Venus-3 reactor in 3-D has also been performed. The Venus-3 reactor is owned and operated by SCK-CEN nuclear energy research laboratory, located in Mol, Belgium. This facility houses Venus-3, a zero power research reactor designed to test partial length fuel assemblies and various test fuels. The core includes sixteen "15x15" sub- assemblies (as opposed to the typical "17x17" type). Although each assembly rack has fewer pins, the pin-to-pin lattice pitch is 1.26 cm--- resembling a conventional assembly. Therefore the Venus-3 facility serves as a practical model of a PWR reactor. Agreement with experimental flux measurements in the Venus-3 facility was excellent. Comparing 370 experimental reaction rate measurements from nickel, indium, and aluminum dosimeters, 95% of the calculated-to-experiment (C/E) values were within +/-10%, with the remaining 5% to within +/-15%. Again, please consult the literature for specific test results. For the Venus-3 model, quite good agreement was achieved between computational and experimental reaction rates, while achieving parallel efficiencies between 80% and 90%. The exact performance observed depended on the decomposition strategy and the number of processors used for the calculation. A cross section cut of z-level 2 through the Venus-3 reactor model (generated by *PENMSH*) is provided below:



Venus-3 Z-Level 2 rendered by *PENMSH*

3. *PENTRAN* Input

3.1 *PENTRAN* Input Processing

All input to the *PENTRAN* code is performed using an input file read and processed by each independent processor. The filename is always input by the user on processor 1, and if the execution is parallel, processor 1 will broadcast the input filename to all other processors. (Due to constraints on some parallel file systems, the name of the input deck is fixed to 'prb.pen'). Parallel input is reported to be the fastest means of initializing problem data on distributed memory machines, and typically involves far less overhead than processing the input data on one process with message passing data to all other processors (Gerner, 1995). In the latter case, since no other computations can be performed, all processes must pay for CPU time waiting for the source process to finish, in addition to the latency penalty and bandwidth limitations of message passing. To avoid these difficulties, input processing by each independent processor is the only input option, *but is transparent to the user.*

To accomplish parallel *FIDO* data input on each processor that is completely consistent with *FORTRAN* character/numeric data restrictions, a small (typically 5-50 kb) scratch file is created. To avoid file I/O conflicts in a distributed file system, a uniquely named scratch file is generated and used independently by each processor; the scratch file name (*fileprefix(1:5)*+ *processor#* +'.dat') is based on the processor number. (A uniquely *named* scratch file is necessary because a *FORTRAN* compiler may create a default *un-named* scratch file, 'fort.2,' for all processes, even during parallel execution). Although independently naming scratch files may not be necessary on all clustered workstation file systems, the current treatment prevents having to handle system-unique implementations, and prevents "fixes" that in some cases might violate strict *ANSI FORTRAN* programming. These *FIDO* scratch (.dat) files are only used for *FIDO* input processing and communicator minimization, and can be deleted after problem execution.

Similarly, another set of independent scratch files is created for material specification in each coarse mesh. All material scratch files (.mat) can be deleted following execution. On processor 1, the material file (containing data identical to other .mat files) is intended for archive and is stored using an input file prefix+'.M1'. The material file is required if processing geometry output from a geometry file in *Mathematica*. The material file can require ~5 bytes per mesh per processor.

Overall, the input deck is composed of :

```
PENTRAN CODE PARAMETERS FOR THIS PROBLEM
      :
---------------------Start Problem Deck----------------------
Problem Header card
Title Card 1
    :
Title Card 10
Block 1: General Problem Parameters  T
Block 2: Geometry  T
Block 3: Cross Section Parameters T
(Optional) Cross Sections (T required if xsecs within input deck)
Block 4: Control Options  T
Block 5: Sources  T
Block 6: Boundary Conditions  T
Block 7: Print Controls T
```

Outline of *PENTRAN* Parameters/Block Input

The input deck portion reserved for the parameter cards falls between the first line of each input deck, beginning with "PENTRAN CODE PARAMETERS FOR THIS PROBLEM." The parameter portion ends with with "-------------------Start Problem Deck--------------------". In reality, the labels are "dummy" lines and can contain ANY ASCII phrase, as are the lines that contain the names of the parameter labels that serve as a guide for the user.

3.2 Code Memory Input Parameters and *FIDO* Data Formats

The input deck for *PENTRAN* is designed in a block structure, emulating the block structure used in the *DANTSYS* code packages (O'Dell, et al, 1995). After the code parameters for establishing the memory on each processor are read in, subsequent input for each block is performed via the so-called free-field *FIDO* implementation, which includes control characters avialable for abbreviated input syntax. It is true that most of the issues surrounding the determination of correct parameter settings and *FIDO* commands are eliminated if using the *PENMSH* and *PENINP* automated mesh and input generation codes. However, should one wish to understand memory allocation issues, or tune an exisitng deck to implement a different decomposition, some discussion is necessary. In 1997, PENTRAN was modified to allow for adjustable-sized arrays, which avoids the issue of code recompilations for specific numbers of processors, etc, and permits more efficient use of dynamic memory. The following is an example of a parameter card set that should appear *at the very top of each PENTRAN input deck.* The user must enter the code parameters in a prescribed order for each parameter. Note that above each card image is a text "dummy" comment line naming the parameters below it. The "Code Parameters" section is the only part of the input deck that will *not* utilize FIDO notation.

Sample of Code Parameters at Top of Input Deck:

```
PENTRAN CODE PARAMETERS FOR THIS PROBLEM:
maxmem,  maxpcs,  maxgcm   maxxsg
80       8         4       0
maxcmc,  maxcrs,  maxmmc,  maxmed,  maxfmc,  maxfin
4        4        64       100      64      100
maxgrp,  maxglc,  maxswp,  maxqdm,  maxmat,  maxleg
2        2         8       10       1        3
maxsrc,  maxslc,  maxcmr,  maxlin,  maxarr,  nctlim
4        4        4        200      4000     20
-----------------Start Problem Deck-------------------
```

Desciption of Each Code Memory Parameter

maxmem	Maximum Process Memory, Mb	maxglc	Maximum Local Energy Groups
maxpcs	Maximum No. Procs Allowed For	maxswp	Maximum Local Sweep Octants
maxgcm	Maximum Global Coarse Meshes	maxqdm	Maximum Quadrature Angles/octant
maxxsg	Maximum Restart Total Groups	maxmat	Maximum Material (xsec)Types
maxcmc	Maximum Local Coarse Meshes	maxleg	Maximum Legendre Scattering Order
maxcrs	Maximum Coarse Meshes on an axis	maxsrc	Maximum Global Fixed Sources
maxmmc	Max Local Med Meshes/Coarse	maxslc	Maximum Local Fixed Sources
maxmed	Maximum Medium Meshes on axis	maxcmr	Max Contiguous CMR/PCR Cells
maxfmc	Max Local Fine Meshes/Coarse	maxlin	Maximum Lines in Input Deck
maxfin	Maximum Fine Meshes on an axis	maxarr	Maximum # Inputs in FIDO vector
maxgrp	Maximum Global Energy Groups	nctlim	Maximum FIDO Chars per Variable

Note that maxqdm can be determined from the order of the quadrature, where the maximum number of angles per octant maxqdm=(N(N+2)/8) for a given Sn order; e.g. S8 requires that maxqdm=10.

The *FIDO* syntax was originally implemented in several codes at the national laboratories. The following *general rules* apply for an input deck:

- 1 to 79 column line (card) images, no "special columns," no special ordering
- Data delimiters: blank space or comma
- All entries following a slash (/) are ignored
- Input is made according to the following syntax: *varname=number1 number2...* Named variables must be input *followed immediately by an "=" and the first entry in the array field, with no spaces between the "=" and the first array element*. *Any* spacing option can be used for *subsequent array elements* (with or without commas, intermixed slashes, etc). No spaces or commas should be used immediately following a *FIDO* control character (see table).
- No distinction is made between *real* and *integer* data, although *real entries used for integer fields are rounded.* Scientific notation should only use a lower case "e" for the exponent.
- A Block Terminator " T " is required at the end of each data block. The block terminator should immediately follow one or more spaces after (but always on the same line as) the last field data entry.

Summary of *FIDO* Array Control Characters

Control Character	*Syntax*	*Description*
R-Repeat	nRd	Repeat data d n times
I-Interpolate	nId d+1	Interpolate n items between d,d+1
C-sCale	nCd	Scale n previous items by d
F-Fill+	Fd	Fill the remainder of array with item d
Y-string repeat+	nYm	Repeat m strings n times
L-Log Interpolate	nLd d+1	Log Interpolate n items between d,d+1
Z-Zero	nZ	enter Zero n times
S-Skip++	nS	Skip the next n data items
A-pointer set	nA	set pointer to nth data item in array
Q-Q repeat	nQm	repeat the last m entries n times
G-G repeat	nGm	same as Q but change sign every repeat
N-N repeat	nNm	same as Q but invert order every repeat
M-M repeat	nMm	same as N but change sign every repeat
X-check entries	nX	check the number of entries against n
&-string skip+	&	skips to the end of the string

+ Identified but not Currently supported in *PENTRAN*
++ Data items that contain *FIDO* control characters count as a single entry

- The number of *FIDO* control characters in a given array field is typically set to a maximum of 200 (*nctlim=200*). If more than 200 *FIDO* characters per array field are required, the user should change this maximum to reflect the new *nctlim* value. (Note: the *PENMSH* code will provide a recommended upper limit).
- The number of lines in any input deck, including comments, is typically set to 1000 lines (*maxlin=1000).* If more than 1000 lines are required, the user should change this parameter to reflect the new *maxlin* value.
- *Interpreted FIDO* is output in a logfile, generated on one or more processors. If fatal errors are encountered, the cause should be identified in the logfile. (see the *loglevel* # phrase/option in the Header card of an input deck). *Note:* On some computers, we have observed that if a block appears to be correct but still encounters read errors/fatal errors, the user should add one (or more) spaces following a T block terminator. Also, block errors encoutered in a block that seems correct may be due to an improper or multiple parameter entry in the previous block. Read errors may also be encountered if input is specified beyond the 79 column limit in the input deck.

3.3 Problem Header Card and Title Cards

The problem header card is required, is a maximum of 79 columns, and serves as a problem label. Also, the header card serves a special purpose for *logfile* and *output file* control *when a* "loglevel" *string is present,* described below. The *logfile* is a file that displays *processed FIDO* input, as well as all warning and error messages, as-read cross section data, problem decomposition information, rebalance acceleration matrix solution data, processor iteration progress, and a process decomposition mapping table for all processes, cells, groups, and angles. The decomposition mapping table provides the user with an outline of where parallel output data is located, again depending on the automatic processor assignment. **Therefore, disabling all logfiles is not recommended.** A *logfile* can, depending on the optional "loglevel" string located somewhere in the header card, be generated on one or all processes; see the table below.

Summary of *loglevel* string in Header Card on Log/Output Files

loglevel string	*Effect*
loglevel 0	*Disables* all logfiles.
No loglevel string	Same as loglevel 1 *"Default"*
loglevel 1	*Enables* logfile on ***processor*** 1 *only*
loglevel 2	*Enables* logfiles on ***all processors***
loglevel 3	*Enables* logfile *and* prints CMR-PCR/SR factors, xsec reads on ***processor*** 1 *only*
loglevel 4	*Enables* logfile *and* prints CMR-PCR/SR factors, xsec reads on ***all processors***
loglevel all	*Enables* logfiles, CMR-PCR/SR factors, xsec reads, *and* data output on ***all processors***

Iteration progress will **only** be provided by a processor with an active logfile during execution, echoed to the terminal and written to the logfile. A **loglevel 4** string *forces **all** processors to print a log file* with CMR(PCR)/SR results (if rebalancing is activated). Furthermore, a **loglevel all** string is similar to **loglevel 4**, but *also forces **all** processors to print an output file.* The user should *exercise caution with these settings* when scaling to a large number of processors, as 3-D file outputs may be voluminous (especially with complete angular decomposition), possibly saturating all available disk space. Each log file is saved using the prefix of the input filename + '.L' + processor#; each output file is saved using the input filename prefix+'.'+processor#. For example: on processor 1 using input file *test.pen,* the logfile would be saved under *test.L1,* and the output file would be saved as *test.1.* More discussion on output is made under Block 7 (print options). On the *IBM-SP2,* processor numbers (alias task or rank numbers) are numbered starting from zero.

In the *PENTRAN* code, all processors are assumed to be numbered from 1 to n. Therefore, all machine rank identifications (e.g. resulting from an MPI_COMM_RANK call to identify the process rank) are shifted by +1 for *reporting purposes.* When referring to the process rank *within* MPI commands for actual message passing, reported ranks are converted back to machine ranks by adding -1).

There are 10 title cards required, 1-79 columns each, which can be used for problem description and documentation. If unused, these must still remain as the first 10 blank lines following the header card.

3.4 *PENTRAN* Input Blocks

Block I: General Problem Parameters

Block I includes general problem input parameters; all fields are required for code execution, and can be defined *in any order*. Twelve possible parameters in this block include: **ngeom, ngroup, isn, nmatl, ixcrs, jycrs, kzcrs, decmpv, lodbal, timcut, tolmgd, modadj.**

Summary of Block I Inputs

Variable	Decription	Syntax Example
ngeom	Geometry identifier	ngeom=3d
ngroup	Energy groups in calc/window/above restart	ngroup=*Grps, Win_Grps, Restart_Grps*
isn	Order of 3-D *level symmetric* quadrature (S2, S4, S6, S8, S10, S12, S14, S16, S18, S20)	isn=*quadrature_order*
nmatl	Number of materials in problem	nmatl=*Number_of_Materials*
ixcrs	Number of Coarse x mesh zones	ixcrs=*Number_of_CoarseX*
jycrs	Number of Coarse y mesh zones	jycrs=*Number_of_CoarseY*
kzcrs	Number of Coarse z mesh zones	kzcrs=*Number_of_CoarseZ*
decmpv	Decomposition vector	decmpv=*Angular_weight, Group_weight, Spatial Cell_weight*
lodbal	Automatic load balancing	lodbal=0 (off) or =1 (on)
timcut	Wall-Clock time cutoff limit, minutes	timcut=*Wall-clock_minutes*
tolmgd	Multigrid tolerance variable	tolmgd=*value*
modadj	Adjoint transport mode	modadj=0 (forward) or =1 (adjoint)

Termination of the Block with a T is required

Block I NOTES:

- The **ngeom** variable, at present, must always be =3d.
- Currently, the maximum allowed quadrature is S_8 (**isn**=8), although higher quadratures are in place, but commented out in the SETQAD routine. This is to reduce memory requirements for single processor testing. (note: the *maxqdm* parameter must be at least **isn*(isn+2)/8**)
- The **ixcrs, jycrs, kzcrs** variables are the number of coarse mesh cells projected along the x, y, and z axes, respectively, which yield (**ixcrs*jycrs*kzcrs**) total number of coarse mesh cells.
- The **decmpv** "decomposition vector" allows the user to specify the priority at which problem decomposition occurs for each variable, optimizing decomposition (within the restrictions of the assigned weights) during parallel execution. The input is an ordered triple of weight factors.
- The following special rules apply to **decmpv** weight factors:

1. A ***negative value*** over-rides *any* optimization and *assigns the absolute value of the negative weight as the number of processes for decomposition in that variable;* if this number exceeds the number of available processors, scaling is as in (3) below.
2. A ***zero value*** blocks *any* parallel decomposition from occurring for that variable.

3. A ***positive value*** scales normally based on the decmpv weighting vector. See the procedure described below. If ***no clear variable has decomposition priority,*** then the priority follows angular, group, and then space.

• In effect, **decmpv** defines the user desired aspect ratio of the 3-D processor array for the problem. During execution, these weights are then compared with the *actual number* of S_N directions (angles), energy groups, coarse mesh cells required, and *number of parallel processors* executing. The number of processors physically assigned to each level of decomposition is

• Decomposition procedure using **decmpv:** Consider a 3-D virtual rectangular parallelepiped processor array, of dimensions *A*G*S. A* processors are devoted to angular decomposition, *G* processors are devoted to group decomposition, and *S* processors are devoted to spatial (coarse mesh level only) decomposition, with a total number of processors totalling (*A*G*S*). The **decmpv** vector allows one to prioritize which variable is decomposed first, second, and third with then systematically performed by *PENTRAN based on the weighting system imposed by the user* with **decmpv**. It is usually best to lock in a set number of processors for one variable, and then let *PENTRAN* scale the other variables. For example, if a problem has 80 directions (**isn**=8), 2 energy groups, and 4 coarse meshes, one decmpv strategy could be decmpv=-2 1 0.5. Calling for 8 processors at execution, 2 processors would be locked in for angular decomposition, followed by maximizing group decomposition to 2 processors, followed by maximizing spatial decomposition with the remaining pool of processors. Note that in this case, the decomposition priority is allocated to energy groups, since it carries the largest (most positive) weighting factor. *PENTRAN* will attempt to autoscale to an assigned number of processors (as in the above example) to a problem that is consistent with a user's specified weighting vector.

• Also a consideration in assigning decmpv weights is that *PENTRAN always breaks the angles up into sweep octants, with a subsequent number of directions omega per octant.* This will affect the way angular decomposition is applied for a given number of processors, as there are *two* levels of decomposition that follow processor assignments in angular decomposition: there are always 8 octants, and *(isn*(isn+2)/8)* directions $\hat{\Omega}$ per octant. Therefore, octants should be decomposed using an evenly numbered processor assignment (see the next paragraph).

• The **decmpv** weights <u>must be chosen carefully by the user</u>, as setting a decomposition vector component too high will block possible scaling of useful processor work to another decomposition variable. ***If the processor utilization in each decomposition is less than 100%, execution halts. An integer number of processes must evenly divide so that integer portions can be evenly distributed on allocated processors at execution time. This is imposed due to the possibility of MPI communication synchronization failures among specific processors that are assigned odd numbers of angles, groups, or coarse mesh cells.***

• **ngroup** declares the group structure of the calculation. The first field in this 3-vector is the total number of groups in the current calculation. The second field is optional, and is the group width of the group window. The group window is a minimum of 1 group wide. A window of 1 group would therefore use only a single memory location through which to cycle energy groups, rather than saving angular fluxes in each locally computed group. The third parameter (also optional) is the total number of restart groups converged from a previous run. (note: this requires binary flux moment files to be available--see below). If positions (2) and (3) are not set, then position (2) is set to the number of groups in the first field, and position (3) is set to zero.

• Comments on ***Restart***: In the Restart procedure, flux moments are read as results from previous batch groups, and are used to calculate the transfer scattering source inside the code with restart for the new group batch. A new batch of groups will start with the medium grid if the simplified multigrid option is on, although there is only a fine grid transfer source (qinscf). In such cases, the 'closest mesh' approach is used by projecting the fine grid source back onto medium grid from the input fluxes. The cross sections are read in for the global total number of groups to be computed based on the **maxxsg** parameter, since this parameter is the ceiling of the number of groups to be considered among all multiple restart calculations. As an example, for a 20 group (total) calculation, if there are ten groups (Group 1 to Group 10) in the first restart batch with supporting binary flux moment files from an initial calculation, the next 10 (Group 11 to Group 20) require calculation. This requires all 20 groups to be loaded in for the cross sections, and therefore the parameter maxxsg=20. One may use ngroup=10, 1, 10 to indicate 10 groups starting at group 11 and proceeding through group 20, 1 group in the window, and groups 1 to 10 loaded from restart. Note that the *same parallel decomposition must be used* to allow the binary files to be read by the correct processor. Depending on the processor ID#, the binary flux moments are stored as input filename + '.r' + processor#. These restart files have the same format as dumped flux moments that are stored as input filename + '.f' + processor#--these can be renamed for a restart run by replacing the '.f' with a '.r.' Note: binary files can be very large for large 3-D problems, especially so for P_3 calculations. (Note: for a *non-restart c*alculation, the maxxsg parameter should be set to 0).

• A wall-clock time cutoff via the **timcut** variable insures a "safe" problem stop and data dump after the specified wall clock time is exceeded (in minutes). *If **timcut**=0, no cutoff time is activated.* The **timcut** option gaurantees that the user will have final results if the execution time is limited for any reason. Execution is halted after all processors signal a "wall time exceeded" synchronization, only after an iteration is complete, whereupon data is dumped. Note that since no intermediate data dumps are performed to maximize parallel efficiency, this feature provides a "safety net" if wall clock times are limited in a batch queue structure. This is also another means of job control, in addition to solution tolerances and maximum iterations (Block 4).

• Load balancing is handled by the code automatically if selected by the user. (The automatic load balancing option is selectable via the **lodbal=1** switch in Block 1). For example, if a coarse mesh contains significantly more medium/fine meshes relative to other coarse meshes, a sequential coarse mesh assignment may lead to an imbalanced processor workload; one processor effectively drags down the rest while it finishes calculations, and others sit idle waiting for message completion. With load balancing, the workload for each coarse mesh is evaluated and ranked, and cells with the densest medium/fine meshing and/or highest within group scattering ratio are paired with cells having the sparsest meshing and/or lowest scattering ratio. Coarse mesh cells and/or energy groups are then reracked according the workload in each, thus attempting to force each processor to carry the same computational load. While this spreads work (more) evenly, there are drawbacks:

1. An important disadvantage of automatic load balancing is that it may limit the effectiveness of the automatic red-black coloring feature (Block 4), as it is possible only "red" cells are assigned to a processor based on the computational load.

2. The rerack of cells may also inhibit problem convergence with regard to angular flux sweep progression.

• The **tolmgd** variable defines the grid structure to be used in solving the transport problem. **tolmgd<0** indicates the fine grid **only** is used in the iteration sequence. **tolmgd=0** indicates the medium grid **only** is used in the iteration sequence. Note that the **mathmg** array sets how materials on the medium grid are used, either using a "closest approach" or homogenized by volume assignment (see Block 2). **tolmgd>0** indicates a simplified multigrid sequence is used, where a solution is converged on the medium mesh grid to a relative local tolerance equal to **tolmgd**, whereupon the values are projected to the fine grid (where medium grid values are overwritten with projected fine grid values to conserve memory).

• Because converged medium grid values are projected and overwritten, there is no return to the medium grid –hence, the "simplified" or "slash" multigrid algorithm. The user is free to determine the tolerance for **tolmgd** and all grid structures. However, if too small a tolerance is chosen, the time spent converging to the less accurate medium mesh may outweigh the benefit of using two grids. If the **tolmgd** tolerance is set too loosely, the maximum benefit of using the multigrid acceleration will not be realized to provide an effective pre-conditioning of the fine grid values. Experience has shown that **tolmgd** should fall somewhere between 20 and 200 times the fine grid tolerance, but should also be no greater than the truncation error of the medium grid.

• Since coarser grids require more iterations to converge (in spite of being fewer in number), the difference between grids should not vary significantly from a factor of two in any one direction. If the problem is a criticality problem, the outer iteration tolerance used for the medium grid is implicitly determined from (**tolmgd/tolin*tolout**), but is also restricted from being any greater than **tolmgd**. Note that no differencing is based on the coarse mesh grids.

• Based on testing already performed, multigrid performance is enhanced when the Hiromoto-Wienke source iteration scheme (**methit**=2) is selected, especially in the case of criticality problems. See Block 4.

• To solve adjoint transport problems, the procedure has been somewhat automated in *PENTRAN*. By setting **modadj=1**, forward cross sections are reversed internally, with full automatic transposition of the scattering matrix, with $\nu\sigma_{f\,g}$ and χ_g also transposed internally. However, the user must recognize that:

1. Group G is reported as Group 1
2. Group 1 is reported as Group G (groups are reported in reverse order)
3. Directions $\hat{\Omega}$ are implicitly $-\hat{\Omega}$
4. A Group *G Adjoint* Source is input/reported as a Group 1 Source
5. A Group 1 Adjoint Source is input/reported as a Group *G* source

• A warning message is printed in each output file describing the effect of the adjoint calculation on the output data. Essentially, as long as the user has properly defined the source, the adjoint calculation proceeds automatically; the user must take heed of the warning message in analyzing the adjoint function output.

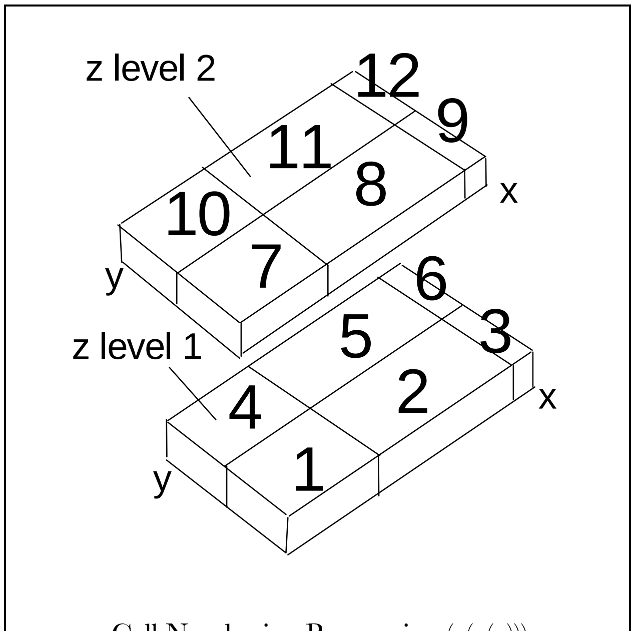
Block 2: Geometry

Block 2 includes problem geometry input parameters; all fields are required for code execution, and can be defined in any order. Twelve possible parameters in this block include: **xmesh, ixmed, ixfine, ymesh, jymed, jyfine, zmesh, kzmed, kzfine, nmattp, flxini, mathmg.**

Summary of Block 2 Inputs

Variable	Decription	Syntax Example
xmesh	coarse x mesh bdys	xmesh=*Coarse_xBdy_1, Coarse_xBdy_2, ...Coarse_xBdy* $_{(ixcrs+1)}$
ixmed	medium x mesh intervals in coarse mesh	ixmed=*medium_x_in_Coarse_Cell_1..*
ixfine	fine x mesh intervals in coarse mesh	ixfine=*fine_x_in_Coarse_Cell_1, ...*
ymesh	coarse y mesh bdys	ymesh=*Coarse_yBdy_1, Coarse_yBdy_2, ...Coarse_yBdy* $_{(jycrs+1)}$
jymed	medium y mesh intervals in coarse ymesh	jymed=*medium_y_in_Coarse_Cell_1..*
jyfine	fine y mesh intervals in coarse mesh	jyfine=*fine_y_in_Coarse_Cell_1, ...*
zmesh	coarse z mesh bdys	zmesh=*Coarse_zBdy_1, Coarse_zBdy_2, ...Coarse_zBdy* $_{(kzcrs+1)}$
kzmed	medium z mesh intervals in coarse mesh	kzmed=*medium_z_in_Coarse_Cell_1..*
kzfine	fine z mesh intervals in coarse mesh	kzfine=*fine_z_in_Coarse_Cell_1...*
nmattp	nmattp card for each coarse mesh	nmattp=*coarse mesh #, material#_in_fine-mesh_1..*
flxini	initial scalar flux in each coarse mesh	flxini=*initial_flux_in_Coarse_Cell_1..*
mathmg	sets material/homogenization for med grid 0: use closest matl; 1: use homog matl	mathmg=*Setting_in_Coarse_Cell_1..*

Termination of the Block with a T is required



Cell Numbering Progression (z(y(x)))

Block 2 NOTES:

- ***All three dimensional cells are numbered sequentially*** according to a ($z(y(x))$) integral loop count, where coarse cells are numbered by proceeding along all *x*, incrementing *y*, proceeding until the limit of *x* and *y* cells are reached, then incrementing z, and so on.
- ***This numbering scheme is also used throughout the hierarchy of cells, where medium and fine cells within each coarse mesh follow this mapping scheme.*** For example, consider that ixcrs=3, jycrs=2, kzcrs=2, or 12 total cells. Coarse cell numbers progress along x, then y, then z from 1 to 12, as shown in Fig 3.3.
- Benefits of the sequential numbering scheme, ***used for all grid hierarchies*** (coarse, medium, and fine grids) are: (1) it allows for a more compact, sequential dimensioning of spatial arrays, reducing the span of arrays in memory; (2) looping through spatial variables uses a one dimensional index; and (3) surrounding cells are easily identified using forward and reverse translation mapping functions, enabling efficient problem setup.

- Medium (fine) mesh intervals along each axis (e.g. *x, y*, and *z* in **ixmed, jymed, kzmed**, ... respectively) must be assigned for each coarse mesh number, in order, using the prescribed mesh numbering scheme. Again, *Taylor Projection Mesh Coupling* (*TPMC*) is used to couple transport sweeps across cell surface interfaces where medium (fine) meshes may be discontinuous.

- An **nmattp** card must be present for each coarse mesh number. The first entry in each **nmattp** card is the coarse mesh cell number, followed by the material number assigned for each fine mesh contained in the coarse mesh cell using the order of the standard sequential mesh numbering scheme (proceeding along all x, incrementing y, ...). The material number must directly correspond to the rank order of the material in the cross section input deck. Material numbers must not exceed the number of materials (**nmatl**) defined in Block I.

- The **flxini** array, if non-zero for each coarse mesh, is energy group-weighted by **chig** (in Block 3), where **chig** is the fission spectrum group probability distribution variable. The **flxini** values are weighted by **chig** *even if there is no fission present*. Note: fission is *only* computed if the problem type (**nprtyp** in Block 4) is set for a criticality eigenvalue problem. If a problem is solely a fixed source problem, any fission cross sections read in.

- The **mathmg** array sets up how materials are treated on the medium grid, and requires a value for each coarse mesh number, assigned according to the sequential numbering scheme. If set to zero for a coarse mesh cell, this indicates that the material defined on the medium grid shall use the "closest approaching material" defined on the fine grid as a medium grid material specification. If set to unity, the cross sections on the medium grid are homogenized by volume based on the fine grid material specification for that coarse mesh cell. If material boundaries can be approximately resolved, the closest approaching material setting may perform best.

- A geometry file is always generated (unless deactivated in Block 7) as *fileprefix*+'.geo'. This file contains a *Mathematica*^{TM} graphics command deck to automatically render a full color, 3-D geometry image of the problem. This is useful for viewing/verifying problem geometries. Read the *filename*.geo file into *Mathematica*^{TM} directly as a text file, or cut and paste portions of it from a text editor. Using *Mathematica*^{TM}, coarse meshes can be made invisible for viewing other coarse meshes buried within the problem structure. See the cover of this manual for examples. *Mathematica*^{TM} is available from Wolfram Research, Inc for Unix-x stations, PC-Windows, and Mac operating systems. A very powerful workstation may be required to render a suitable 3-D image for very large problems.

- Translations between sequential cell numbers and **ixcrs**, **jycrs**, and **kzcrs** coordinate position are made by calling the CELMAP and CIVMAP routines in *PENTRAN*.

Block 3: Cross Section Parameters

Block 3 includes problem macroscopic material cross section parameters; all fields are required for code execution, and can be defined in any order. Ten possible parameters in this block include: **lib, legord, legoxs, nxtyp, ihm, iht, his, ihng, chig, nxcmnt.**

Summary of Block 3 Inputs

Variable	Decription	Syntax Example
lib	cross section library location	lib=cards *(input card images from input deck)* lib=file:*filename (load xsecs from filename)*
legord	Legendre scattering order (up to 7)	legord=*scattering_order* (0,1,3,5, or 7)
legoxs	Legendre scattering order (up to 7) of cross sections to be read	legoxs=*scattering_order* (0,1,3,5, or 7)
nxtyp	cross section type (See Notes)	nxtyp=*xsec_type* (0,1,2,3,4,5,6,7, or 8)
ihm	number of rows of xsec data	ihm=*position_of_last_row*
iht	total xsec row position	iht=*total_xsec_row_position*
ihs	within group scatter xsec	ihs=*g->g_xsec_row_position*
ihng	position of last neutron scatter xsec prior to coupled gamma xsecs	ihng=*last_neutron_xsec_row_position*
chig	group fission probabilities for each material (1,2,...), by (group...)	chig= *mat1_Grp1_prob, mat1_Grp2_prob, ... mat2_Grp1_prob, mat2_Grp2_prob, ...*
nxcmnt	number of cross section comment cards	nxcmnt=*number_of_79_col_cards*
	Termination of the Block with a T is required	

Block 3 NOTES:

- The library input **lib** can be from a datafile or on card images in the input deck.
- Cross sections read directly from the input deck should immediately follow Block 3, and must be terminated with a Block "T" terminator on the same line as, and immediately following the last cross section entry.
- Zero fields should be used to "pad" null values in the scattering matrix, as applicable.
- All filenames assume a path with an 8 character filename prefix, followed by a 3 character (maximum) filename suffix ('.xxx').
- The Legendre order **legord** must be no greater than **isn**-1 (from Block 1) to properly integrate Legendre expansions using level-symmetric quadratures.
- The Legendre scattering order of the cross sections **legoxs** can equal or exceed the the **isn** value, as long as the scattering order called for in computations is less than **isn**.
- Both **legord** and **legoxs** must be zero or odd (to be able to represent peaked scattering situations with odd moment expansions). Violating these rules results in an execution halt.
- The cross section type parameter **nxtyp** allows for 9 different cross section file formats (0-8). The format refers to row or column input, ASCII or BINARY data types, and whether or not the cross sections include the $(2l+1)$ normalization factor for Legendre moments.

- The available formats are listed as follows:

  - 0: STANDARD (row) form: NO, Legendre consts NOT pre-multiplied
  - 1: STANDARD (row) form: YES, Legendre consts ARE pre-multiplied
  - 2: NON-STD (col) form: NO, Legendre consts NOT pre-multiplied
  - 3: NON-STD (col) form: YES, Legendre consts ARE pre-multiplied
  - 4: STANDARD (row) BINARY FILE form: NO, Legendre consts NOT pre-multiplied
  - 5: STANDARD (row) BINARY FILE form: YES, Legendre consts ARE pre-multiplied
  - 6: NON-STD (col) BINARY FILE form: NO, Legendre consts NOT pre-multiplied
  - 7: NON-STD (col) BINARY FILE form: YES, Legendre consts ARE pre-multiplied
  - 8: GIP-ORNL BINARY FILE form: YES, Legendre consts ARE pre-multiplied

- There is no difference between ASCII and BINARY file read *formats* in the standard row and non-standard column form; BINARY data is assumed to be stored in the same relative order as the ASCII form. The GIP-ORNL format assumes that a binary file, generated by the GIP program for mixing materials, generated the cross section file. (The GIP-ORNL format reads blocks of data for all materials and Legendre orders by energy group, which differs from the STANDARD and NON-STANDARD formats).

- To be compatible with binary file formats, all cross sections are input and stored as single precision.

- Examples of the above file formats for a typical 7-group set of cross sections are below. In all cases, the **iht**, **ihs**, and **ihm** parameters are indicated, as applicable. Since there are no gamma groups, ihng=0; this parameter is in place as an indicator. Note the structure of each is repeated for every scattering moment.

- STANDARD FORMAT (rows) With UP *and* DOWN Scatter: iht=3, ihs=10, ihm=16

```
...nxcmnt comment cards (check file for compliance if lib=file:filename) ...
siga1 rnsigf1 sigt1 sig7->1 sig6->1 ...sig2->1 sig1->1    0       0       0...
siga2 rnsigf2 sigt2    0    sig7->2 ...sig3->2 sig2->2 sig1->2    0       0...
siga3 rnsigf3 sigt3    0       0    ...sig4->3 sig3->3 sig2->3 sig1->3    0...
siga4 rnsigf4 sigt4    0       0    ...sig5->4 sig4->4 sig3->4 sig2->4 sig1->4...
siga5 rnsigf5 sigt5    0       0    ...sig6->5 sig5->5 sig4->5 sig3->5 sig2->5...
siga6 rnsigf6 sigt6    0       0    ...sig7->6 sig6->6 sig5->6 sig4->6 sig3->6...
siga7 rnsigf7 sigt7    0       0    ...   0    sig7->7 sig6->7 sig5->7 sig4->7...
```

- STANDARD FORMAT (rows) With DOWN Scatter only: iht=3, ihs=4, ihm=10

```
...nxcmnt comment cards (check file for compliance if lib=file:filename) ...
siga1 rnsigf1 sigt1 sig1->1    0       0       0       0       0      0
siga2 rnsigf2 sigt2 sig2->2 sig1->2    0       0       0       0      0
siga3 rnsigf3 sigt3 sig3->3 sig2->3 sig1->3    0       0       0      0
siga4 rnsigf4 sigt4 sig4->4 sig3->4 sig2->4 sig1->4    0       0      0
siga5 rnsigf5 sigt5 sig5->5 sig4->5 sig3->5 sig2->5 sig1->5    0      0
siga6 rnsigf6 sigt6 sig6->6 sig5->6 sig4->6 sig3->6 sig2->6 sig1->6   0
siga7 rnsigf7 sigt7 sig7->7 sig6->7 sig5->7 sig4->7 sig3->7 sig2->7 sig1->7
```

➢ NON-STANDARD FORMAT With UP and DOWN Scatter: iht=3, ihs=10, ihm=16

```
...nxcmnt comment cards (check file for compliance if lib=file:filename) ...
 siga1   siga2   siga3   siga4   siga5   siga6   siga7
 rnsigf1 rnsigf2 rnsigf3 rnsigf4 rnsigf5 rnsigf6 rnsigf7
 sigt1   sigt2   sigt3   sigt4   sigt5   sigt6   sigt7
 sig7->1    0       0       0       0       0       0
 sig6->1 sig7->2    0       0       0       0       0
 sig5->1 sig6->2 sig7->3    0       0       0       0
 sig4->1 sig5->2 sig6->3 sig7->4    0       0       0
 sig3->1 sig4->2 sig5->3 sig6->4 sig7->5    0       0
 sig2->1 sig3->2 sig4->3 sig5->4 sig6->5 sig7->6    0
 sig1->1 sig2->2 sig3->3 sig4->4 sig5->5 sig6->6 sig7->7
    0    sig1->2 sig2->3 sig3->4 sig4->5 sig5->6 sig6->7
    0       0    sig1->3 sig2->4 sig3->5 sig4->6 sig5->7
    0       0       0    sig1->4 sig2->5 sig3->6 sig4->7
    0       0       0       0    sig1->5 sig2->6 sig3->7
    0       0       0       0       0    sig1->6 sig2->7
    0       0       0       0       0       0    sig1->7
```

➢ NON-STANDARD FORMAT With DOWN Scatter: iht=3, ihs=4, ihm=10

```
 ...nxcmnt comment cards (check file for compliance if lib=file:filename) ...
 siga1   siga2   siga3   siga4   siga5   siga6   siga7
 rnsigf1 rnsigf2 rnsigf3 rnsigf4 rnsigf5 rnsigf6 rnsigf7
 sigt1   sigt2   sigt3   sigt4   sigt5   sigt6   sigt7
 sig1->1 sig2->2 sig3->3 sig4->4 sig5->5 sig6->6 sig7->7
    0    sig1->2 sig2->3 sig3->4 sig4->5 sig5->6 sig6->7
    0       0    sig1->3 sig2->4 sig3->5 sig4->6 sig5->7
    0       0       0    sig1->4 sig2->5 sig3->6 sig4->7
    0       0       0       0    sig1->5 sig2->6 sig3->7
    0       0       0       0       0    sig1->6 sig2->7
    0       0       0       0       0       0    sig1->7
```

- **chig** is the group fission probability for each material, and must correspond to each material number (in the order the materials are read in). Therefore, **chig** must be input as a vector of length **ngroup*nmatl**. Note that in the absence of fission, **chig** is **still used** as a weighting factor for the initial guess of scalar flux in each coarse mesh (**flxini**--see Block 2). *For this reason,* **chig** *is <u>un-normalized</u>. The user should insure when fission is present, group fission fractions sum to 1.0.*

- **nxcmnt** is the number of comment cards preceeding each Legendre order cross section set. (This setting has no effect on GIP-ORNL cross section formats). If capital letters are used in the cross section comment fields, the comments should be preceeded by a slash to avoid confusion with *FIDO* control characters during input processing.

- ***Adjoint*** **cross sections are** ***autoimatically transposed internally*** by setting the **modadj** parameter--see Block I inputs. By transposing the cross sections (inside *PENTRAN*), a forward code can be used to solve for the adjoint function, with proper attention to source definitions and fission parameters. Formats for up- and down-scatter and down scatter only are provided here for completeness.

➢ STANDARD *ADJOINT* TRANSPOSED FORM,UP-DOWN Scatter: iht=3,ihs=10, ihm=16

```
siga7 rnsigf7 sigt7 sig7->1 sig7->2 ...sig7->6 sig7->7    0       0       0...
siga6 rnsigf6 sigt6    0    sig6->1 ...sig6->5 sig6->6 sig6->7    0       0...
siga5 rnsigf5 sigt5    0       0    ...sig5->4 sig5->5 sig5->6 sig5->7    0...
siga4 rnsigf4 sigt4    0       0    ...sig4->3 sig4->4 sig4->5 sig4->6 sig4->7...
siga3 rnsigf3 sigt3    0       0    ...sig3->2 sig3->3 sig3->4 sig3->5 sig3->6...
siga2 rnsigf2 sigt2    0       0    ...sig2->1 sig2->2 sig2->3 sig2->4 sig2->5...
siga1 rnsigf1 sigt1    0       0    ...   0    sig1->1 sig1->2 sig1->3 sig1->4...
```

➢ STANDARD *ADJOINT* TRANSPOSED FORM, DOWN Scatter: iht=3,ihs=4, ihm=10

```
siga7 rnsigf7 sigt7 sig7->7    0       0       0       0       0      0
siga6 rnsigf6 sigt6 sig6->6 sig6->7    0       0       0       0      0
siga5 rnsigf5 sigt5 sig5->5 sig5->6 sig5->7    0       0       0      0
siga4 rnsigf4 sigt4 sig4->4 sig4->5 sig4->6 sig4->7    0       0      0
siga3 rnsigf3 sigt3 sig3->3 sig3->4 sig3->5 sig3->6 sig3->7    0      0
siga2 rnsigf2 sigt2 sig2->2 sig2->3 sig2->4 sig2->5 sig2->6 sig2->7   0
siga1 rnsigf1 sigt1 sig1->1 sig1->2 sig1->3 sig1->4 sig1->5 sig1->6 sig1->7
```

Block 4: Control Options

Block 4 includes execution control options. Fields can be defined in any order. Twelve possible parameters in this block include: **nprtyp, nrdblk, tolin, tolout, maxitr, methit, methac, ncoupl, ndmeth, nzonrb, dtwmxw, nquit.**

<u>Summary of Block 4 Inputs</u>

Variable	Decription	Syntax Example
nprtyp	problem type classification	nprtyp=*problem_type* ([-maxsrc, 0, maxsrc])
nrdblk	automatic red-black coloring switch	nrdblk =0 (off) or =1 (on)
tolin	inner local flux iteration tolerance	tolin=*tolerance* OR tolin=CM1_tol, CM2_tol...
tolout	outer criticality tolerance, med grid multiplier	tolout=*tolerance, med_grid_multiplier*
maxitr	max inner & outer iters/group, k-inner iter limit	maxitr=*max_no_of_iterations, criticality_inner_limit*
methit	source iteration method	methit=*method_no* =1 ("Multigroup") or =2 ("Hiromoto-Wienke")
methac	acceleration method, csda parameters	methac=*method_no* (0,1,2,3,4,5, or 6), csda parameters
ncoupl	*Taylor Projection (TPMC)* coupling order	ncoupl=*coupling order* (0 or 1)
ndmeth	differencing method by coarse mesh no	ndmeth= *Coarse1_diffmeth, Coarse2_diffmeth, ...* =0(DD), =1(DZ), =2(DTW), =3(EDW), =4(EDH)
nzonrb	zones, damping, skip iterations for methac	nzonrb=*number_of_zones, damping_fact, skip_Iter*
dtwmxw	maximum acceptable DTW weight permitted before a shift to EDW is made	dtwmxw=*maximum_weight* ([0.5,1.0]) recommended is 0.96
nquit	# iters a non-converging group is stopped	nquit=*number_of_iterations*

Termination of the Block with a T is required

Block 4 NOTES:

- **nprtyp** determines the type of transport problem to be solved:
  - ✦ **nprtyp**=0 indicates a criticality eigenvalue problem (no fixed sources)
  - ✦ **nprtyp**>=1 corresponds to 1 or more fixed sources, equal to the number of fixed sources in the problem. There is one fixed volumetric and planar source permitted for each coarse mesh--see Block 5. This input requires sdef cards, defined in Block 5.
  - ✦ **nprtyp**<=-1 is a combination of the first two options: a criticality k-eigenvalue *with* fixed sources present
  - ✦ the number of fixed sources is limited by the maxsrc parameter in *PENTRAN*
- **tolin** defines the maximum local relative flux error for convergence. The **tolout** variable defines the relative tolerance on the *k*-effective system eigenvalue, and is only used in a criticality calculation. Solution convergence progression, including intermediate *k*-effective values and relative tolerances, are stored in logfiles, if used (see the **loglevel** indicator in the Header card). The final system *k*-effective is reported in the output file.
- Note that while a solution may be converged based on the local relative convergence criteria **tolin**, the <u>solution *must* satisfy the scalar balance equation to be converged</u>. The integral system balance is collected and computed on each processor during problem output. If the net balance reported is not on the order of the

solution tolerance (e.g. within an order of magnitude of the solution tolerance, sometimes a bit more if the problem is fully decomposed in parallel), then convergence has not been reached.

• **maxitr** is a vector containing iteration limits. The first entry is a standard inner iteration limit. The second (optional) entry is an inner iteration limit for criticality problems. The first entry iteration limit holds for both the inner and outer loop tolerances. It is recommended to set the maxitr variable to be based on the inner iteration loop. A data dump occurs is **maxitr** is exceeded. The wall clock time (**timcut** in Block 1) can be used to control execution for the outer loop, if necessary.

• **methit** refers to the algorithm for the source iteration sequence, either the "*multigroup*" (methit=1) method or the one-level scheme (methit=2) of Hiromoto and Wienke (1989). Either scheme may be used with any phase space decomposition strategy, although multigrid acceleration seems to perform best with the Hiromoto-Wienke scheme (see Sjoden and Haghighat, 1996).

• **methac** is a vector input variable, where the first position refers to the rebalance acceleration method used. At present, damped Coarse Mesh Rebalance (CMR) and System Rebalance (SR) are available, based on coarse mesh cells. Damping restricts oscillations and divergence in the rebalance (Rhoades, 1981). Note that rebalance is used to scale only the scalar flux (as opposed to the angular fluxes), and is performed to avoid additional reductions if angular decomposition is used. Options for **methac** are:

1. **methac=0**: No Acceleration
2. **methac=1** : *SR* only (based on **nzonrb** coarse mesh cells-see below)
3. **methac=2** : *PCR* only (based on **nzonrb** coarse mesh cells-see below)
4. **methac=3** : Alternating *PCR/SR* (based on **nzonrb** coarse mesh cells-see below)
5. **methac=4, 5, or 6**: Same as 1,2,3, respectively, but utilizing global synchronization.

Subsequent vector positions are for CSDA (collapsed source diffusion acceleration) sources. Note: at present, this is an unproven method by which to accelerate the transport solution, and is not recommended for use at this time. This is a completely experimental feature where an analytical diffusion approximation is used in each energy group to preset flux iterates in PENTRAN based on a collapsed source from integration over problem geometry. The vector positions for this are as follows, starting with methac position (2) entry: csda init Off=0/Point=1/Cosine=2), csda xcenter (0=auto), csda ycenter (0=auto), csda zcenter (0=auto), inner iterate # for application use of csda, outer iterate # for application of csda. Zero entries for "center" values force PENTRAN to determine an average source location based on the distribution. "Point" refers to a collapsed point source; "Cosine" refers to a cosine source distribution.

• **methac**=1,2, or 3 settings use processor communications for rebalancing (using group-wise processor communicators), and can be used with any source iteration scheme.

• **methac**=4 ,5, or 6 requires an intermediate step of global process communications (which can be significantly more expensive), and can only be used with **methit=2**. This distinction between the use of communications in rebalancing methods was necessary due to limitations encountered on some parallel system implementations of the MPI standard.

• *PCR* rebalancing is performed using a direct Cholesky-LU factorization of the group rebalance matrix (the rebalance matrix is stored as an augmented system in an array, and is replaced in memory by Cholesky-factorized lower and upper triangular matrices). There is a limit on the size of the augmented rebalance matrix. Although the rebalance matrix is dimensioned as n x $(n+1)$ of single precision numbers. The number of contiguously numbered coarse mesh cells (e.g. a block of coarse meshes) that can be rebalanced at one time is limited by the **maxcmr** parameter in *PENTRAN*; as a result, multiple "blocks" of rebalance operations may be required, depending on how many coarse meshes are defined for a problem. A direct solution (rather than an iterative one) is used for rebalance in *PENTRAN* because efficient synchronization required, especially with complete phase space decomposition.

• The **nzonrb** vector permits the user to specify: (1) how many coarse meshes should be considered in a rebalance zone (up to a current maximum of **maxcmr**), loaded into position 1; (2) the damping factor to be used with partial current rebalance, where *dampf* <1 is under-damped, *dampf* =1 is critically damped, and *dampf* >1 is *over*-damped (in general, not recommended), loaded into vector position 2; (3) the iteration *period* wherein rebalance is skipped, loaded inot position 3 of the **nzonrb** vector. If the execution is parallel, a check is made to determine if the locally assigned coarse meshes (determined by the **nzonrb**(1) setting) belong to the zone of coarse meshes undergoing rebalance; if not, rebalance is bypassed, saving execution time. *The user should be aware that setting **nzonrb** too small (covering a small block of coarse meshes) may cause iteration instability and divergence of the solution, as may using overdamping--of course, this is problem dependent.*

• **ncoupl** defines the order of coupling for Taylor Projection Mesh Coupling. Setting **ncoupl**=0 forces *0th* order, while setting **ncoupl**=1 implements first order coupling. A zero setting forces all coupling coefficients to be zero, thus invoking only a straight flow balance at each interface during a transport sweep. Note that restricting the coupling to *0th* order does not save significant computational work.

• **ndmeth** defines the spatial differencing method to be used, and is a vector containing the differencing method to be used in each coarse mesh cell. At present, setting this =0 selects DD (with no fixup, no upgrade); =1 selects DZ (can be upgraded to DTW, EDW); =2 in each coarse mesh selects DTW (can be upgraded to EDW); =3 selects EDW, while =4 selects EDH. An upgradeable diamond (DZ) setting will automatically use DTW differencing if a negative (set to zero) flux fixup is detected during any angular flux sweep. From there, if a DTW maximum weight (in any direction) is greater than **dtwmxw**, then the DTW method is upgraded to EDW. This allows EDW to take over from DTW, since high weights in DTW tend to occur in thicker cells with streaming, where EDW is more accurate. A negative differencing number locks that method (with the exception of DD, already locked using a zero setting), blocking upgrade options; for example, setting **ndmeth** to -1 in a coarse mesh cell locks DZ differencing regardless of fixup calls, blocking any upgrade. For more information on differencing, refer to Chapter 2 of this document.

• **dtwmxw** is the optional user specification of the maximum accepted weight for DTW (if used in an adaptive differencing status) that is used to trigger an upgrade to EDW differencing, or use of EDW rather than DTW for a given discrete ordinate in the case fo EDH differencing, If this is not specified in the input deck, a value of 0.96 is assumed. Note that **dtwmxw** must fall within the valid range for DTW weights: [0.5,1.0].

• **nrdblk**=1 can be set to engage automatic red-black coloring based on problem coarse meshes, used only to accelerate parallel spatial decomposition. Coarse meshes are re-ordered in a "stacked checkerboard" sequence to use the most recent iteration angular fluxes as soon as they are available, to the greatest extent possible. Note that use of this setting when not performing parallel spatial decomposition can (in some cases) hinder convergence, and should be avoided. A warning message is issed if **nrdblk** is engaged without spatial decomposition. Also note that the effectiveness of red-black coloring may be limited by automatic load balancing, especially if only "red" (or "black") cells are allocated to one processor due to load reracking (see Block 1).

• **nquit** sets the maximum number of iterations permitted to stop a non-converging group when that group fails to converge based on a specific location (coarse and medium or fine mesh number) in the problem. The defailt value is nquit=4 if no value is specified. After failure, the group is set as "converged" with warnings.

Block 5: Source Definition and Options

Block 5 includes source definition options. Fields can be defined in any order. Ten possible parameters in this block include: **nsdef, nscmsh, ssnrm, sref, serg, smag, spacpf, omegap, scalsf, rkdef.**

<u>Summary of Block 5 Inputs</u>

Variable	Decription	Syntax Example
nsdef	Source type of each source (0=volumetric, 1=planar source)	nsdef=*source1_type , source2_type, ...*
nscmsh	Corse mesh number of each source	nscmsh=*coarse_mesh#_source1, ...*
ssnrm **	Surface Source normal vector	ssnrm=*u1,v1,w1,u2,v2,w2, ...*
sref	Arbitrary reference coordinate for each source	sref=*x1,y1,z1,x2,y2,z2, ...*
serg	Source energy distribution with respect to energy groups (1...ngroup), each source	serg=*g1_prob_source1, g2_prob_source1, ...*
smag	Source integral magnitude, each source	smag=*source1_magnitude, source2_magnitude ...*
spacpf *	Source fine mesh spatial dist, specified	spacpf=*src#, grp#, #cells, cell1_prob, cell2_prob ...*
omegap *	Source angular dist, specified by octant no	omegap=*src#, grp#, octant#,* Ω_{1}*_prob,...* Ω_{n}*_prob.*
scalsf *	Source fine mesh scale factor, specified	scalsf=*src#, grp#, scale_factor ...*
rkdef	Criticality eigenvalue estimate for system	rkdef= k*eff_estimate*

*Optional
**Required for each source if nsdef=1 detected

Termination of the Block with a T is required

Block 5 NOTES:

- In the event of a criticality problem with no fixed sources, *only* the **rkdef** card is required. **rkdef** is the user-supplied initial guess for the integral system k_{eff} (criticality eigenvalue). In the event of a fixed source problem with no fission, the **rkdef** card is not required.
- All sources are defined as isotropic with equal probability in each fine spatial mesh ***unless modified*** by **omegap** and/or **spacpf**. To be considered "active," these probability distributions must assign values to have an integral probability magnitude >1.E-15. The medium mesh source is automatically constructed using a "nearest neighbor" approach from the fine grid source definition. If the medium grid density is too sparse to resolve a source distribution based on the fine grid, a warning message is issued.
- **nsdef** defines either a volumetric source (#/cm^3/s) or a planar (#/cm^2/s) source. One of each is possible in each coarse mesh cell. The number of entries in **nsdef** must equal the absolute value of **nprtyp** (see Block 4).
- **nscmsh** is a vector providing the reference location of sources (in order of source number) by coarse mesh number. The number of entries in **nscmsh** must equal the absolute value of **nprtyp** (see Block 4).
- **sref** permits an arbitrary reference coordinate to be specified for each source, in order of source number. The number of entries in **nsref** must equal the absolute value of **nprtyp*3** (see Block 4).
- **serg** is a vector containing the energy group probability distribution for each source, in order of source number. The number of entries in **serg** must equal the absolute value of **nprtyp*ngroup** (see Blocks 1, 4).

• smag defines the integral source magnitude (over all variables), in order of source number. The number of entries in smag must equal the absolute value of nprtyp (see Block 4). Negative smag entries activate source normalization for the source number.

• ssnrm is only required if there are nsdef entries equal to 1, indicating planar sources. *In that case,* ssnrm *entries are required for* *__all__* *sources,* although they are not used in the case where sources are volumetric. The ssnrm defines a vector for each source pointing from the center of the coarse mesh where the source is located. *PENTRAN* turns this vector into a *unit vector,* and then assigns a planar source to the surface of the 3-D coarse mesh boxoid normal to the largest component of that unit vector. If the surface is on a shared *interior* problem boundary, a similar planar source is implicitly defined for the adjacent mesh cell sharing the common boundary. *Since only **one** surface source can be defined for one coarse mesh, the coarse mesh containing the implicitly defined planar source cannot contain any other planar source.* This restriction was made due to memory limitations. Therefore, planar sources on interior surfaces cannot use discontinuous meshing between coarse mesh cells on the common surface where the source is located. Note that a spatial distribution can also be assigned to a volumetric source(s) to simulate a planar source, although volumetric sources are assumed to be averaged at the cell center. The number of entries in ssnrm must equal the absolute value of nprtyp*3 (see Block 4).

• Source particles from planar sources located at system boundaries entering into the system show up *implicitly* in boundary currents of particle balances. Planar sources defined on *interior* surfaces will be indicated in the system balances as "source particles."

• omegap is an *optional variable* where the user can specify an angular probability distribution for any coarse mesh source. *If this variable **is not defined** for the source number,* an equal (isotropic) angular probability is assigned for the source. *If this variable **is defined** for the source number,* it must be specified to indicate the source number, group number, angular octant number, and corresponding probability for each angle in the octant *only for the sources that require a non-uniform angular probability distribution. If the group number is entered as a negative group number, the angular probability function is systematically applied to all energy groups.* Each angular probability distribution is un-normalized. Probabilities for the total number of angles in any specified octant are required, and it is the user's responsibility to supply the correct number. Probabilities for octants not defined are implicitly assumed to be zero. A numbering pattern to guide probability assignment for sweep octants (and angles in each sweep octant) is provided in the Appendix; a complete quadrature set can be printed showing octants, vectors, and all other quadrature information using the *PENQUAD* utility if this information is required.

• spacpf is an *optional variable* where the user can specify a fine mesh spatial distribution to a any coarse mesh source. *If this variable **is not** defined for the source number,* an equal spatial probability is assigned for the source. *If this variable **is** defined for the source number,* it must be specified to indicate the source number, group number, number of sequentially numbered meshes, and corresponding mesh spatial probabilities *only for the sources that require a non-uniform spatial probability distribution.*

• *If the group number is entered as a negative group number in* spacpf, *the spatial probability function is systematically applied to* *__all__* *energy groups.* If a scalsf is declared, then this is used for all groups as well (see below). *Probabilities for cells not defined are assumed to be zero.* Note that if an un-normalized distribution is specified (the default), a warning message is reported.

• For a planar boundary source, each spacpf probability is still referenced by fine sequential cell number-- although probabilities in cells not on the coarse mesh boundary technically have no effect on the boundary cell source probabilities. For clarity, the user should still include zeros for cell location probabilities not on the boundary surface. Again, the sequential numbering scheme described in Block 2 for coarse meshes also applies to fine (and medium) meshes contained within each coarse mesh. (Note that cell probabilities are automatically transferred to the implicitly defined planar source for the adjacent coarse mesh on the common boundary--this is why discontinuous meshing on the boundary surface containing the source is not permitted).

• **scalsf** is an optional scale factor available for applying a scalar multiplier to any *spatial* probability distribution. To be "active," the magnitude of the scale factor must be greater than 1.E-50. The source number, group number, and positive scale factor, in order, are required for each applicable source. Note: if the **spacpf** distribution applied to all groups (so that the group number is entered as a negative number), then the scale factors defined in **scalsf** only need to be entered for group 1, and will be automatically apllied throughout all groups.

Block 6: Boundary Conditions

Block 6 includes boundary conditions. All input fields are required and can be defined in any order. Six possible parameters in this block include: **ibback, ibfrnt, jbeast, jbwest, kbsout, kbnort.**

Summary of Block 6 Inputs

Variable	Decription	Syntax Example
ibback	Global "back" (-x) surface boundary condition	ibback=*type, group1_albedo, group2_albedo,...*
ibfrnt	Global "front" (+x) surface boundary condition	ibfrnt=*type, group1_albedo, group2_albedo,...*
jbeast	Global "east" (-y) surface boundary condition	jbeast=*type, group1_albedo, group2_albedo,...*
jbwest	Global "west" (+y) surface boundary condition	jbwest=*type, group1_albedo, group2_albedo,...*
kbsout	Global "south" (-z) surface boundary condition	kbsout=*type, group1_albedo, group2_albedo,...*
kbnort	Global "north" (+z) surface boundary condition	kbnort=*type, group1_albedo, group2_albedo,...*

Termination of the Block with a T is required

Block 6 NOTES:

- Note the name of each boundary condition begins with the axis normal to each surface. Global boundary condition names correspond to global system boundaries with a right handed 3-D Cartesian coordinate system, "back" (-x), "front" (+x), "east" (-y), "west" (+y), "south" (-z), and "north" (+z).
- Boundary types (illustrated with **ibback**):
  - ✦ **ibback**=0 is a *vacuum* boundary. *No albedo factors are required*.
  - ✦ **ibback**=1, ... is an albedo boundary--***ngroup*** *albedo factors are required immediately following the type* (the "1").
- For perfect spectrally reflective boundaries, all group albedo factors should be *unity* (1.0).
- Gray/White boundaries are **not** supported.
- Periodic boundaries are **not** supported.

Block 7: Print Controls

Block 7 includes print controls. All input fields are optional and can be defined in any order. No specification for any print controls sets maximum printing. . Nine possible parameters in this block include: **nxspr, ngeopr, nsumpr, meshpr, nfdump, nsrcpr, nsdump, nmatpr, nadump**

Summary of Block 7 Inputs

Variable	Decription	Syntax Example
nxspr	Print cross section tables in output	nxspr=0 (off) or =1 (on)
ngeopr	Print a *Mathematica*™ readable geometry file	ngeopr=0 (off) or =1 (on)
nsumpr	Print local coarse mesh summary tables in output	nsumpr=0 (off) or =1 (on)
meshpr	Vector list of coarse mesh cells with formatted output	meshpr=0 (none) or *coarse_mesh#, ...*
nfdump	Binary data dump of mesh scalar fluxes/moments	nfdump=0 (off)or =(*legord_dumped+1*)
nsrcpr	Print source/distribution tables in output	nsrcpr=0 (off) or =1 (on), =2 (detailed)
nsdump	Binary data dump of mesh scalar sources	nsdump=0 (off) or =1 (on)
nmatpr	Print a material map in output files for local coarse cells	nmatpr=0 (off) or =1 (on)
nadump	Vector list of Binary data dump for angular fluxes	nadump=*coarse_mesh#, dump_type,*

Termination of the Block with a T is required

Block 7 NOTES:

- **Default settings are as follows:** nxspr=1, ngeopr=2, nsumpr=1, meshpr=0, nfdump=1, nsrcpr=1, nsdump=1, nmatpr=1, nadump=0. These settings are used if the parameter is not listed in the input deck.

- **ngeopr** can be set to values between [0,2], and prints increasingly detailed information about the problem geometry. If **ngeopr=1**, only coarse mesh geometry information is output; if **ngeopr=2**, then medium/fine mesh details are printed.

- If no **meshpr** field is specified, full formatted output, incuding six-face partial and net currents, are output for all coarse and medium (fine) meshes. **A negative coarse mesh number supresses the mesh partial current output, yielding formatted output of mesh scalar fluxes only.** It is not recommended that medium (fine) mesh data be specified for many coarse meshes here, since flux moments are accessible from binary files (see **nfdump**), automatically managed using the PENDATA utility. Still, it is helpful to request data from a few sample coarse meshes, since optical thickness, current balance, and other helpful data are printed into the output file.

- The value of **nfdump** specifies the order of the angular flux moments to be dumped; setting this =1 indicates that only *zeroth order moments* (scalar fluxes) will be dumped. The maximum is (**legord**+1), which would dump all moments through **legord**. If one desires only scalar fluxes from one portion of the problem, it may be better to select the appropriate coarse mesh using **meshpr**, since **nfdump** yields (binary) moment data for the entire problem phase space. Again, binary files are dumped by each processor only for the phase space computed on that processor. Data from multiple processor files can be automatically gathered with the *PENDATA* utility.

- Binary files generated by **nfdump** and **nsdump** have file input prefixes + '**.f***processor#*' and '**.s***processor#*', respectively. Note that since medium (fine) mesh data are stored locally, the output on each processor number is dependent on coarse mesh and group processor assignment.

• The **nadump** switch permits the cell centered fine mesh angular fluxes from a particular coarse mesh to be dumped to binary file format. Each coarse mesh number listed must be followed *immediately* by an angular flux ***dump type***, which specifies the hemisphere of angular fluxes to dump for each medium/fine mesh in the coarse mesh, as follows:

- 1 indicates Type 1: - x hemisphere, sweep octants 1, 3, 6, 8
- 2 . . . . . . Type 2: +x hemisphere, sweep octants 2, 4, 5, 7
- 3 . . . . . . Type 3: - y hemisphere, sweep octants 1, 3, 5, 7
- 4 . . . . . . Type 4: +y hemisphere, sweep octants 2, 4, 6, 8
- 5 . . . . . . Type 5: - z hemisphere, sweep octants 1, 4, 5, 8
- 6 . . . . . . Type 6: +z hemisphere, sweep octants 2, 3, 6, 7
- 7 . . . . . . Type 7: complete sphere, sweep octants 1 through 8

• Binary files generated by **nadump** have file input prefixes + '**.a***processor#*'. WARNING! Be very careful of how much angular flux data you ask for with **nadump**--this requires a great deal of disk space! Since angular fluxes can be dumped based on a coarse mesh, you may want to set up a "thin wall" coarse mesh to obtain a hemisphere of exiting angular fluxes to use for coupling/input for another calculation. Data from multiple processor files can be automatically gathered with the *PENDATA* utility.

4. Appendix

4.1 Angular Quadrature Sets

- Level symmetric quadrature weights are given as point weights; these *initially* sum to 1.0 *for each octant.* Following initial assignment in *PENTRAN,* all weights are then multiplied by 1/8 for normalization on the unit sphere.
- The number of **Ω**'s per octant is (isn*(isn+2)/8), with (isn/2) *distinct* μ's, assuming isn is the discrete ordinates quadrature order.
- Point weights derived from level weights for S14 and above can vary due to more than one positive real root possible to satisfy the criteria for level symmetry; therefore, point weights derived here for S14 and above may or may not differ from those found elsewhere.
- All quadratures in *PENTRAN* were derived with a numerical precision of at least 1.0D-15. A more detailed *PENTRAN* quadrature listing can be obtained using the *PENQUAD* utility.

Example: S6 Level Symmetric   *PENTRAN* **Ω** Sampling order:

(μ>0, η>0, ξ>0)

```
      ξ                       ξ
      1                       1
     2 2                     2 3
    1 2 1                   4 5 6
  μ       η               μ       η
```

6*8/8 = 6 **Ω**'s per octant   6/2 = 3 unique μ's

PENTRAN Assignment of S6 **Ω**'s:

Ω #	μ=μm	η=μm	ξ=μm	w=wm
1	1	1	3	1
2	2	1	2	2
3	1	2	2	2
4	3	1	1	1
5	2	2	1	2
6	1	3	1	1

❑ S6 Level Symmetric Quadrature

```
         S6                  ξ         ξ
  (+ + +) Octant             1         1
  m-Level Diagram           2 2       2 3
  (Sweep 2)                1 2 1     4 5 6
                         μ     η   μ     η
```

wm(1)=0.1761261308633819D0   wm(2)=0.1572072024699513D0

μm(1)=0.2666354015167032D0   μm(2)=0.6815077265365472D0

μm(3)=0.9261809355174899D0

❑ S2 Level Symmetric Quadrature

```
          S2                ξ             ξ
     (+ + +) Octant         1             1
     m-Level Diagram    μ   η         μ   η
     (Sweep 2)
```

wm(1)=1.D0

μm(1)=0.5773502691896257D0

❑ S4 Level Symmetric Quadrature

```
          S4                ξ              ξ
     (+ + +) Octant         1              1
     m-Level Diagram       1 1            2 3
     (Sweep 2)          μ    η         μ    η
```

wm(1)=0.3333333333333333D0

μm(1)=0.3500211745815406D0 μm(2)=0.8688903007222013D0

❑ S8 Level Symmetric Quadrature

```
          S8                       ξ                       ξ
     (+ + +) Octant                1                       1
     m-Level Diagram             2   2                   2   3
     (Sweep 2)                 2   3   2               4   5   6
                             1   2   2   1           7   8   9  10
                           μ              η        μ             η
```

wm(1)=0.1209876543209866D0 wm(2)=0.0907407407407413D0

wm(3)=0.0925925925925926D0

μm(1)=0.2182178902359909D0 μm(2)=0.5773502691896258D0

μm(3)=0.7867957924694435D0 μm(4)=0.9511897312113425D0

❑ S10 Level Symmetric Quadrature

```
          S10               ξ                          ξ
     (+ + +) Octant         1                          1
     m-Level Diagram      2   2                      2   3
     (Sweep 2)          3   4   3                  4   5   6
                      2   4   4   2              7   8   9  10
                    1   2   3   2   1          11  12  13  14  15
                  μ                  η       μ                   η
```

wm(1)=0.08930314798435302D0 wm(2)=0.07252915171236890D0

wm(3)=0.04504376743640288D0 wm(4)=0.05392811448783971D0

μm(1)=0.1893213264780056D0 μm(2)=0.5088817555826185D0

μm(3)=0.6943188875943850D0 μm(4)=0.8397599622366860D0

μm(5)=0.9634909811104704D0

❑ S12 Level Symmetric Quadrature

```
     S12                  ξ                            ξ
(+ + +) Octant            1                             1
m-Level Diagram         2   2                         2   3
(Sweep 2)             3   4   3                     4   5   6
                    3   5   5   3                 7   8   9  10
                  2   4   5   4   2            11  12  13  14  15
                1   2   3   3   2   1        16  17  18  19  20  21
              μ                     η      μ                     η
```

wm(1)=0.07076258997008411D0 wm(2)=0.05588110156489365D0
wm(3)=0.03733767375882513D0 wm(4)=0.05028190106005664D0
wm(5)=0.02585129165575492D0

µm(1)=0.1672126528227026D0 µm(2)=0.4595476346425931D0
µm(3)=0.6280190966421315D0 µm(4)=0.7600210148336660D0
µm(5)=0.8722705430257244D0 µm(6)=0.9716377192513620D0

❑ S14 Level Symmetric Quadrature

```
     S14                    ξ                              ξ
(+ + +) Octant              1                             1
m-Level Diagram           2   2                         2   3
(Sweep 2)               3   5   3                     4   5   6
                      4   6   6   4                   7   8   9  10
                    3   6   7   6   3              11  12  13  14  15
                  2   5   6   6   5   2          16  17  18  19  20  21
                1   2   3   4   3   2   1      22  23  24  25  26  27  28
              μ                         η   μ                          η
```

wm(1)=0.05799704089709301D0 wm(2)=0.04890079763671375D0
wm(3)=0.02214970797116879D0 wm(4)=0.04070085313525794D0
wm(5)=0.03938673868440395D0 wm(6)=0.02455175510137073D0
wm(7)=0.01213253759421592D0

µm(1)=0.1519858614611801D0 µm(2)=0.4221569823048237D0
µm(3)=0.5773502691896257D0 µm(4)=0.6988920867758852D0
µm(5)=0.8022262552313840D0 µm(6)=0.8936910988743190D0
µm(7)=0.9766271529257242D0

❑ S16 Level Symmetric Quadrature

```
     S16               ξ                                  ξ
(+ + +) Octant         1                                  1
m-Level Diagram      2   2                              2   3
(Sweep 2)          3   5   3                          4   5   6
                 4   6   6   4                        7   8   9   10
               4   7   8   7   4                   11  12  13  14  15
             3   6   8   8   6   3               16  17  18  19  20  21
           2   5   6   7   6   5   2           22  23  24  25  26  27  28
         1   2   3   4   4   3   2   1       29  30  31  32  33  34  35  36
       μ                              η    μ                                η
```

wm(1)=0.04898723915796233D0 wm(2)=0.04132959786990422D0
wm(3)=0.02244759769020243D0 wm(4)=0.02440567883044038D0
wm(5)=0.03361864689378468D0 wm(6)=0.01567390172962426D0
wm(7)=0.03692573110461228D0 wm(8)=0.00608816393663137D0

μm(1)=0.1389568750676416D0 μm(2)=0.3922892614447836D0
μm(3)=0.5370965613008739D0 μm(4)=0.6504264506287802D0
μm(5)=0.7467505736146995D0 μm(6)=0.8319965569100706D0
μm(7)=0.9092855009437586D0 μm(8)=0.9805008790117792D0

❑ S18 Level Symmetric Quadrature

```
     S18             ξ                                       ξ
(+ + +) Octant       1                                       1
m-Level Diagram    2   2                                   2   3
(Sweep 2)        3   6   3                               4   5   6
               4   7   7   4                           7   8   9  10
             5   8   9   8   5                      11  12  13  14  15
           4   8  10  10   8   4                  16  17  18  19  20  21
         3   7   9  10   9   7   3              22  23  24  25  26  27  28
       2   6   7   8   8   7   6   2          29  30  31  32  33  34  35  36
     1   2   3   4   5   4   3   2   1     37  38  39  40  41  42  43  44  45
   μ                              η     μ                                     η
```

wm(1)=0.04226464488217149D0 wm(2)=0.03761274738552265D0
wm(3)=0.00669073200689742D0 wm(4)=0.03919193289514417D0
wm(5)=0.00425499717425421D0 wm(6)=0.04236619014295116D0
wm(7)=0.00923962764409376D0 wm(8)=0.01566475086155585D0
wm(9)=0.01365760464592128D0 wm(10)=0.01399031490160748D0

μm(1)=0.1293445045421084D0 μm(2)=0.3680438160525554D0
μm(3)=0.5041651517249193D0 μm(4)=0.6106625499349821D0
μm(5)=0.7011668842525139D0 μm(6)=0.7812561994964660D0
μm(7)=0.8538662066922110D0 μm(8)=0.9207680210618902D0
μm(9)=0.9831276612370913D0

```
     S20          ξ                                       ξ
(+ + +) Octant     1                                        1
m-Level Diagram  2   2                                    2   3
(Sweep 2)      3   6   3                                4   5   6
             4   7   7   4                            7   8   9  10
           5   8   9   8   5                       11  12  13  14  15
         5   9  10  10   9   5                   16  17  18  19  20  21
       4   8  10  11  10   8   4               22  23  24  25  26  27  28
     3   7   9  10  10   9   7   3           29  30  31  32  33  34  35  36
   2   6   7   8   9   8   7   6   2       37  38  39  40  41  42  43  44  45
 1   2   3   4   5   5   4   3   2   1   46  47  48  49  50  51  52  53  54  55
 μ                                  η  μ                                   η
```

wm(1)=0.03702104906169415D0 wm(2)=0.03328421653654888D0
wm(3)=0.01396701489265008D0 wm(4)=0.02908513232048754D0
wm(5)=0.00623193004605474D0 wm(6)=0.02621667000444185D0
wm(7)=0.00228753938881476D0 wm(8)=0.03639912902091424D0
wm(9)=0.00899059601452543D0 wm(10)=0.00297606912156027D0
wm(11)=0.01095707875225638D0

μm(1)=0.1206033430392688D0 μm(2)=0.3475742923164429D0
μm(3)=0.4765192661438829D0 μm(4)=0.5773502691896257D0
μm(5)=0.6630204036531319D0 μm(6)=0.7388225619100911D0
μm(7)=0.8075404016607585D0 μm(8)=0.8708525837599884D0
μm(9)=0.9298639389547678D0 μm(10)=0.9853474855580162D0

4.2 Angular Octant Sweeping Assignments

❑ Octant numbers are assigned on the unit sphere with the signs given for each direction cosine. Also listed are the starting corners where sweeping originates in a Cartesian system. For a more detailed quadrature list, use the *PENQUAD* utility.

Summary of *PENTRAN* GENERAL OCTANT Sweeping Assignments

Sweep	μ	η	ξ	Start	Sweep	μ	η	ξ	Start
1	-	-	-	FWN	2	+	+	+	BES
3	-	-	+	FWS	4	+	+	-	BEN
5	+	-	-	BWN	6	-	+	+	FES
7	+	-	+	BWS	8	-	+	-	FEN

B=Back(-x) F=Front(+x) E=East(-y) W=West(+y) S=South(-z) N=North(+z)

4.3 *PENQUAD*: Supplemental Angular Data

In the event the user requires direct access to all of the angular quadrature data for a particular quadrature set, or would like to print out the entire quadrature set for assigning an angular source distribution in *PENTRAN*, the *PENQUAD* utility has been developed. This utility outputs all of the quadrature data for any level-symmetric set in *PENTRAN* from *Sn*=S_2 to S_{20}, including all evaluated Legendre polynomials through P_7, depending on the user-specified options. A sample output from the *PENQUAD* utility for S_4 quadrature and P_1 Legendre functions is given below ("aphi" is the azimuthal angle in radians):

```
PENQUAD S 4 Level Symmetric Quadrature in PENTRAN 4.xxbeta with P1 Legendre
  i Sweep Omega    w           mu           eta          xi           aphi
  1   1     1  4.16667E-02 -3.50021E-01 -3.50021E-01 -8.68890E-01 -1.95375E+00
    P(0)= 1.00000E+00 P(1)=-3.50021E-01
        P(1,1)*COS(1*aphi)= 3.50021E-01
        P(1,1)*SIN(1*aphi)= 8.68890E-01
  i Sweep Omega    w           mu           eta          xi           aphi
  2   1     2  4.16667E-02 -8.68890E-01 -3.50021E-01 -3.50021E-01 -2.35619E+00
    P(0)= 1.00000E+00 P(1)=-8.68890E-01
        P(1,1)*COS(1*aphi)= 3.50021E-01
        P(1,1)*SIN(1*aphi)= 3.50021E-01
  i Sweep Omega    w           mu           eta          xi           aphi
  3   1     3  4.16667E-02 -3.50021E-01 -8.68890E-01 -3.50021E-01 -2.75864E+00
    P(0)= 1.00000E+00 P(1)=-3.50021E-01
        P(1,1)*COS(1*aphi)= 8.68890E-01
        P(1,1)*SIN(1*aphi)= 3.50021E-01
  i Sweep Omega    w           mu           eta          xi           aphi
  4   2     1  4.16667E-02  3.50021E-01  3.50021E-01  8.68890E-01  1.18785E+00
    P(0)= 1.00000E+00 P(1)= 3.50021E-01
        P(1,1)*COS(1*aphi)=-3.50021E-01
        P(1,1)*SIN(1*aphi)=-8.68890E-01
  i Sweep Omega    w           mu           eta          xi           aphi
  5   2     2  4.16667E-02  8.68890E-01  3.50021E-01  3.50021E-01  7.85398E-01
    P(0)= 1.00000E+00 P(1)= 8.68890E-01
        P(1,1)*COS(1*aphi)=-3.50021E-01
        P(1,1)*SIN(1*aphi)=-3.50021E-01
  i Sweep Omega    w           mu           eta          xi           aphi
  6   2     3  4.16667E-02  3.50021E-01  8.68890E-01  3.50021E-01  3.82950E-01
    P(0)= 1.00000E+00 P(1)= 3.50021E-01
        P(1,1)*COS(1*aphi)=-8.68890E-01
        P(1,1)*SIN(1*aphi)=-3.50021E-01
  i Sweep Omega    w           mu           eta          xi           aphi
  7   3     1  4.16667E-02 -3.50021E-01 -3.50021E-01  8.68890E-01  1.95375E+00
    P(0)= 1.00000E+00 P(1)=-3.50021E-01
        P(1,1)*COS(1*aphi)= 3.50021E-01
        P(1,1)*SIN(1*aphi)=-8.68890E-01
  i Sweep Omega    w           mu           eta          xi           aphi
  8   3     2  4.16667E-02 -8.68890E-01 -3.50021E-01  3.50021E-01  2.35619E+00
    P(0)= 1.00000E+00 P(1)=-8.68890E-01
        P(1,1)*COS(1*aphi)= 3.50021E-01
        P(1,1)*SIN(1*aphi)=-3.50021E-01
  i Sweep Omega    w           mu           eta          xi           aphi
  9   3     3  4.16667E-02 -3.50021E-01 -8.68890E-01  3.50021E-01  2.75864E+00
    P(0)= 1.00000E+00 P(1)=-3.50021E-01
        P(1,1)*COS(1*aphi)= 8.68890E-01
        P(1,1)*SIN(1*aphi)=-3.50021E-01
  i Sweep Omega    w           mu           eta          xi           aphi
 10   4     1  4.16667E-02  3.50021E-01  3.50021E-01 -8.68890E-01 -1.18785E+00
    P(0)= 1.00000E+00 P(1)= 3.50021E-01
        P(1,1)*COS(1*aphi)=-3.50021E-01
        P(1,1)*SIN(1*aphi)= 8.68890E-01
  i Sweep Omega    w           mu           eta          xi           aphi
 11   4     2  4.16667E-02  8.68890E-01  3.50021E-01 -3.50021E-01 -7.85398E-01
    P(0)= 1.00000E+00 P(1)= 8.68890E-01
        P(1,1)*COS(1*aphi)=-3.50021E-01
        P(1,1)*SIN(1*aphi)= 3.50021E-01
  i Sweep Omega    w           mu           eta          xi           aphi
 12   4     3  4.16667E-02  3.50021E-01  8.68890E-01 -3.50021E-01 -3.82950E-01
    P(0)= 1.00000E+00 P(1)= 3.50021E-01
        P(1,1)*COS(1*aphi)=-8.68890E-01
        P(1,1)*SIN(1*aphi)= 3.50021E-01
  i Sweep Omega    w           mu           eta          xi           aphi
 13   5     1  4.16667E-02  3.50021E-01 -3.50021E-01 -8.68890E-01 -1.95375E+00
    P(0)= 1.00000E+00 P(1)= 3.50021E-01
        P(1,1)*COS(1*aphi)= 3.50021E-01
        P(1,1)*SIN(1*aphi)= 8.68890E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 14   5     2  4.16667E-02  8.68890E-01 -3.50021E-01 -3.50021E-01 -2.35619E+00
    P(0)= 1.00000E+00 P(1)= 8.68890E-01
        P(1,1)*COS(1*aphi)= 3.50021E-01
        P(1,1)*SIN(1*aphi)= 3.50021E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 15   5     3  4.16667E-02  3.50021E-01 -8.68890E-01 -3.50021E-01 -2.75864E+00
    P(0)= 1.00000E+00 P(1)= 3.50021E-01
        P(1,1)*COS(1*aphi)= 8.68890E-01
        P(1,1)*SIN(1*aphi)= 3.50021E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 16   6     1  4.16667E-02 -3.50021E-01  3.50021E-01  8.68890E-01  1.18785E+00
    P(0)= 1.00000E+00 P(1)=-3.50021E-01
        P(1,1)*COS(1*aphi)=-3.50021E-01
        P(1,1)*SIN(1*aphi)=-8.68890E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 17   6     2  4.16667E-02 -8.68890E-01  3.50021E-01  3.50021E-01  7.85398E-01
    P(0)= 1.00000E+00 P(1)=-8.68890E-01
        P(1,1)*COS(1*aphi)=-3.50021E-01
        P(1,1)*SIN(1*aphi)=-3.50021E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 18   6     3  4.16667E-02 -3.50021E-01  8.68890E-01  3.50021E-01  3.82950E-01
    P(0)= 1.00000E+00 P(1)=-3.50021E-01
        P(1,1)*COS(1*aphi)=-8.68890E-01
        P(1,1)*SIN(1*aphi)=-3.50021E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 19   7     1  4.16667E-02  3.50021E-01 -3.50021E-01  8.68890E-01  1.95375E+00
    P(0)= 1.00000E+00 P(1)= 3.50021E-01
        P(1,1)*COS(1*aphi)= 3.50021E-01
        P(1,1)*SIN(1*aphi)=-8.68890E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 20   7     2  4.16667E-02  8.68890E-01 -3.50021E-01  3.50021E-01  2.35619E+00
    P(0)= 1.00000E+00 P(1)= 8.68890E-01
        P(1,1)*COS(1*aphi)= 3.50021E-01
        P(1,1)*SIN(1*aphi)=-3.50021E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 21   7     3  4.16667E-02  3.50021E-01 -8.68890E-01  3.50021E-01  2.75864E+00
    P(0)= 1.00000E+00 P(1)= 3.50021E-01
        P(1,1)*COS(1*aphi)= 8.68890E-01
        P(1,1)*SIN(1*aphi)=-3.50021E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 22   8     1  4.16667E-02 -3.50021E-01  3.50021E-01 -8.68890E-01 -1.18785E+00
    P(0)= 1.00000E+00 P(1)=-3.50021E-01
        P(1,1)*COS(1*aphi)=-3.50021E-01
        P(1,1)*SIN(1*aphi)= 8.68890E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 23   8     2  4.16667E-02 -8.68890E-01  3.50021E-01 -3.50021E-01 -7.85398E-01
    P(0)= 1.00000E+00 P(1)=-8.68890E-01
        P(1,1)*COS(1*aphi)=-3.50021E-01
        P(1,1)*SIN(1*aphi)= 3.50021E-01
  i Sweep Omega    w           mu          eta           xi          aphi
 24   8     3  4.16667E-02 -3.50021E-01  8.68890E-01 -3.50021E-01 -3.82950E-01
    P(0)= 1.00000E+00 P(1)=-3.50021E-01
        P(1,1)*COS(1*aphi)=-8.68890E-01
        P(1,1)*SIN(1*aphi)= 3.50021E-01
```

4.4 *PENDATA*: Automated Post-Processing of Massively Parallel Data

The *PENDATA* utility has been developed to permit the user to gather and have selective access to any data field generated by a parallel run. In a simple sense, the *PENDATA* utility removes most of the pain of handling large, massively parallel binary and ASCII output data files from *PENTRAN*. *PENDATA* can *automatically* process *all* output files or a single output file. Fully automated data processing progresses by reading parallel storage information from the decomposition mapping table located at the end of any logfile from a parallel run. Automated or single file selection and gather of any output data, including Coarse Mesh Summary data (from ASCII output files) and any binary stored output data, can be performed using *PENDATA*. The opening screen of *PENDATA* offers the following data manipulation choices, where the user must input an option number [1,8]:

```
                         PENDATA 3.4 SINGLE PRECISION
                    Data Post-Processor for PENTRAN Outputs
                        Supporting PENTRAN Version 8.xx
                                  G. Sjoden
                                 A. Haghighat

                       Penn State Transport Theory Group
                       Department of Nuclear Engineering
                       The Pennsylvania State University
                                 November 1999

                             SCALAR  FLUX Options:
         (1): Get COARSE MESH SUMMARY  Data from a single OUTPUT FILE
         (2): Get Binary FLUX MOMENT   Data from a single OUTPUT FILE
         (3): Get Binary SCALAR SOURCE Data from a single OUTPUT FILE
         (4): Get Logfile & AutoMap PARALLEL COARSE MESH SUMMARY Data
         (5): Get Logfile & AutoMap PARALLEL Binary FLUX MOMENT  Data
         (6): Get Logfile & AutoMap PARALLEL Binary SOURCE       Data

                             ANGULAR FLUX Options:
         (7): Get Binary ANGULAR FLUX  Data from a single OUTPUT FILE
         (8): Get Logfile & AutoMap PARALLEL Binary ANGULAR FLUX Data
```

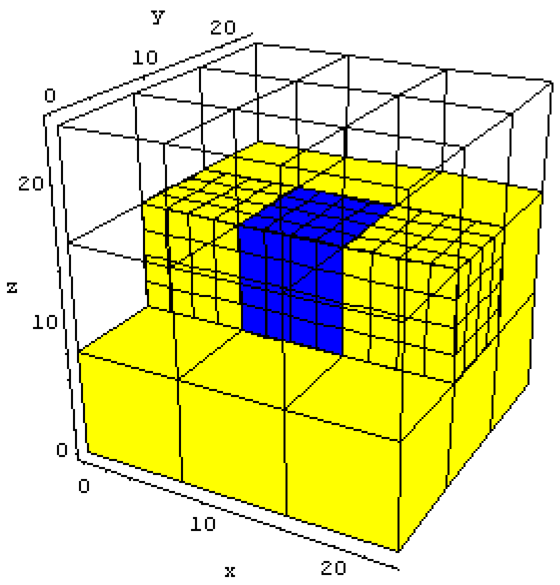
A simple, self explanatory menu format follows each option, showing what data fields can be selectively stripped and automatically gathered. ***If an automated processing is selected, at least one logfile from a processor participating in the parallel run must be located with all other output files in a common directory.*** *PENDATA* will prompt for all required user input, and then report progress as output files are scanned, data is stripped, and collection is made in ASCII table (column) format. The user has freedom to name the output files in *single* file processing. Automated data output files processed and stored by *PENDATA* based on the post-processing choice number:

- Option 4: *fileprefix*.crs Option 5: *fileprefix*.flx Option 6: *fileprefix*.src Option 8: *fileprefix*.ang

Note: In binary flux moment files (Options (2) or (5)), moments are reported by *PENDATA* beginning with a column entitled "flux moments," with flux moments written in columns in the order of scalar, all cosine, and all sine angular moments. Columns therefore include $\phi_{0,}\, \phi_1, \phi^1_{C1,}\, \phi^1_{S1}, \phi_2, \phi^1_{C2}, \phi^2_{C2}, \phi^1_{S2}, \phi^2_{S2}$, etc, as given in equations (2.19) to (2.21) etc. The angular Legendre terms for any quadrature set, if required, can be obtained directly from the *PENQUAD* utility. Note that source files only yield *scalar* sources, and use of a *single precision PENTRAN* version requires a *single precision PENDATA* version for binary data compatability.

4.5 Problem Deck Example: A Simple 3-D Box-in-a-Box

A simple example is always beneficial to describe a code input format. *PENTRAN* has many features and options; in addition, the added complexity of potential parallelism in any one or combinations of all three dimensions (angles, energy groups, and spatial (coarse mesh) cells) forces the user to consider issues that are typically *not* considered with conventional transport codes.



For completeness, we mention here that the *PENTRAN Code System* encomapsses several codes that serve as very useful tools for the user in operating *PENTRAN*. The *PENTRAN* Code System is composed of the following individual codes:

PENMSH --allows the user to automatically design a mesh in 2-D z-plane "slices" called "z-levels" so as to automatically match material boundaries to user defined grid densitities in 3-D, and creates a postscript file of the z-level image.

PENDRW -- renders a two-dimensional z-level image on the PC. A version of PENDRW is already integrated into PENMSH to output a Post-Script image

PENINP --uses the mesh definition files generated by *PENMSH* to render an input deck to allow the user to run the problem in parallel on a set of processors.

PENTRAN – Parallel Environment Neutral Particle Transport Code

PENDATA –A data sorting/handling tool that actively determines the location of a binary file where your information is stored

PENPRL – Performs 3-D interpolation of non-contiguous data; other uses.

Each of these can be obtained through H&S Advanced Computing Technologies.

The "Box-in-a-Box" is given as a simple example of how an input deck might be cast for a problem. This input deck was generated using the *PENMSH* and *PENINP* tools. There is a uniform source in group 1 for this three group problem. The geometry is depicted in the figure at right. There are 27 coarse meshes, each containing a 4x4x4 fine grid structure. The problem was run on a single machine for clarity and compact presentation. Note that PENTRAN will echo the input deck as loaded on processor 1 (the "principal" or "root" processor) as part of the output file. Note: on one processor, about 15 Mb is required. This number drops proportional to the number of processors added in parallel to accomplish the calculation.

The results of this run are provided below (run on a single node K6-2 PC at 380 MHz):

```
                    3-D Transport Code PENTRAN:
               Parallel Environment Neutral particle
                  Transport in Cartesian Geometry
                     with MPI/ANSI FORTRAN-90

                           Version 9.00R

                       Principal Developers:
                            G. Sjoden
                           A. Haghighat

                     Window/Restart Developer:
                          V. Kucukboyaci

                    Penn State Transport Theory Group
                    Department of Nuclear Engineering
                    The Pennsylvania State University
                             April 2000

-----------------------------------------------------------------------------
 Problem Input Deck (As Read):         Memory Required:     14.8 Mb/Processor
 Input Deck file: boxx.ref
-----------------------------------------------------------------------------
 PENTRAN CODE PARAMETERS FOR THIS PROBLEM
  maxmem,  maxpcs,  maxgcm,  maxxsg
      15        1       27        0
  maxcmc,  maxcrs,  maxmmc,  maxmed,  maxfmc,  maxfin
      27        3       64       12       64       12
  maxgrp,  maxglc,  maxswp,  maxqdm,  maxmat,  maxleg
       3        3        8       10        2        1
  maxsrc,  maxslc,  maxcmr,  maxlin,  maxarr,  nctlim
       1        1       27      320      204      133
-----------------Start Problem Deck-------------------
boxx
1 Boxx: Box-in-a-box Simple Problem
2
3
4    G1       G2       G3
5    1.D0     0.0      0.0    bin prob
6
7
8
9
0
/
/-------------BLOCK I (GENERAL PROBLEM info.)-----------
/
  ngeom=3d
  modadj=0
  ngroup=3
  isn=8
  nmatl=2
  ixcrs=3
  jycrs=3
  kzcrs=3
  lodbal=0
  timcut=0.
  tolmgd=.200
  decmpv=1 0 0  T
/
/------------------BLOCK II(geometry)------------------
/
  xmesh=.00    8.00   16.00   24.00
/
/ fine mesh distribution for zlev=   1
/
  ixfine=4   4   4
   4   4   4
   4   4   4
/
/ fine mesh distribution for zlev=   2
/
   4   4   4
   4   4   4
   4   4   4
/
/ fine mesh distribution for zlev=   3
/
   4   4   4
   4   4   4
   4   4   4
/
/   medium mesh distribution for zlev=   1
/
  ixmed=2   2   2
   2   2   2
   2   2   2
/
/   medium mesh distribution for zlev=   2
/
   2   2   2
   2   2   2
   2   2   2
/
/   medium mesh distribution for zlev=   3
/
   2   2   2
   2   2   2
   2   2   2
/
/   coarse-mesh y-position
/
  ymesh=.00    8.00   16.00   24.00
/
/  number of fine-mesh along y for zlev=  1
/
  jyfine=4   4   4
   4   4   4
   4   4   4
/
/
/  number of fine-mesh along y for zlev=  2
/
   4   4   4
   4   4   4
   4   4   4
/
/
/  number of fine-mesh along y for zlev=  3
/
```

```
   4   4   4
   4   4   4
   4   4   4
/
/ number of medium-mesh along y for zlev=  1
/
  jymed=2   2   2
   2   2   2
   2   2   2
/
/ number of medium-mesh along y for zlev=  2
/
   2   2   2
   2   2   2
   2   2   2
/
/ number of medium-mesh along y for zlev=  3
/
   2   2   2
   2   2   2
   2   2   2
/
/   z coarse-mesh boundaries
/
  zmesh=.00    8.00   16.00   24.00
/
/   number of fine-mesh along z, per coarse-mesh for z-level=  1
/
  kzfine=4  4  4
         4  4  4
         4  4  4
/
/   number of fine-mesh along z, per coarse-mesh for z-level=  2
/
         4  4  4
         4  4  4
         4  4  4
/
/   number of fine-mesh along z, per coarse-mesh for z-level=  3
/
         4  4  4
         4  4  4
         4  4  4
/
/    number of medium-mesh along z, per coarse-mesh for z-level=  1
/
  kzmed=2  2  2
        2  2  2
        2  2  2
/
/    number of medium-mesh along z, per coarse-mesh for z-level=  2
/
        2  2  2
        2  2  2
        2  2  2
/
/    number of medium-mesh along z, per coarse-mesh for z-level=  3
/
        2  2  2
        2  2  2
        2  2  2
/
```

```
/  material distribution for zlev=  1
/
    nmattp=1
  4R1      3Q4      3Q16
    nmattp=2
  4R1      3Q4      3Q16
    nmattp=3
  4R1      3Q4      3Q16
    nmattp=4
  4R1      3Q4      3Q16
    nmattp=5
  4R1      3Q4      3Q16
    nmattp=6
  4R1      3Q4      3Q16
    nmattp=7
  4R1      3Q4      3Q16
    nmattp=8
  4R1      3Q4      3Q16
    nmattp=9
  4R1      2Q4
  4R1      3Q16
/
/  material distribution for zlev=  2
/
    nmattp=10
  4R1      3Q4      3Q16
    nmattp=11
  4R1      3Q4      3Q16
    nmattp=12
  4R1      3Q4      3Q16
    nmattp=13
  4R1      3Q4      3Q16
    nmattp=14
  4R2      3Q4      3Q16
    nmattp=15
  4R1      3Q4      3Q16
    nmattp=16
  4R1      3Q4      3Q16
    nmattp=17
  4R1      3Q4      3Q16
    nmattp=18
  4R1      2Q4
  4R1      3Q16
/
/  material distribution for zlev=  3
/
    nmattp=19
  4R1      3Q4      3Q16
    nmattp=20
  4R1      3Q4      3Q16
    nmattp=21
  4R1      3Q4      3Q16
    nmattp=22
  4R1      3Q4      3Q16
    nmattp=23
  4R1      3Q4      3Q16
    nmattp=24
  4R1      3Q4      3Q16
    nmattp=25
  4R1      3Q4      3Q16
    nmattp=26
  4R1      3Q4      3Q16
```

```
    nmattp=27
  4R1      2Q4
  4R1      3Q16
  flxini=27R1.0
  mathmg=27R0   T
/
/ ------------- BLOCK III (CROSS SECTIONS) -----------
/
  lib=file:boxx.xs
  legord=1  legoxs=1
  nxtyp=0
  ihm=8
  iht=3  ihs=6
  ihng=0
  chig=6R1.0
  nxcmnt=1   T
/
/------------- BLOCK IV (CONTROL OPTIONS) --------------
/
  ncoupl=1
  nprtyp=1
  nrdblk=0
  tolin=.000500
  tolout=.000500
  dtwmxw=.96
  maxitr=50
  methit=2
/
/   Starting or selected differencing scheme, for each coarse-mesh, for z-level
/
  ndmeth=2  2  2
         2  2  2
         2  2  2
/
/   Starting or selected differencing scheme, for each coarse-mesh, for z-level
/
         2  2  2
         2  2  2
         2  2  2
/
/   Starting or selected differencing scheme, for each coarse-mesh, for z-level
/
         2  2  2
         2  2  2
         2  2  2
  nzonrb=27 0.99, 0
  methac=0  T
/
/-------------------BLOCK V (Source)--------------------
/
  nsdef=0
  nscmsh=14
  sref=0 0 0
  serg=.1000E+01
       .0000E+00
       .0000E+00
  smag=1R1.0
  spacpf=1  -1      64
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01
   .10000E+01  .10000E+01  .10000E+01  .10000E+01 T
/
/------------- BLOCK VI (BOUNDARY CONDITIONS) ---------
/
/ var   type  Group albedos
  ibback=0
  ibfrnt=0
  jbeast=0
  jbwest=0
  kbsout=0
  kbnort=0       T
/
/------------- BLOCK VII (PRINTING CONDITIONS) ---------
/
/
  nxspr=1 nmatpr=0 ngeopr=1 nsrcpr=0 nsumpr=1
  meshpr=0
  nfdump=1 nsdump=0 nadump=0  T
```

```
-----------------------------------------------------------------------------
 Fixed Source Problem with  1 Fixed Sources Present
 Problem Output/Results:
-----------------------------------------------------------------------------
 Run  Started: 05/06/2000 05:34:04
 Results file: boxx.1
 Header and Title Cards:

boxx
1 Boxx: Box-in-a-box Simple Problem
2
3
4    G1       G2       G3
5    1.D0     0.0      0.0    bin prob
6
7
8
9
0
-----------------------------------------------------------------------------


        GLOBAL PROCESSOR/PROBLEM INFORMATION

 Decomposition Weight Vector (decmpv)**
      Angular:    1.00
        Group:     .00
      Spatial:     .00
 Decomposition Priority Emphasis: Angular
 0 Unique Communicators Built on proc   1.

 (**NEGATIVE Values LOCK-IN the total number of scaled
  processors for the respective decomposition; ZERO Values
  BLOCK decomposition; POSITIVE Values WEIGHT Parallel Scaling)


    PARALLEL VIRTUAL 3-D TOPOLOGY:   1 TOTAL Processor(s)
   Decomposition       Total    Processors   Locally    Total
     Variable        Processed   Allocated  Processed Percentage
-------------------- ---------- ---------- ---------- ----------
Coarse Cells             27          1         27       100.
Energy Groups             3          1          3       100.
Directions Omega         80          1         80       100.

 (Ang Sweep Octants)      8          1          8       100.
 (Omegas/Octant)         10          1         10       100.


Automatic Load Balance Strategy: NONE (Sequential Assignment)
    Red-Black Coloring Strategy: NONE (Local Sequential Assignment)

  Parallel Max/Min Load Ratio in Problem:
          Spatial Coarse Cells: 1.0000E+00
                 Energy Groups: 1.0000E+00
         Angular Sweep Octants: 1.0000E+00
                 Omegas/Octant: 1.0000E+00


       LOCAL PROCESSOR INFORMATION

     Coarse Cells on Processor  1:
     1      2      3      4      5      6      7      8      9     10
    11     12     13     14     15     16     17     18     19     20
    21     22     23     24     25     26     27

    Energy Groups on Processor  1:
     1      2      3

   Angular Sweeps on Processor  1:
     1      5      3      7      8      4      6      2

     Sweep Omegas on Processor  1:
     1      2      3      4      5      6      7      8      9     10

             (See Logfile for Decomposition Mapping)


Geometry: 3d

S8 3D Level Symmetric Angular Quadrature
Number of Omegas per Octant  :  10
ABS Minimum Direction Cosine :  .218217900000000

       (+,+,+) Direction Cosines(in sampling order):

Omega     w (weight)      mu (x-axis)     eta (y-axis)      xi (z-axis)
----- ---------------- ---------------- ---------------- ----------------
  1   .015123460000000 .218217900000000 .218217900000000 .951189800000000
  2   .011342590000000 .577350300000000 .218217900000000 .786795800000000
  3   .011342590000000 .218217900000000 .577350300000000 .786795800000000
  4   .011342590000000 .786795800000000 .218217900000000 .577350300000000
  5   .011574070000000 .577350300000000 .577350300000000 .577350300000000
  6   .011342590000000 .218217900000000 .786795800000000 .577350300000000
  7   .015123460000000 .951189800000000 .218217900000000 .218217900000000
  8   .011342590000000 .786795800000000 .577350300000000 .218217900000000
  9   .011342590000000 .577350300000000 .786795800000000 .218217900000000
 10   .015123460000000 .218217900000000 .951189800000000 .218217900000000

             Omega Sampling Order:

               S8                xi
          (+ + +) Octant       1
          m-Level Diagram    2   3
            (Sweep 2)      4   5   6
                         7   8   9  10
                       mu            eta


       GENERAL OCTANT Sweeping Assignments

  Sweep   mu   eta   xi   Start  Sweep   mu   eta   xi   Start
  ----- ----- ----- ----- -----  ----- ----- ----- ----- -----
    1     -     -     -    FWN     2     +     +     +    BES
    3     -     -     +    FWS     4     +     +     -    BEN
    5     +     -     -    BWN     6     -     +     +    FES
    7     +     -     +    BWS     8     -     +     -    FEN

B=Back(-x) F=Front(+x) E=East(-y) W=West(+y) S=South(-z) N=North(+z)



         3 GROUP/CROSS SECTION Information

Cross Sections: P1 ORDER USED,  P1 read in

     Length of xsec table each Pn : (ihm)   8
     Total Cross Section  position: (iht)   3
     In-group Scatter     position: (ihs)   6
     Last neutron (n,g)   position: (ihng)  0
***WARNING: Upscatter xsecs

Cross sections read from file: boxx.xs
Cross Section   Type: (nxtyp) 0
              Format: STANDARD with (2l+1) multiplied after READ
        Print Switch: (nxspr) 1


Material   1 Cross Sections: P1 order used,  P1 read in

          Material   1 Principal Group Xsecs/Group Chi

Grp     siga_g      nusigf_g      sigt_g          chi_g
--- ------------ ------------ ------------    ------------

  1  7.50000E-02  0.00000E+00  1.25000E-01     1.00000E+00
  2  1.06700E-01  0.00000E+00  1.66700E-01     1.00000E+00
  3  1.00000E-01  0.00000E+00  2.50000E-01     1.00000E+00

          Material   1 Scattering Moments

Grp Pn  sig_G->g,   sig_(G-1)->g...
--- -- ---------------------------------------------------------------------
```

```
  1  0  0.00000E+00  0.00000E+00  1.50000E-02
  2  0  5.00000E-02  3.00000E-02  1.50000E-02
  3  0  1.00000E-01  3.00000E-02  2.00000E-02
  1  1  0.00000E+00  0.00000E+00  3.00000E-05
  2  1  1.50000E-04  0.00000E+00  3.00000E-05
  3  1  0.00000E+00  3.00000E-05  3.00000E-05

***WARNING: Material   1
  GROUP   1 Absorption+Scatter Cross Sections DO NOT sum to the total


Material   2 Cross Sections: P1 order used,  P1 read in

          Material   2 Principal Group Xsecs/Group Chi

Grp     siga_g      nusigf_g      sigt_g          chi_g
--- ------------ ------------ ------------    ------------

  1  7.50000E-03  0.00000E+00  1.25000E-02     1.00000E+00
  2  1.06700E-02  0.00000E+00  1.66700E-02     1.00000E+00
  3  1.00000E-02  0.00000E+00  1.50000E-02     1.00000E+00

          Material   2 Scattering Moments

Grp Pn  sig_G->g,   sig_(G-1)->g...
--- -- ----------------------------------------------------------------

  1  0  0.00000E+00  0.00000E+00  1.50000E-03
  2  0  5.00000E-03  3.00000E-03  1.50000E-03
  3  0  1.00000E-02  3.00000E-03  2.00000E-03
  1  1  0.00000E+00  0.00000E+00  3.00000E-05
  2  1  1.50000E-04  0.00000E+00  3.00000E-05
  3  1  0.00000E+00  3.00000E-05  3.00000E-05

***WARNING: Material   2
  GROUP   3 Absorption+Scatter Cross Sections DO NOT sum to the total


       FIXED SOURCE INFORMATION

       Crs    Coord/Vector Reference                  Integral
No Src Cell  x cm/i   y cm/j   z cm/k  Grp    Source Particles/Bdy Flux
-- --- ---- -------- -------- -------- --- --------------------------------
 1 Vol   14  0.0E+00  0.0E+00  0.0E+00   1  Q.V= 5.1200E+02


Source Dist  Print Switch: (nsrcpr) 0
Binary Source Mesh Dump Switch: (nsdump) 0


       BOUNDARY CONDITION INFORMATION


BACK  (-1) Boundary Condition at x=    .0000:
      VACUUM (zero entering current)

FRONT (-2) Boundary Condition at x=  24.0000:
      VACUUM (zero entering current)

EAST  (-3) Boundary Condition at y=    .0000:
      VACUUM (zero entering current)
```

```
WEST  (-4) Boundary Condition at y=  24.0000:
      VACUUM (zero entering current)

SOUTH (-5) Boundary Condition at z=    .0000:
      VACUUM (zero entering current)

NORTH (-6) Boundary Condition at z=  24.0000:
      VACUUM (zero entering current)


        PENTRAN Sn DIFFERENCING SCHEMES

Diff              Method              Avg Metric   Upgrade  Method
 No       Acronym-Description         Description  Criteria Lock-in
---- -------------------------------  ------------ -------- -------
  0  DD  = Linear Diamond/No Fixup    Not Used     None        0
  1  DZ  = Linear Diamond/Zero Fixup  Fixups/Sweep Fixup      -1
  2  DTW = Directional Theta Weighted MaxWgt/Sweep W= .9600   -2
  3  EDW = Exp-Directional Weighted   DTWuse/Sweep None       -3
  4  EDH = Exp-Direct. Omega Hybrid   DTWuse/Sweep None       -4
  5  EDA = Exp-Directional Averaged   DTWuse/Sweep None       -5


                    MEDIUM Mesh Grid
        COARSE CELL/DIFFERENCING MAPPING TABLE**
Crs     Global    Difference   Mat  In   Out  Rgt  Lft  Bot  Top  X   Y   Z
Cell  i   j   k  Method/Metric Hmg  Bdy  Bdy  Bdy  Bdy  Bdy  Bdy Med Med Med
---- --- --- --- ------------- --- ---- ---- ---- ---- ---- ---- --- --- ---
   1   1   1   1  2DTW 6.0E-01   0   -1    2   -3    4   -5   10   2   2   2
   2   2   1   1  2DTW 6.1E-01   0    1    3   -3    5   -5   11   2   2   2
   3   3   1   1  2DTW 6.0E-01   0    2   -2   -3    6   -5   12   2   2   2
   4   1   2   1  2DTW 6.1E-01   0   -1    5    1    7   -5   13   2   2   2
   5   2   2   1  2DTW 6.2E-01   0    4    6    2    8   -5   14   2   2   2
   6   3   2   1  2DTW 6.1E-01   0    5   -2    3    9   -5   15   2   2   2
   7   1   3   1  2DTW 6.0E-01   0   -1    8    4   -4   -5   16   2   2   2
   8   2   3   1  2DTW 6.1E-01   0    7    9    5   -4   -5   17   2   2   2
   9   3   3   1  2DTW 6.0E-01   0    8   -2    6   -4   -5   18   2   2   2
  10   1   1   2  2DTW 6.1E-01   0   -1   11   -3   13    1   19   2   2   2
  11   2   1   2  2DTW 6.2E-01   0   10   12   -3   14    2   20   2   2   2
  12   3   1   2  2DTW 6.1E-01   0   11   -2   -3   15    3   21   2   2   2
  13   1   2   2  2DTW 6.2E-01   0   -1   14   10   16    4   22   2   2   2
  14   2   2   2  2DTW 7.1E-01   0   13   15   11   17    5   23   2   2   2
  15   3   2   2  2DTW 6.2E-01   0   14   -2   12   18    6   24   2   2   2
  16   1   3   2  2DTW 6.1E-01   0   -1   17   13   -4    7   25   2   2   2
  17   2   3   2  2DTW 6.2E-01   0   16   18   14   -4    8   26   2   2   2
  18   3   3   2  2DTW 6.1E-01   0   17   -2   15   -4    9   27   2   2   2
  19   1   1   3  2DTW 6.0E-01   0   -1   20   -3   22   10   -6   2   2   2
  20   2   1   3  2DTW 6.1E-01   0   19   21   -3   23   11   -6   2   2   2
  21   3   1   3  2DTW 6.0E-01   0   20   -2   -3   24   12   -6   2   2   2
  22   1   2   3  2DTW 6.1E-01   0   -1   23   19   25   13   -6   2   2   2
  23   2   2   3  2DTW 6.2E-01   0   22   24   20   26   14   -6   2   2   2
  24   3   2   3  2DTW 6.1E-01   0   23   -2   21   27   15   -6   2   2   2
  25   1   3   3  2DTW 6.0E-01   0   -1   26   22   -4   16   -6   2   2   2
  26   2   3   3  2DTW 6.1E-01   0   25   27   23   -4   17   -6   2   2   2
  27   3   3   3  2DTW 6.0E-01   0   26   -2   24   -4   18   -6   2   2   2
                                                    XYZ mesh total        216

                     FINE Mesh Grid
        COARSE CELL/DIFFERENCING MAPPING TABLE**

Crs     Global    Difference   Dom  In   Out  Rgt  Lft  Bot  Top  X   Y   Z
Cell  i   j   k  Method/Metric Mtl  Bdy  Bdy  Bdy  Bdy  Bdy  Bdy Fin Fin Fin
```

```
---- --- --- --- ------------ --- ---- ---- ---- ---- ---- ---- --- --- ---
   1   1   1   1  2DTW 6.6E-01   1   -1    2   -3    4   -5   10   4   4   4
   2   2   1   1  2DTW 6.7E-01   1    1    3   -3    5   -5   11   4   4   4
   3   3   1   1  2DTW 6.6E-01   1    2   -2   -3    6   -5   12   4   4   4
   4   1   2   1  2DTW 6.7E-01   1   -1    5    1    7   -5   13   4   4   4
   5   2   2   1  2DTW 6.8E-01   1    4    6    2    8   -5   14   4   4   4
   6   3   2   1  2DTW 6.7E-01   1    5   -2    3    9   -5   15   4   4   4
   7   1   3   1  2DTW 6.6E-01   1   -1    8    4   -4   -5   16   4   4   4
   8   2   3   1  2DTW 6.7E-01   1    7    9    5   -4   -5   17   4   4   4
   9   3   3   1  2DTW 6.6E-01   1    8   -2    6   -4   -5   18   4   4   4
  10   1   1   2  2DTW 6.7E-01   1   -1   11   -3   13    1   19   4   4   4
  11   2   1   2  2DTW 6.8E-01   1   10   12   -3   14    2   20   4   4   4
  12   3   1   2  2DTW 6.7E-01   1   11   -2   -3   15    3   21   4   4   4
  13   1   2   2  2DTW 6.8E-01   1   -1   14   10   16    4   22   4   4   4
  14   2   2   2  2DTW 7.5E-01   2   13   15   11   17    5   23   4   4   4
  15   3   2   2  2DTW 6.8E-01   1   14   -2   12   18    6   24   4   4   4
  16   1   3   2  2DTW 6.7E-01   1   -1   17   13   -4    7   25   4   4   4
  17   2   3   2  2DTW 6.8E-01   1   16   18   14   -4    8   26   4   4   4
  18   3   3   2  2DTW 6.7E-01   1   17   -2   15   -4    9   27   4   4   4
  19   1   1   3  2DTW 6.6E-01   1   -1   20   -3   22   10   -6   4   4   4
  20   2   1   3  2DTW 6.7E-01   1   19   21   -3   23   11   -6   4   4   4
  21   3   1   3  2DTW 6.6E-01   1   20   -2   -3   24   12   -6   4   4   4
  22   1   2   3  2DTW 6.7E-01   1   -1   23   19   25   13   -6   4   4   4
  23   2   2   3  2DTW 6.8E-01   1   22   24   20   26   14   -6   4   4   4
  24   3   2   3  2DTW 6.7E-01   1   23   -2   21   27   15   -6   4   4   4
  25   1   3   3  2DTW 6.6E-01   1   -1   26   22   -4   16   -6   4   4   4
  26   2   3   3  2DTW 6.7E-01   1   25   27   23   -4   17   -6   4   4   4
  27   3   3   3  2DTW 6.6E-01   1   26   -2   24   -4   18   -6   4   4   4
                                                   XYZ mesh total       1728

    **Negative Cell Numbers  Indicate a BOUNDARY CONDITION


       COARSE MESH INFORMATION

       27 Total COARSE Meshes
Number of X Coarse Meshes:   3
 1  Bounded Between x=    .0000 and x=   8.0000 cm
 2  Bounded Between x=   8.0000 and x=  16.0000 cm
 3  Bounded Between x=  16.0000 and x=  24.0000 cm

          Y Coarse Meshes:   3
 1  Bounded Between y=    .0000 and y=   8.0000 cm
 2  Bounded Between y=   8.0000 and y=  16.0000 cm
 3  Bounded Between y=  16.0000 and y=  24.0000 cm

          Z Coarse Meshes:   3
 1  Bounded Between z=    .0000 and z=   8.0000 cm
 2  Bounded Between z=   8.0000 and z=  16.0000 cm
 3  Bounded Between z=  16.0000 and z=  24.0000 cm


 Material Print Switch: (nmatpr) 0



       LOCAL MATERIAL INVENTORY/VOLUME TABLE
       (Local Volume :  1.3824E+04 cm3 in    1728 meshes)

 Material   Local  Number     Local (cm3)  Percent
  Number      of  Meshes       Volume     of Local
 --------  ----------------  -----------  --------
```

```
     0         0 /    1728   0.0000E+00       .0
     1      1664 /    1728   1.3312E+04     96.3
     2        64 /    1728   5.1200E+02      3.7


       CONVERGENCE CRITERIA

Inner Medium Mesh Min Tolerance:  2.0000E-01
Inner  Fine  Mesh Min Tolerance:  5.0000E-04
Outer Medium Mesh Set Tolerance:  5.0000E-04
Outer  Fine  Mesh Set Tolerance:  5.0000E-04
Allowed Max Inner Iter/Group; Outer:   50
Allowed Max Wall-Time:       .0 min

Source iteration Method 2: Hiromoto-Wienke Group Convergence
Acceleration Method 0: (No Acceleration)
Taylor Projection Coupling Order 1: (O(h^2) spatial truncation)

Inner Iterations:
Group   1 CONVERGED in    16 iterations
Group   2 CONVERGED in    16 iterations
Group   3 CONVERGED in    16 iterations
           Total Medium Mesh (inner)  21
           Total  Fine  Mesh (inner)  27
           Total Inner    48


       EXECUTION AND TIMING

Run     Start: 05/06/2000 05:34:04
Problem Start: 05/06/2000 05:34:08
Problem  Stop: 05/06/2000 05:36:02
    Run  Time:     1. min   59.0 sec  (      119.0       sec)
Problem  Time:     1. min   55.2 sec  (      115.2       sec)
CPU  Estimate:      92.9%  Dedicated  (      107.0 total sec)
Cumulative Run:      119.0 sec  Problem:      115.2 sec

       Event %  Using Cumulative RUN Time:
       (Message Other Applies to Other Events)

Proc  File  Setup  Source Xport  Fluxes  Conv   Rebal  Multi  Msg    Msg
 No  Inputs  Prob  Update Sweeps  &Curr in/out  accel  Grid   Sweep  Other
---- ------ ------ ------ ------ ------ ------ ------ ------ ------ ------
  1      .0     .0   26.0   49.6   13.4     .8     .0   15.1     .0     .0

Avg%     .0     .0   26.0   49.6   13.4     .8     .0   15.1     .0     .0
Sec (    .0     .0   31.0   59.0   16.0    1.0     .0   18.0     .0     .0)


       Event  % Using Cumulative PROBLEM Time:
       (Message Other Applies to Other Events)

Proc  File  Setup  Source Xport  Fluxes  Conv   Rebal  Multi  Msg    Msg
 No  Inputs  Prob  Update Sweeps  &Curr in/out  accel  Grid   Sweep  Other
---- ------ ------ ------ ------ ------ ------ ------ ------ ------ ------
  1   ----      .0   26.9   51.2   13.9     .9     .0   15.6     .0     .0

Avg%  ----      .0   26.9   51.2   13.9     .9     .0   15.6     .0     .0
Sec ( ----      .0   31.0   59.0   16.0    1.0     .0   18.0     .0     .0)
```

```
       INTEGRAL BOUNDARY LEAKAGE

Grp  BackNet|-x  FrontNet|+x EastNet|-y  WestNet|+y  SouthNet|-z NorthNet|+z
---- ----------- ----------- ----------- ----------- ----------- -----------
  1  -2.4970E+01  2.4970E+01 -2.4970E+01  2.4970E+01 -2.4970E+01  2.4970E+01
  2  -2.9750E+00  2.9751E+00 -2.9750E+00  2.9751E+00 -2.9750E+00  2.9751E+00
  3  -2.5749E+00  2.5751E+00 -2.5749E+00  2.5751E+00 -2.5749E+00  2.5751E+00

Tot  -3.0520E+01  3.0520E+01 -3.0520E+01  3.0520E+01 -3.0520E+01  3.0520E+01


       INTEGRAL SYSTEM BALANCE

Grp  -(Leakage)  -Collisions +ScatterSrc +FissionSrc +Vol&BdySrc  = Balance
---- ----------- ----------- ----------- ----------- ----------- -----------
  1  -1.4982E+02 -4.1157E+02  4.9388E+01  0.0000E+00  5.1200E+02 -1.2589E-04
  2  -1.7850E+01 -6.3876E+01  8.1727E+01  0.0000E+00  0.0000E+00 -2.2888E-05
  3  -1.5450E+01 -1.0358E+02  1.1903E+02  0.0000E+00  0.0000E+00  0.0000E+00

Tot  -1.8312E+02 -5.7902E+02  2.5015E+02  0.0000E+00  5.1200E+02 -1.4877E-04


Coarse Mesh Summary Switch: (nsumpr) 1


       COARSE MESH SUMMARY DATA  Group   1

 Crs:DomMatl     x-cm        y-cm        z-cm     Volume cm3    MaxHOpt
 Net Current   Phi(Cell)  Q0:Scatter  Q0:Fission  Q0:VolSrc    Error Norm
 -(Leakage)  -Collisions +ScatterSrc +FissionSrc +Vol&BdySrc =  Balance
 ----------- ----------- ----------- ----------- -----------  -----------
     1    1      4.0000      4.0000      4.0000  5.1200E+02   1.1456E+00
 -7.6073E-02  8.6446E-02  1.2967E-03  0.0000E+00  0.0000E+00   2.0661E-07
  4.8686E+00 -5.5326E+00  6.6391E-01  0.0000E+00  0.0000E+00  -5.3644E-07
     2    1     12.0000      4.0000      4.0000  5.1200E+02   1.1456E+00
 -1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   1.8412E-07
  1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00   1.5497E-06
     3    1     20.0000      4.0000      4.0000  5.1200E+02   1.1456E+00
 -7.6073E-02  8.6446E-02  1.2967E-03  0.0000E+00  0.0000E+00   2.0661E-07
  4.8686E+00 -5.5326E+00  6.6391E-01  0.0000E+00  0.0000E+00  -9.5367E-07
     4    1      4.0000     12.0000      4.0000  5.1200E+02   1.1456E+00
 -1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   1.2803E-07
  1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00  -2.3842E-07
     5    1     12.0000     12.0000      4.0000  5.1200E+02   1.1456E+00
 -4.6936E-01  5.3337E-01  8.0005E-03  0.0000E+00  0.0000E+00   1.9455E-07
  3.0039E+01 -3.4136E+01  4.0963E+00  0.0000E+00  0.0000E+00   2.2411E-05
     6    1     20.0000     12.0000      4.0000  5.1200E+02   1.1456E+00
 -1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   1.1180E-07
  1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00   2.5034E-06
     7    1      4.0000     20.0000      4.0000  5.1200E+02   1.1456E+00
 -7.6073E-02  8.6446E-02  1.2967E-03  0.0000E+00  0.0000E+00   2.0661E-07
  4.8686E+00 -5.5326E+00  6.6391E-01  0.0000E+00  0.0000E+00  -5.9605E-08
     8    1     12.0000     20.0000      4.0000  5.1200E+02   1.1456E+00
 -1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   1.1484E-07
  1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00  -3.5763E-07
     9    1     20.0000     20.0000      4.0000  5.1200E+02   1.1456E+00
 -7.6073E-02  8.6446E-02  1.2967E-03  0.0000E+00  0.0000E+00   2.1668E-07
  4.8686E+00 -5.5326E+00  6.6391E-01  0.0000E+00  0.0000E+00  -1.3709E-06
    10    1      4.0000      4.0000     12.0000  5.1200E+02   1.1456E+00
 -1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   2.2968E-07
  1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00  -1.4305E-06
```

```
   11    1      12.0000      4.0000     12.0000  5.1200E+02   1.1456E+00
-4.6936E-01  5.3337E-01  8.0005E-03  0.0000E+00  0.0000E+00   1.1407E-07
 3.0039E+01 -3.4136E+01  4.0963E+00  0.0000E+00  0.0000E+00   1.7643E-05
   12    1      20.0000      4.0000     12.0000  5.1200E+02   1.1456E+00
-1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   1.1484E-07
 1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00   1.3113E-06
   13    1       4.0000     12.0000     12.0000  5.1200E+02   1.1456E+00
-4.6936E-01  5.3337E-01  8.0005E-03  0.0000E+00  0.0000E+00   2.1951E-07
 3.0039E+01 -3.4136E+01  4.0963E+00  0.0000E+00  0.0000E+00   1.1444E-05
   14    2      12.0000     12.0000     12.0000  5.1200E+02   1.1456E-01
 7.6927E+00  3.4921E+00  5.2381E-03  0.0000E+00  1.0000E+00   6.2484E-08
-4.9233E+02 -2.2349E+01  2.6819E+00  0.0000E+00  5.1200E+02  -1.2016E-04
   15    1      20.0000     12.0000     12.0000  5.1200E+02   1.1456E+00
-4.6936E-01  5.3337E-01  8.0005E-03  0.0000E+00  0.0000E+00   1.4634E-07
 3.0039E+01 -3.4136E+01  4.0963E+00  0.0000E+00  0.0000E+00   6.1989E-06
   16    1       4.0000     20.0000     12.0000  5.1200E+02   1.1456E+00
-1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   1.2781E-07
 1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00   2.3842E-07
   17    1      12.0000     20.0000     12.0000  5.1200E+02   1.1456E+00
-4.6936E-01  5.3337E-01  8.0005E-03  0.0000E+00  0.0000E+00   1.2092E-07
 3.0039E+01 -3.4136E+01  4.0963E+00  0.0000E+00  0.0000E+00   5.7220E-06
   18    1      20.0000     20.0000     12.0000  5.1200E+02   1.1456E+00
-1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   1.4016E-07
 1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00   1.1921E-06
   19    1       4.0000      4.0000     20.0000  5.1200E+02   1.1456E+00
-7.6073E-02  8.6446E-02  1.2967E-03  0.0000E+00  0.0000E+00   1.9825E-07
 4.8686E+00 -5.5326E+00  6.6391E-01  0.0000E+00  0.0000E+00  -4.7684E-07
   20    1      12.0000      4.0000     20.0000  5.1200E+02   1.1456E+00
-1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   2.2968E-07
 1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00   2.1458E-06
   21    1      20.0000      4.0000     20.0000  5.1200E+02   1.1456E+00
-7.6073E-02  8.6446E-02  1.2967E-03  0.0000E+00  0.0000E+00   1.1029E-07
 4.8686E+00 -5.5326E+00  6.6391E-01  0.0000E+00  0.0000E+00  -1.0133E-06
   22    1       4.0000     12.0000     20.0000  5.1200E+02   1.1456E+00
-1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   1.4016E-07
 1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00   2.1458E-06
   23    1      12.0000     12.0000     20.0000  5.1200E+02   1.1456E+00
-4.6936E-01  5.3337E-01  8.0005E-03  0.0000E+00  0.0000E+00   1.2092E-07
 3.0039E+01 -3.4136E+01  4.0963E+00  0.0000E+00  0.0000E+00  -1.0490E-05
   24    1      20.0000     12.0000     20.0000  5.1200E+02   1.1456E+00
-1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   1.8411E-07
 1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00   2.2650E-06
   25    1       4.0000     20.0000     20.0000  5.1200E+02   1.1456E+00
-7.6073E-02  8.6446E-02  1.2967E-03  0.0000E+00  0.0000E+00   1.7490E-07
 4.8686E+00 -5.5326E+00  6.6391E-01  0.0000E+00  0.0000E+00  -8.3447E-07
   26    1      12.0000     20.0000     20.0000  5.1200E+02   1.1456E+00
-1.6058E-01  1.8248E-01  2.7372E-03  0.0000E+00  0.0000E+00   1.4040E-07
 1.0277E+01 -1.1679E+01  1.4014E+00  0.0000E+00  0.0000E+00   3.9339E-06
   27    1      20.0000     20.0000     20.0000  5.1200E+02   1.1456E+00
-7.6073E-02  8.6446E-02  1.2967E-03  0.0000E+00  0.0000E+00   1.2690E-07
 4.8686E+00 -5.5326E+00  6.6391E-01  0.0000E+00  0.0000E+00  -8.3447E-07


      COARSE MESH SUMMARY DATA  Group   2

Crs:DomMatl      x-cm        y-cm        z-cm     Volume cm3    MaxHOpt
Net Current   Phi(Cell)  Q0:Scatter  Q0:Fission  Q0:VolSrc    Error Norm
-(Leakage)  -Collisions +ScatterSrc +FissionSrc +Vol&BdySrc =  Balance
----------- ----------- ----------- ----------- -----------  -----------
    1    1       4.0000      4.0000      4.0000  5.1200E+02   1.5278E+00
 5.6582E-04  1.4223E-02  2.4417E-03  0.0000E+00  0.0000E+00   6.8631E-05
-3.6212E-02 -1.2139E+00  1.2501E+00  0.0000E+00  0.0000E+00  -1.1921E-07
```

```
    2    1       12.0000       4.0000       4.0000  5.1200E+02   1.5278E+00
  4.6195E-03  2.5861E-02  4.8885E-03  0.0000E+00  0.0000E+00   1.0394E-04
 -2.9565E-01 -2.2073E+00  2.5029E+00  0.0000E+00  0.0000E+00  -2.9802E-08
    3    1       20.0000       4.0000       4.0000  5.1200E+02   1.5278E+00
  5.6629E-04  1.4224E-02  2.4418E-03  0.0000E+00  0.0000E+00   1.7372E-04
 -3.6243E-02 -1.2140E+00  1.2502E+00  0.0000E+00  0.0000E+00  -7.4506E-08
    4    1        4.0000      12.0000       4.0000  5.1200E+02   1.5278E+00
  4.6195E-03  2.5861E-02  4.8885E-03  0.0000E+00  0.0000E+00   1.0415E-04
 -2.9565E-01 -2.2073E+00  2.5029E+00  0.0000E+00  0.0000E+00  -3.5763E-07
    5    1       12.0000      12.0000       4.0000  5.1200E+02   1.5278E+00
  3.0426E-02  5.2472E-02  1.2550E-02  0.0000E+00  0.0000E+00   1.3793E-04
 -1.9473E+00 -4.4785E+00  6.4258E+00  0.0000E+00  0.0000E+00  -7.1526E-07
    6    1       20.0000      12.0000       4.0000  5.1200E+02   1.5278E+00
  4.6204E-03  2.5862E-02  4.8888E-03  0.0000E+00  0.0000E+00   2.0698E-04
 -2.9570E-01 -2.2074E+00  2.5031E+00  0.0000E+00  0.0000E+00  -4.7684E-07
    7    1        4.0000      20.0000       4.0000  5.1200E+02   1.5278E+00
  5.6629E-04  1.4224E-02  2.4418E-03  0.0000E+00  0.0000E+00   1.7335E-04
 -3.6243E-02 -1.2140E+00  1.2502E+00  0.0000E+00  0.0000E+00   2.9802E-07
    8    1       12.0000      20.0000       4.0000  5.1200E+02   1.5278E+00
  4.6203E-03  2.5862E-02  4.8888E-03  0.0000E+00  0.0000E+00   2.0706E-04
 -2.9570E-01 -2.2074E+00  2.5031E+00  0.0000E+00  0.0000E+00  -4.1723E-07
    9    1       20.0000      20.0000       4.0000  5.1200E+02   1.5278E+00
  5.6677E-04  1.4224E-02  2.4420E-03  0.0000E+00  0.0000E+00   2.6195E-04
 -3.6273E-02 -1.2140E+00  1.2503E+00  0.0000E+00  0.0000E+00  -5.9605E-08
   10    1        4.0000       4.0000      12.0000  5.1200E+02   1.5278E+00
  4.6195E-03  2.5861E-02  4.8885E-03  0.0000E+00  0.0000E+00   1.0404E-04
 -2.9565E-01 -2.2073E+00  2.5029E+00  0.0000E+00  0.0000E+00  -5.3644E-07
   11    1       12.0000       4.0000      12.0000  5.1200E+02   1.5278E+00
  3.0426E-02  5.2472E-02  1.2550E-02  0.0000E+00  0.0000E+00   1.3783E-04
 -1.9473E+00 -4.4785E+00  6.4258E+00  0.0000E+00  0.0000E+00   1.1921E-06
   12    1       20.0000       4.0000      12.0000  5.1200E+02   1.5278E+00
  4.6203E-03  2.5862E-02  4.8888E-03  0.0000E+00  0.0000E+00   2.0741E-04
 -2.9570E-01 -2.2074E+00  2.5031E+00  0.0000E+00  0.0000E+00  -9.5367E-07
   13    1        4.0000      12.0000      12.0000  5.1200E+02   1.5278E+00
  3.0426E-02  5.2472E-02  1.2550E-02  0.0000E+00  0.0000E+00   1.3793E-04
 -1.9473E+00 -4.4785E+00  6.4258E+00  0.0000E+00  0.0000E+00   2.1458E-06
   14    2       12.0000      12.0000      12.0000  5.1200E+02   1.5278E-01
  3.6372E-02  9.4309E-02  6.1186E-03  0.0000E+00  0.0000E+00   9.6270E-05
 -2.3278E+00 -8.0493E-01  3.1327E+00  0.0000E+00  0.0000E+00  -3.5763E-07
   15    1       20.0000      12.0000      12.0000  5.1200E+02   1.5278E+00
  3.0428E-02  5.2473E-02  1.2551E-02  0.0000E+00  0.0000E+00   2.3897E-04
 -1.9474E+00 -4.4786E+00  6.4260E+00  0.0000E+00  0.0000E+00   0.0000E+00
   16    1        4.0000      20.0000      12.0000  5.1200E+02   1.5278E+00
  4.6203E-03  2.5862E-02  4.8888E-03  0.0000E+00  0.0000E+00   2.0708E-04
 -2.9570E-01 -2.2074E+00  2.5031E+00  0.0000E+00  0.0000E+00  -4.1723E-07
   17    1       12.0000      20.0000      12.0000  5.1200E+02   1.5278E+00
  3.0428E-02  5.2473E-02  1.2551E-02  0.0000E+00  0.0000E+00   2.3906E-04
 -1.9474E+00 -4.4786E+00  6.4260E+00  0.0000E+00  0.0000E+00  -4.7684E-07
   18    1       20.0000      20.0000      12.0000  5.1200E+02   1.5278E+00
  4.6212E-03  2.5863E-02  4.8890E-03  0.0000E+00  0.0000E+00   2.9550E-04
 -2.9575E-01 -2.2074E+00  2.5032E+00  0.0000E+00  0.0000E+00  -2.3842E-07
   19    1        4.0000       4.0000      20.0000  5.1200E+02   1.5278E+00
  5.6629E-04  1.4224E-02  2.4418E-03  0.0000E+00  0.0000E+00   1.7372E-04
 -3.6243E-02 -1.2140E+00  1.2502E+00  0.0000E+00  0.0000E+00   2.9802E-07
   20    1       12.0000       4.0000      20.0000  5.1200E+02   1.5278E+00
  4.6204E-03  2.5862E-02  4.8888E-03  0.0000E+00  0.0000E+00   2.0719E-04
 -2.9570E-01 -2.2074E+00  2.5031E+00  0.0000E+00  0.0000E+00  -3.5763E-07
   21    1       20.0000       4.0000      20.0000  5.1200E+02   1.5278E+00
  5.6677E-04  1.4224E-02  2.4420E-03  0.0000E+00  0.0000E+00   2.6216E-04
 -3.6273E-02 -1.2140E+00  1.2503E+00  0.0000E+00  0.0000E+00   1.9372E-07
   22    1        4.0000      12.0000      20.0000  5.1200E+02   1.5278E+00
  4.6203E-03  2.5862E-02  4.8888E-03  0.0000E+00  0.0000E+00   2.0693E-04
```

```
-2.9570E-01 -2.2074E+00  2.5031E+00  0.0000E+00  0.0000E+00  -6.5565E-07
   23    1      12.0000     12.0000     20.0000  5.1200E+02   1.5278E+00
 3.0428E-02  5.2473E-02  1.2551E-02  0.0000E+00  0.0000E+00   2.3887E-04
-1.9474E+00 -4.4786E+00  6.4260E+00  0.0000E+00  0.0000E+00   2.3842E-07
   24    1      20.0000     12.0000     20.0000  5.1200E+02   1.5278E+00
 4.6212E-03  2.5863E-02  4.8890E-03  0.0000E+00  0.0000E+00   2.9559E-04
-2.9575E-01 -2.2074E+00  2.5032E+00  0.0000E+00  0.0000E+00  -5.9605E-08
   25    1       4.0000     20.0000     20.0000  5.1200E+02   1.5278E+00
 5.6677E-04  1.4224E-02  2.4420E-03  0.0000E+00  0.0000E+00   2.6197E-04
-3.6273E-02 -1.2140E+00  1.2503E+00  0.0000E+00  0.0000E+00   7.4506E-08
   26    1      12.0000     20.0000     20.0000  5.1200E+02   1.5278E+00
 4.6212E-03  2.5863E-02  4.8890E-03  0.0000E+00  0.0000E+00   2.9550E-04
-2.9575E-01 -2.2074E+00  2.5032E+00  0.0000E+00  0.0000E+00   3.2783E-07
   27    1      20.0000     20.0000     20.0000  5.1200E+02   1.5278E+00
 5.6725E-04  1.4225E-02  2.4422E-03  0.0000E+00  0.0000E+00   3.5235E-04
-3.6304E-02 -1.2141E+00  1.2504E+00  0.0000E+00  0.0000E+00  -7.0035E-07


      COARSE MESH SUMMARY DATA  Group   3

Crs:DomMatl     x-cm        y-cm        z-cm     Volume cm3    MaxHOpt
Net Current   Phi(Cell)  Q0:Scatter  Q0:Fission  Q0:VolSrc    Error Norm
-(Leakage)  -Collisions +ScatterSrc +FissionSrc +Vol&BdySrc =  Balance
----------- ----------- ----------- ----------- -----------  -----------
    1    1       4.0000      4.0000      4.0000  5.1200E+02   2.2913E+00
 8.4159E-06  1.4365E-02  3.5922E-03  0.0000E+00  0.0000E+00   7.3584E-05
-5.3862E-04 -1.8387E+00  1.8392E+00  0.0000E+00  0.0000E+00  -2.9802E-08
    2    1      12.0000      4.0000      4.0000  5.1200E+02   2.2913E+00
 2.3983E-03  2.7506E-02  7.1763E-03  0.0000E+00  0.0000E+00   1.2098E-04
-1.5349E-01 -3.5208E+00  3.6743E+00  0.0000E+00  0.0000E+00   1.0431E-06
    3    1      20.0000      4.0000      4.0000  5.1200E+02   2.2913E+00
 8.9703E-06  1.4365E-02  3.5925E-03  0.0000E+00  0.0000E+00   2.1323E-04
-5.7410E-04 -1.8388E+00  1.8393E+00  0.0000E+00  0.0000E+00   2.5332E-07
    4    1       4.0000     12.0000      4.0000  5.1200E+02   2.2913E+00
 2.3983E-03  2.7506E-02  7.1763E-03  0.0000E+00  0.0000E+00   1.2054E-04
-1.5349E-01 -3.5208E+00  3.6743E+00  0.0000E+00  0.0000E+00   8.3447E-07
    5    1      12.0000     12.0000      4.0000  5.1200E+02   2.2913E+00
 2.6530E-02  5.9506E-02  1.8193E-02  0.0000E+00  0.0000E+00   1.6524E-04
-1.6979E+00 -7.6167E+00  9.3147E+00  0.0000E+00  0.0000E+00   4.7684E-07
    6    1      20.0000     12.0000      4.0000  5.1200E+02   2.2913E+00
 2.3993E-03  2.7507E-02  7.1768E-03  0.0000E+00  0.0000E+00   2.5838E-04
-1.5355E-01 -3.5210E+00  3.6745E+00  0.0000E+00  0.0000E+00   4.7684E-07
    7    1       4.0000     20.0000      4.0000  5.1200E+02   2.2913E+00
 8.9703E-06  1.4365E-02  3.5925E-03  0.0000E+00  0.0000E+00   2.1317E-04
-5.7410E-04 -1.8388E+00  1.8393E+00  0.0000E+00  0.0000E+00   1.3411E-07
    8    1      12.0000     20.0000      4.0000  5.1200E+02   2.2913E+00
 2.3993E-03  2.7507E-02  7.1768E-03  0.0000E+00  0.0000E+00   2.5790E-04
-1.5355E-01 -3.5210E+00  3.6745E+00  0.0000E+00  0.0000E+00   1.7881E-07
    9    1      20.0000     20.0000      4.0000  5.1200E+02   2.2913E+00
 9.5312E-06  1.4366E-02  3.5927E-03  0.0000E+00  0.0000E+00   3.3400E-04
-6.0999E-04 -1.8389E+00  1.8395E+00  0.0000E+00  0.0000E+00  -3.5763E-07
   10    1       4.0000      4.0000     12.0000  5.1200E+02   2.2913E+00
 2.3983E-03  2.7506E-02  7.1763E-03  0.0000E+00  0.0000E+00   1.2043E-04
-1.5349E-01 -3.5208E+00  3.6743E+00  0.0000E+00  0.0000E+00   5.6624E-07
   11    1      12.0000      4.0000     12.0000  5.1200E+02   2.2913E+00
 2.6530E-02  5.9506E-02  1.8193E-02  0.0000E+00  0.0000E+00   1.6515E-04
-1.6979E+00 -7.6167E+00  9.3147E+00  0.0000E+00  0.0000E+00   9.5367E-07
   12    1      20.0000      4.0000     12.0000  5.1200E+02   2.2913E+00
 2.3993E-03  2.7507E-02  7.1768E-03  0.0000E+00  0.0000E+00   2.5782E-04
-1.5355E-01 -3.5210E+00  3.6745E+00  0.0000E+00  0.0000E+00   4.1723E-07
   13    1       4.0000     12.0000     12.0000  5.1200E+02   2.2913E+00
 2.6530E-02  5.9506E-02  1.8193E-02  0.0000E+00  0.0000E+00   1.6543E-04
 -1.6979E+00 -7.6167E+00  9.3147E+00  0.0000E+00  0.0000E+00  -7.1526E-07
    14    2      12.0000     12.0000     12.0000  5.1200E+02   1.3748E-01
  5.3358E-02  1.1950E-01  8.4622E-03  0.0000E+00  0.0000E+00   1.0519E-04
 -3.4149E+00 -9.1777E-01  4.3326E+00  0.0000E+00  0.0000E+00   2.3842E-07
    15    1      20.0000     12.0000     12.0000  5.1200E+02   2.2913E+00
  2.6532E-02  5.9508E-02  1.8193E-02  0.0000E+00  0.0000E+00   2.9955E-04
 -1.6980E+00 -7.6170E+00  9.3150E+00  0.0000E+00  0.0000E+00   1.6689E-06
    16    1       4.0000     20.0000     12.0000  5.1200E+02   2.2913E+00
  2.3993E-03  2.7507E-02  7.1768E-03  0.0000E+00  0.0000E+00   2.5811E-04
 -1.5355E-01 -3.5210E+00  3.6745E+00  0.0000E+00  0.0000E+00   1.1921E-07
    17    1      12.0000     20.0000     12.0000  5.1200E+02   2.2913E+00
  2.6532E-02  5.9508E-02  1.8193E-02  0.0000E+00  0.0000E+00   2.9974E-04
 -1.6980E+00 -7.6170E+00  9.3150E+00  0.0000E+00  0.0000E+00   1.6689E-06
    18    1      20.0000     20.0000     12.0000  5.1200E+02   2.2913E+00
  2.4002E-03  2.7509E-02  7.1772E-03  0.0000E+00  0.0000E+00   3.7676E-04
 -1.5361E-01 -3.5211E+00  3.6747E+00  0.0000E+00  0.0000E+00   1.1325E-06
    19    1       4.0000      4.0000     20.0000  5.1200E+02   2.2913E+00
  8.9707E-06  1.4365E-02  3.5925E-03  0.0000E+00  0.0000E+00   2.1323E-04
 -5.7413E-04 -1.8388E+00  1.8393E+00  0.0000E+00  0.0000E+00   1.0431E-07
    20    1      12.0000      4.0000     20.0000  5.1200E+02   2.2913E+00
  2.3993E-03  2.7507E-02  7.1768E-03  0.0000E+00  0.0000E+00   2.5804E-04
 -1.5355E-01 -3.5210E+00  3.6745E+00  0.0000E+00  0.0000E+00  -2.0862E-07
    21    1      20.0000      4.0000     20.0000  5.1200E+02   2.2913E+00
  9.5288E-06  1.4366E-02  3.5927E-03  0.0000E+00  0.0000E+00   3.3420E-04
 -6.0984E-04 -1.8389E+00  1.8395E+00  0.0000E+00  0.0000E+00  -2.0862E-07
    22    1       4.0000     12.0000     20.0000  5.1200E+02   2.2913E+00
  2.3993E-03  2.7507E-02  7.1768E-03  0.0000E+00  0.0000E+00   2.5804E-04
 -1.5355E-01 -3.5210E+00  3.6745E+00  0.0000E+00  0.0000E+00   4.4703E-07
    23    1      12.0000     12.0000     20.0000  5.1200E+02   2.2913E+00
  2.6532E-02  5.9508E-02  1.8193E-02  0.0000E+00  0.0000E+00   2.9946E-04
 -1.6980E+00 -7.6170E+00  9.3150E+00  0.0000E+00  0.0000E+00   4.7684E-07
    24    1      20.0000     12.0000     20.0000  5.1200E+02   2.2913E+00
  2.4002E-03  2.7509E-02  7.1772E-03  0.0000E+00  0.0000E+00   3.7640E-04
 -1.5361E-01 -3.5211E+00  3.6747E+00  0.0000E+00  0.0000E+00   7.7486E-07
    25    1       4.0000     20.0000     20.0000  5.1200E+02   2.2913E+00
  9.5293E-06  1.4366E-02  3.5927E-03  0.0000E+00  0.0000E+00   3.3400E-04
 -6.0987E-04 -1.8389E+00  1.8395E+00  0.0000E+00  0.0000E+00  -4.7684E-07
    26    1      12.0000     20.0000     20.0000  5.1200E+02   2.2913E+00
  2.4002E-03  2.7509E-02  7.1772E-03  0.0000E+00  0.0000E+00   3.7676E-04
 -1.5361E-01 -3.5211E+00  3.6747E+00  0.0000E+00  0.0000E+00   1.4901E-07
    27    1      20.0000     20.0000     20.0000  5.1200E+02   2.2913E+00
  1.0084E-05  1.4367E-02  3.5930E-03  0.0000E+00  0.0000E+00   4.5395E-04
 -6.4535E-04 -1.8390E+00  1.8396E+00  0.0000E+00  0.0000E+00  -5.5134E-07


Binary Flux  Moment Dump Switch: (nfdump) 1

Binary Angular Flux Dump Switch: (nadump) 0


 Problem Output Complete:    05/06/2000 05:36:02   on proc   1
 Out I/O:       .0 sec on proc   1
 Cumulative Problem Time      115.2 sec on proc   1
```

The Logfile for the "Box-in-a-Box" problem is provided below:

```
 PENTRAN 3-D Parallel Discrete Ordinates Code Version 9.00R
 LOGFILE FOR PROBLEM : boxx.ref

 FIDO-COMM SCRATCH FILE: boxx1.dat
 PROCESSOR    1 Processed FIDO Inputs:
 ---------------------------------------------------

 COMPILED PARAMETER SETTINGS/PROBLEM LIMITS:

 Max Proc Memory Required (Mb/Proc):    14.8

 Max Proc Memory Limit, Mb (maxmem):    15
 Max No. Procs Allowed For (maxpcs):     1
 Max Global Coarse Meshes  (maxgcm):    27

 Max Restart Total Groups  (maxxsg):     0
 Max Local  Coarse Meshes  (maxcmc):    27
 Max Coarse Meshes/1 axis  (maxcrs):     3
 Max Medium Meshes/Coarse  (maxmmc):    64
 Max Medium Meshes/1 axis  (maxmed):    12
 Max Fine   Meshes/Coarse  (maxfmc):    64
 Max Fine   Meshes/1 axis  (maxfin):    12

 Max Global Energy Groups  (maxgrp):     3
 Max Local  Energy Groups  (maxglc):     3
 Max Local  Sweep Octants  (maxswp):     8
 Max Quad.  Angles/Octant  (maxqdm):    10
 Max Material (xsec)Types  (maxmat):     2
 Max Leg Scat dimensioned  (maxleg):     1

 Max Global Fixed Sources  (maxsrc):     1
 Max Local  Fixed Sources  (maxslc):     1
 Max Contiguous CMR Cells  (maxcmr):    27
 Max Lines  in Input Deck  (maxlin):   320
 Max Inputs vector w/FIDO  (maxarr):   204
 Max FIDO  Chars/Variable  (nctlim):   133

 ---------------------------------------------------

  PENTRAN VARIABLE NAMES BY BLOCK:

   BLOCK 1 - 12 Possible:
ngeom ngroup isn nmatl ixcrs jycrs kzcrs decmpv lodbal timcut tolmgd modadj

   BLOCK 2 - 12 Possible:
xmesh ixmed ixfine ymesh jymed jyfine zmesh kzmed kzfine nmattp flxini mathmg

   BLOCK 3 - 10 Possible:
lib legord nxtyp ihm iht ihs ihng chig nxcmnt legoxs

   BLOCK 4 - 12 Possible:
nprtyp nrdblk tolin tolout maxitr methit
                                     methac ncoupl ndmeth nzonrb dtwmxw nquit

   BLOCK 5 - 10 Possible:
nsdef nscmsh ssnrm sref serg smag rkdef spacpf omegap scalsf

   BLOCK 6 -  6 Possible:
```

```
ibback ibfrnt jbeast jbwest kbsout kbnort

   BLOCK 7 -  9 Possible:
nxspr ngeopr nsumpr meshpr nfdump nsrcpr nsdump nmatpr nadump

 ---------------------------------------------------

>>> Scanning Data  on Processor    1 of    1 <<<


#### Scanning Block  1 Data on Processor    1 of    1 ####

   ngeom= 3d

  modadj=  0 (forward/adjoint switch)

   ngroup:           1 values read
  ngroup(1)=  3 (Total energy groups--current ru
  ngroup(2)=  3 (Group processor dimension window)
  ngroup(3)=  0 (Converged groups from Restart)

     isn=  8 (Sn level-symmetric quad order)

   nmatl=  2 (Materials in problem)

   ixcrs=  3 (No. Coarse Meshes in x)

   jycrs=  3 (No. Coarse Meshes in y)

   kzcrs=  3 (No. Coarse Meshes in z)

  lodbal=  0 (auto-parallel load balance switch)

  timcut=      .0 (Inactive wall mins cutoff)

  tolmgd= 2.0000E-01 (Multi-Grid transport convergence tolerance)

  decmpv=Decomp Vector Priorities: Omega,   Group,   Cell
                                   1.00      .00      .00

#### Scanning Block  2 Data on Processor    1 of    1 ####

    xmesh:           4 values read: (x coarse bounds)
  0.0000E+00  8.0000E+00  1.6000E+01  2.4000E+01

   ixfine:          27 values read: (x fine meshes)
    4    4    4    4    4    4    4    4    4    4
    4    4    4    4    4    4    4    4    4    4
    4    4    4    4    4    4    4

    ixmed:          27 values read: (x medium meshes)
    2    2    2    2    2    2    2    2    2    2
    2    2    2    2    2    2    2    2    2    2
    2    2    2    2    2    2    2

    ymesh:           4 values read: (y coarse bounds)
  0.0000E+00  8.0000E+00  1.6000E+01  2.4000E+01

   jyfine:          27 values read: (y fine meshes)
    4    4    4    4    4    4    4    4    4    4
    4    4    4    4    4    4    4    4    4    4
    4    4    4    4    4    4    4
    jymed:          27 values read: (y medium meshes)
    2    2    2    2    2    2    2    2    2    2
    2    2    2    2    2    2    2    2    2    2
    2    2    2    2    2    2    2

    zmesh:           4 values read: (z coarse bounds)
  0.0000E+00  8.0000E+00  1.6000E+01  2.4000E+01

   kzfine:          27 values read: (z fine meshes)
    4    4    4    4    4    4    4    4    4    4
    4    4    4    4    4    4    4    4    4    4
    4    4    4    4    4    4    4

    kzmed:          27 values read: (z medium meshes)
    2    2    2    2    2    2    2    2    2    2
    2    2    2    2    2    2    2    2    2    2
    2    2    2    2    2    2    2

   READ nmattp: Coarse           1;          64 mesh material specs read
     data to material scratch file: boxx.M1


   READ nmattp: Coarse           2;          64 mesh material specs read
     data to material scratch file: boxx.M1


   READ nmattp: Coarse           3;          64 mesh material specs read
     data to material scratch file: boxx.M1


   READ nmattp: Coarse           4;          64 mesh material specs read
     data to material scratch file: boxx.M1


   READ nmattp: Coarse           5;          64 mesh material specs read
     data to material scratch file: boxx.M1


   READ nmattp: Coarse           6;          64 mesh material specs read
     data to material scratch file: boxx.M1


   READ nmattp: Coarse           7;          64 mesh material specs read
     data to material scratch file: boxx.M1


   READ nmattp: Coarse           8;          64 mesh material specs read
     data to material scratch file: boxx.M1


   READ nmattp: Coarse           9;          64 mesh material specs read
     data to material scratch file: boxx.M1


   READ nmattp: Coarse          10;          64 mesh material specs read
     data to material scratch file: boxx.M1


   READ nmattp: Coarse          11;          64 mesh material specs read
     data to material scratch file: boxx.M1
```

```
READ nmattp: Coarse          12;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          13;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          14;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          15;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          16;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          17;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          18;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          19;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          20;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          21;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          22;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          23;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          24;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          25;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          26;          64 mesh material specs read
  data to material scratch file: boxx.M1


READ nmattp: Coarse          27;          64 mesh material specs read
```

```
     data to material scratch file: boxx.M1


  ***WARNING Block 2: Initial fluxes flxini weighted
 in energy by chig in Block 3

   flxini:          27 values read: (initial coarse mesh fluxes)
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00

   mathmg:          27 values read: (Coarse Mesh medium grid matl setting)
    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0



#### Scanning Block  3 Data on Processor    1 of    1 ####

     lib=file:boxx.xs
         (xsec lib source)

  legord=1 (Legendre order of calculation)

  legoxs=1 (Legendre order of xsec library)

   nxtyp=0 (cross section type)

     ihm=  8 (xsec table length)

     iht=  3 (total xsec position)

     ihs=  6 (in-group scatter xsec position)

    ihng=  0 (table marker of last n in n,gamma xsec)

  chig for material:  1 for each of   3 groups:
  1.0000E+00  1.0000E+00  1.0000E+00

  chig for material:  2 for each of   3 groups:
  1.0000E+00  1.0000E+00  1.0000E+00

  nxcmnt=  1 (No. xsec comment cards before matl xsec)

  ***WARNING Block 3:          1 Comment lines/xsec indicated for cross section
 file:
boxx.xs


 Reading Block 3  Cross Section Data on Processor    1 of    1
  nxtyp 0: STANDARD form WITHOUT Legendre Constants
  Library from File: boxx.xs

 READ xsecs: Material   1

 READ xsecs: Material   2
```

```
#### Scanning Block  4 Data on Processor    1 of    1 ####

  ncoupl=  1 (Taylor projection coupling order)

  nprtyp=  1 (problem type)

  nrdblk=  0 (automatic Red-Black subdomain switch)
    tolin: 1 value read: (all coarse cell tolerances)
    5.000000E-04

   tolout:           1 values read
  tolout(1)= 5.000E-04 (outer iteration tolerance)
  tolout(2)= 1.000E+00 (tolout(1) multiplier for med grid)

  dtwmxw= .9600 (max DTW weight for adaptive)

   maxitr:           1 values read
  maxitr(1)=  50 (max inner/outer iterations)
  maxitr(2)=  50 (keff inner iteration limit)

  methit=  2 (source iteration method)

   ndmeth:          27 values read: (Coarse Mesh Sn diff methods)
    2    2    2    2    2    2    2    2    2    2
    2    2    2    2    2    2    2    2    2    2
    2    2    2    2    2    2    2

   nzonrb:           3 values read:
  (Cells/Zone: Dampf: Skip:  )
  nzonrb=   27  .9900     0

   methac:           1 values read:
  methac(1)=  0 (rebalance accel method)
  methac(2)=  0 (csda init Off=0/Point=1/Cosine=2)
  methac(3)=    .0000  (csda xcenter (0=auto))
  methac(4)=    .0000  (csda ycenter (0=auto))
  methac(5)=    .0000  (csda zcenter (0=auto))
  methac(6)=  2 (inner iterate use of csda)
  methac(7)=  1 (outer iterate use of csda)


  ***WARNING Block 4: nquit; no maximum
  #iters to stop a non-converging group
  --Using Default Value nquit=4



#### Scanning Block  5 Data on Processor    1 of    1 ####

    nsdef:           1 values read: (source types)
    0

    nscmsh:           1 values read: (source Coarse Mesh numbers)
   14

    sref:          3 total values read
   sref--for source:   1  global sampling reference pt (x y z):
  0.0000E+00  0.0000E+00  0.0000E+00

    serg:          3 total values read
   serg--for source:   1            energy (1..G) distribution:
  1.0000E+00  0.0000E+00  0.0000E+00
```

```
      smag: for sources 1 to          1
  1.0000E+00

  Source   1  spatial  fine  grid dist data:
    Group   1  No. fine cells     64
***(Applies to all Groups)
  spacpf--for Source:   1 Group   1 normalized  fine  cell probabilities:
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00
  1.0000E+00  1.0000E+00  1.0000E+00  1.0000E+00

  ***WARNING Block 5: Un-normalized  fine  spatial probability
  distribution detected in spacpf:
     Source   1 Group   1  Fine Cell dist sum  6.4000E+01
  Fine Spatial Normalization DISABLED


#### Scanning Block  6 Data on Processor    1 of    1 ####

  ibback=  0 (bdy cond at surf normal to -x)

  ibfrnt=  0 (bdy cond at surf normal to +x)

  jbeast=  0 (bdy cond at surf normal to -y)

  jbwest=  0 (bdy cond at surf normal to +y)

  kbsout=  0 (bdy cond at surf normal to -z)

  kbnort=  0 (bdy cond at surf normal to +z)


#### Scanning Block  7 Data on Processor    1 of    1 ####

   nxspr=  1 (xsec print in outputs)

  nmatpr=  0 (Coarse Mesh material map output switch)

  ngeopr=  1 (geometry print setting)

  nsrcpr=  0 (source dist data output switch)

  nsumpr=  1 (Coarse Mesh summary print switch)

   meshpr:           1 values read: (Coarse Mesh print settings)
    0

  nfdump=  1 (binary flux moment dump setting

  nsdump=  0 (binary scalar source dump switch)
   nadump:           1 values read: (binary angular flux dump setting)
    0


  ***WARNING Block 7: meshpr; 0 cells to output detailed mesh
  Output Restricted.



           5 WARNING(S) DETECTED
           0 FATAL ERROR(S) DETECTED


 Input Complete for Processor    1 of    1

 Mapping Processors/Decomposition Groups:
      1 of    1 (Total) procs

        GLOBAL PROCESSOR/PROBLEM INFORMATION

 Decomposition Weight Vector (decmpv)
      Angular:    1.00  (Angular Utilization:  1.000)
        Group:     .00  (Group   Utilization:  1.000)
      Spatial:     .00  (Spatial Utilization:  1.000)
 Decomposition Priority Emphasis: Angular


      PARALLEL VIRTUAL 3-D TOPOLOGY:   1 TOTAL Processor(s)

    Decomposition       Total    Processors   Locally    Total
      Variable        Processed   Allocated  Processed Percentage
 -------------------- ---------- ---------- ---------- ----------
 Coarse Cells              27          1         27        100.
 Energy Groups              3          1          3        100.
 Directions Omega          80          1         80        100.

  (Ang Sweep Octants)       8          1          8        100.
  (Omegas/Octant)          10          1         10        100.


 Automatic Load Balance Strategy: NONE (Sequential Assignment)
     Red-Black Coloring Strategy: NONE (Local Sequential Assignment)

   Parallel Max/Min Load Ratio in Problem:
           Spatial Coarse Cells: 1.0000E+00
                  Energy Groups: 1.0000E+00
          Angular Sweep Octants: 1.0000E+00
                  Omegas/Octant: 1.0000E+00

 Building Mesh on proc    1.
 Building Materials on proc    1.

 Locating/Logging Sources on proc    1.
 Initializing Fluxes, flags on proc    1.
 Integrating Fixed Sources on proc    1.
  WARNING: Group UPSCATTER Cross Secs Indicated
 Entering Hiromoto-Wienke Group/Sweep Routine on proc    1.

 Initializing Multigrid Parameters on proc    1.

   1m First Iteration Complete  G=  1 05/06/2000 05:34:10
   1m i=   1 G=  1 Err=1.00E+00 C=  27.     1 SR=1.00 T=   0 34:10 DM 2    .0s
   1m First Iteration Complete  G=  2 05/06/2000 05:34:10
   1m i=   1 G=  2 Err=1.00E+00 C=  27.     1 SR=1.00 T=   0 34:10 DM 2    .0s
   1m First Iteration Complete  G=  3 05/06/2000 05:34:10
   1m i=   1 G=  3 Err=1.00E+00 C=  27.     1 SR=1.00 T=   0 34:10 DM 2    .0s
   1m i=   2 G=  1 Err=5.00E-01 C=   1.     1 SR= .34 T=   1 34:12 DM 2   1.3s
   1m i=   2 G=  2 Err=6.95E-01 C=  25.     7 SR= .63 T=   3 34:12 DM 2   1.3s
   1m i=   2 G=  3 Err=6.96E-01 C=  21.     6 SR= .62 T=   3 34:12 DM 2   1.3s
   1m i=   3 G=  1 Err=5.26E-02 C=   1.     1 SR= .10 T=   0 34:13 DM 2    .9s
   1m i=   3 G=  2 Err=6.22E-01 C=  27.     8 SR= .89 T=  13 34:13 DM 2    .9s
   1m i=   3 G=  3 Err=6.42E-01 C=  27.     8 SR= .92 T=  18 34:13 DM 2    .9s
   1m i=   4 G=  1 Err=3.45E-03 C=  27.     8 SR= .07 T=   0 34:14 DM 2   1.0s
   1m i=   4 G=  2 Err=5.21E-01 C=  19.     5 SR= .82 T=   7 34:14 DM 2   1.0s
   1m i=   4 G=  3 Err=5.56E-01 C=  25.     7 SR= .84 T=   9 34:15 DM 2   1.3s
   1m i=   5 G=  1 Err=2.45E-04 C=  19.     6 SR= .08 T=   0 34:16 DM 2   1.3s
   1m i=   5 G=  2 Err=3.77E-01 C=   7.     3 SR= .70 T=   4 34:16 DM 2   1.3s
   1m i=   5 G=  3 Err=4.22E-01 C=   9.     4 SR= .73 T=   5 34:16 DM 2   1.3s
   1m i=   6 G=  1 Err=1.86E-05 C=   9.     3 SR= .08 T=   0 34:17 DM 2   1.3s
   1m i=   6 G=  2 Err=2.22E-01 C=  25.     7 SR= .58 T=   2 34:17 DM 2   1.3s
   1m i=   6 G=  3 Err=2.63E-01 C=   1.     1 SR= .60 T=   3 34:17 DM 2   1.3s
   1m i=   7 G=  1 Err=2.31E-06 C=   3.     3 SR= .12 T=   0 34:19 DM 2   1.3s
   1m i=   7 G=  2 Err=1.10E-01 C=   1.     1 SR= .49 T=   2 34:19 DM 2   1.3s
   1m i=   7 G=  3 Err=1.34E-01 C=  19.     5 SR= .50 T=   2 34:19 DM 2   1.3s
 INNER LOOP COMPLETE on Processor         1
 INNER Medium Grid COMPLETE on ALL Processors
   1f i=   8 G=  1 Err=7.42E+00 C=  23.     7 SR= .50 T=  23 34:38 DM 2   3.5s
   1f i=   8 G=  2 Err=4.66E+00 C=  25.     7 SR= .50 T=  23 34:38 DM 2   3.5s
   1f i=   8 G=  3 Err=5.41E+00 C=  25.     7 SR= .50 T=  23 34:38 DM 2   3.5s
   1f i=   9 G=  1 Err=5.03E-02 C=  26.    60 SR= .02 T=  21 34:48 DM 2   4.3s
   1f i=   9 G=  2 Err=1.73E-01 C=  24.    64 SR= .07 T=  21 34:48 DM 2   4.3s
   1f i=   9 G=  3 Err=1.80E-01 C=  26.    64 SR= .07 T=  21 34:48 DM 2   4.3s
   1f i=  10 G=  1 Err=3.91E-03 C=  23.    64 SR= .08 T=  21 34:59 DM 2   4.9s
   1f i=  10 G=  2 Err=7.82E-02 C=  27.    44 SR= .40 T=  22 34:59 DM 2   4.9s
   1f i=  10 G=  3 Err=9.19E-02 C=  27.    48 SR= .45 T=  22 34:59 DM 2   4.9s
   1f i=  11 G=  1 Err=3.13E-04 C=  27.    43 SR= .08 T=  21 35:09 DM 2   5.4s
   1f i=  11 G=  2 Err=3.38E-02 C=  27.    43 SR= .41 T=  22 35:09 DM 2   5.4s
   1f i=  11 G=  3 Err=4.18E-02 C=  27.    43 SR= .42 T=  22 35:09 DM 2   5.4s
   1f i=  12 G=  1 Err=2.41E-05 C=  27.    43 SR= .08 T=  21 35:19 DM 2   5.8s
   1f i=  12 G=  2 Err=1.40E-02 C=  27.    43 SR= .42 T=  22 35:19 DM 2   5.8s
   1f i=  12 G=  3 Err=1.77E-02 C=  27.    43 SR= .41 T=  22 35:19 DM 2   5.8s
   1f i=  13 G=  1 Err=1.87E-06 C=  27.    42 SR= .10 T=  21 35:30 DM 2   6.2s
   1f i=  13 G=  2 Err=5.64E-03 C=  27.    43 SR= .42 T=  22 35:30 DM 2   6.2s
   1f i=  13 G=  3 Err=7.23E-03 C=  27.    43 SR= .43 T=  22 35:30 DM 2   6.2s
   1f i=  14 G=  1 Err=6.41E-07 C=   1.    30 SR= .24 T=  22 35:40 DM 2   6.4s
   1f i=  14 G=  2 Err=2.25E-03 C=  27.    43 SR= .42 T=  22 35:40 DM 2   6.4s
   1f i=  14 G=  3 Err=2.89E-03 C=  27.    43 SR= .42 T=  22 35:40 DM 2   6.4s
   1f i=  15 G=  1 Err=2.31E-07 C=  24.    35 SR= .49 T=  23 35:50 DM 2   6.7s
   1f i=  15 G=  2 Err=8.91E-04 C=  27.    43 SR= .42 T=  22 35:50 DM 2   6.7s
   1f i=  15 G=  3 Err=1.15E-03 C=  27.    43 SR= .42 T=  22 35:50 DM 2   6.7s
   1f i=  16 G=  1 Err=2.30E-07 C=  10.     2 SR= .82 T=  29 36:01 DM 2   7.0s
   1f i=  16 G=  2 Err=3.52E-04 C=  27.    43 SR= .41 T=  22 36:01 DM 2   7.0s
   1f i=  16 G=  3 Err=4.54E-04 C=  27.    43 SR= .42 T=  22 36:01 DM 2   7.0s
 INNER LOOP COMPLETE on Processor         1
 INNER  Fine  Grid COMPLETE on ALL Processors


         Processor   1 of   1 DECOMPOSITION Mapping


 Processor  Global  Global  Global  Sweep    File
  Number    Coarse  Group   Sweep   Omega   Output.proc
 ---------  ------  ------  ------  ------  ------------
```

```
     1      C   1   G  1    S 1-8  O 1-10   boxx.      1
     1      C   1   G  2    S 1-8  O 1-10   boxx.      1
     1      C   1   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep   File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C   2   G  1    S 1-8  O 1-10   boxx.      1
     1      C   2   G  2    S 1-8  O 1-10   boxx.      1
     1      C   2   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep   File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C   3   G  1    S 1-8  O 1-10   boxx.      1
     1      C   3   G  2    S 1-8  O 1-10   boxx.      1
     1      C   3   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep   File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C   4   G  1    S 1-8  O 1-10   boxx.      1
     1      C   4   G  2    S 1-8  O 1-10   boxx.      1
     1      C   4   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep   File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C   5   G  1    S 1-8  O 1-10   boxx.      1
     1      C   5   G  2    S 1-8  O 1-10   boxx.      1
     1      C   5   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep   File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C   6   G  1    S 1-8  O 1-10   boxx.      1
     1      C   6   G  2    S 1-8  O 1-10   boxx.      1
     1      C   6   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep   File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C   7   G  1    S 1-8  O 1-10   boxx.      1
     1      C   7   G  2    S 1-8  O 1-10   boxx.      1
     1      C   7   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep   File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C   8   G  1    S 1-8  O 1-10   boxx.      1
     1      C   8   G  2    S 1-8  O 1-10   boxx.      1
     1      C   8   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep   File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C   9   G  1    S 1-8  O 1-10   boxx.      1
     1      C   9   G  2    S 1-8  O 1-10   boxx.      1
     1      C   9   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep   File
 Number    Coarse  Group   Sweep   Omega   Output.proc
```

```
---------  ------  ------  ------  ------  -------------
     1      C  10   G  1    S 1-8  O 1-10   boxx.      1
     1      C  10   G  2    S 1-8  O 1-10   boxx.      1
     1      C  10   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep    File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C  11   G  1    S 1-8  O 1-10   boxx.      1
     1      C  11   G  2    S 1-8  O 1-10   boxx.      1
     1      C  11   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep    File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C  12   G  1    S 1-8  O 1-10   boxx.      1
     1      C  12   G  2    S 1-8  O 1-10   boxx.      1
     1      C  12   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep    File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C  13   G  1    S 1-8  O 1-10   boxx.      1
     1      C  13   G  2    S 1-8  O 1-10   boxx.      1
     1      C  13   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep    File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C  14   G  1    S 1-8  O 1-10   boxx.      1
     1      C  14   G  2    S 1-8  O 1-10   boxx.      1
     1      C  14   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep    File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C  15   G  1    S 1-8  O 1-10   boxx.      1
     1      C  15   G  2    S 1-8  O 1-10   boxx.      1
     1      C  15   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep    File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C  16   G  1    S 1-8  O 1-10   boxx.      1
     1      C  16   G  2    S 1-8  O 1-10   boxx.      1
     1      C  16   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep    File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C  17   G  1    S 1-8  O 1-10   boxx.      1
     1      C  17   G  2    S 1-8  O 1-10   boxx.      1
     1      C  17   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep    File
 Number    Coarse  Group   Sweep   Omega   Output.proc
---------  ------  ------  ------  ------  -------------
     1      C  18   G  1    S 1-8  O 1-10   boxx.      1
     1      C  18   G  2    S 1-8  O 1-10   boxx.      1
     1      C  18   G  3    S 1-8  O 1-10   boxx.      1

Processor  Global  Global  Global  Sweep    File
```

```
  Number    Coarse  Group   Sweep   Omega   Output.proc
 ---------  ------  ------  ------  ------  -------------
      1      C  19   G  1    S 1-8  O 1-10   boxx.      1
      1      C  19   G  2    S 1-8  O 1-10   boxx.      1
      1      C  19   G  3    S 1-8  O 1-10   boxx.      1

 Processor  Global  Global  Global  Sweep    File
  Number    Coarse  Group   Sweep   Omega   Output.proc
 ---------  ------  ------  ------  ------  -------------
      1      C  20   G  1    S 1-8  O 1-10   boxx.      1
      1      C  20   G  2    S 1-8  O 1-10   boxx.      1
      1      C  20   G  3    S 1-8  O 1-10   boxx.      1

 Processor  Global  Global  Global  Sweep    File
  Number    Coarse  Group   Sweep   Omega   Output.proc
 ---------  ------  ------  ------  ------  -------------
      1      C  21   G  1    S 1-8  O 1-10   boxx.      1
      1      C  21   G  2    S 1-8  O 1-10   boxx.      1
      1      C  21   G  3    S 1-8  O 1-10   boxx.      1

 Processor  Global  Global  Global  Sweep    File
  Number    Coarse  Group   Sweep   Omega   Output.proc
 ---------  ------  ------  ------  ------  -------------
      1      C  22   G  1    S 1-8  O 1-10   boxx.      1
      1      C  22   G  2    S 1-8  O 1-10   boxx.      1
      1      C  22   G  3    S 1-8  O 1-10   boxx.      1

 Processor  Global  Global  Global  Sweep    File
  Number    Coarse  Group   Sweep   Omega   Output.proc
 ---------  ------  ------  ------  ------  -------------
      1      C  23   G  1    S 1-8  O 1-10   boxx.      1
      1      C  23   G  2    S 1-8  O 1-10   boxx.      1
      1      C  23   G  3    S 1-8  O 1-10   boxx.      1

 Processor  Global  Global  Global  Sweep    File
  Number    Coarse  Group   Sweep   Omega   Output.proc
 ---------  ------  ------  ------  ------  -------------
      1      C  24   G  1    S 1-8  O 1-10   boxx.      1
      1      C  24   G  2    S 1-8  O 1-10   boxx.      1
      1      C  24   G  3    S 1-8  O 1-10   boxx.      1

 Processor  Global  Global  Global  Sweep    File
  Number    Coarse  Group   Sweep   Omega   Output.proc
 ---------  ------  ------  ------  ------  -------------
      1      C  25   G  1    S 1-8  O 1-10   boxx.      1
      1      C  25   G  2    S 1-8  O 1-10   boxx.      1
      1      C  25   G  3    S 1-8  O 1-10   boxx.      1

 Processor  Global  Global  Global  Sweep    File
  Number    Coarse  Group   Sweep   Omega   Output.proc
 ---------  ------  ------  ------  ------  -------------
      1      C  26   G  1    S 1-8  O 1-10   boxx.      1
      1      C  26   G  2    S 1-8  O 1-10   boxx.      1
      1      C  26   G  3    S 1-8  O 1-10   boxx.      1

 Processor  Global  Global  Global  Sweep    File
  Number    Coarse  Group   Sweep   Omega   Output.proc
 ---------  ------  ------  ------  ------  -------------
      1      C  27   G  1    S 1-8  O 1-10   boxx.      1
      1      C  27   G  2    S 1-8  O 1-10   boxx.      1
      1      C  27   G  3    S 1-8  O 1-10   boxx.      1
     1  Dumping binary flux  moments  to file: boxx.f1
```

5. References

Alcouffe, R., E. Larsen, W. Miller, and B. Wienke, "Computational Efficiency of Numerical Methods for the Multigroup, Discrete Ordinates Neutron Transport Equations: The Slab Geometry Case," *Nuclear Science and Engineering,* 71:111-127 (1979).

Barbieri, K., "Introduction to the SP2 at CTC," *Proc Conference on Parallel Programming on the IBM SP-2,* Cornell Theory Center, Cornell University, Ithaca, N.Y. (1995).

Barbucci, P., and F. DiPasquantonio, " Exponential Supplementary Equations for Sn Methods: The One-Dimensional Case," *Nuclear Science and Engineering,* 63: 179-187, (1977).

Barnett, A., J. Morel and D. Harris, "A Multigrid Acceleration Method for the One-Dimensional SN Equations with Anisotropic Scattering," *Nuclear Science and Engineering,* 102: 1 (1989).

Bell, G. and S. Glasstone. *Nuclear Reactor Theory,* Malabar, Florida: Krieger, 1985.

Boltzmann, L., *Lectures on Gas Theory* (English translation from original German manuscript (1872)), Berkely: University of California Press, 1964.

Brandt, A., "Multi-level Adaptive Solutions to Singular Perturbation Problems, in P.W. Hernker (ed.), *Numerical Analysis of Singular Perturbation Problems,* Academic Press, New York, 1979, pp. 53-142.

Briesmeister, J., ed., "MCNP-A General Monte Carlo Code for Neutron and Photon Transport, Version 4b," *Los Alamos National Laboratory,* 1998.

Chandy, K., and J. Misra, *Parallel Program Design--A Foundation,* Reading, MA: Addison-Wesley, 1988.

Chilton, A., J. Shultis, and R. Faw, *Principles of Radiation Shielding,* Englewood Cliffs, New Jersey: Prentice Hall, 1984.

Dorr, M., and E. Salo, "Performance of a Neutron Transport Code with Full Phase Space Decomposition on the Cray Research T3D," *Proc American Nuclear Society Int'l Conference on Mathematics and Computations, Reactor Physics, and Environmental Analyses,* Portland, Oregon (1995).

Flynn, M., "Some Computer Organizations and their Effectiveness," *IEEE Trans Comput,* C-21:948-960 (1972).

Freeman, T. and C. Phillips, *Parallel Numerical Algorithms,* New York: Prentice Hall, 1992.

Gerner, J., "Scalability Issues," *Proc Conference on Parallel Programming on the IBM SP-2,* Cornell Theory Center, Cornell University, Ithaca, N.Y. (1995).

Gropp, W., E. Lusk, and A. Skjellum, *Using MPI: Portable Parallel Programming with the Message Passing Interface,* Cambridge, Massachusetts: MIT Press, 1994.

Haghighat, A., “New Spatial Parallel Sn Algorithms for One Dimensional Spherical Geometry," *Transactions of the American Nuclear Society,* 65: 206-207 (1992).

Haghighat, A., M. Hunter, and R. Mattis, "Iterative Schemes for Parallel Sn Algorithms," *Proc 1994 American Nuclear Society Reactor Physics Conference,* I: 423, Knoxville, Tennessee (1994).

Haghighat, A., M. Hunter, and R. Mattis, "Iterative Schemes for Parallel Sn Algorithms in a Shared Memory Computing Environment," *Nuclear Science and Engineering,* 121: 103-113, (1995).

Haghighat, A., G. Sjoden, and M. Hunter, “Parallel Algorithms for the Linear Boltzmann Equation Based on Complete Phase Space Decomposition," *(Invited) SIAM Annual Meeting,* Kansas City, MO,(1996).

Proc 1st International Workshop/Training Course on Transport Methodologies and Uncertainty Estimation for PWR Vessel Fluence and BWR Shield/Shroud Dose Calculations, A. Haghighat, ed., Penn State University, (1995).

Hill, T. R., "ONETRAN: A discrete Ordinates Finite Element Code for the Solution of the One-Dimensional Multigroup Transport Equation," *LA-5990-MS,* Los Alamos Scientific Laboratory (1975).

Hiromoto, R., and B. Wienke, "A Chaotic, Single-Pass Inner-Iteration Scheme for the Discrete Ordinate Method Sn,," *Proc 1989 American Nuclear Society Mathematics and Computation Conference,* 70: 1-17, Santa Fe, New Mexico (1989).

Hoermann, H., "Application of Parallel Computing Systems to Nuclear Engineering Problems," *Proc Int'l Topical Meeting on Advances in Mathematical Methods for the Solution of Nuclear Engineering Problems,* II:519-527, Munich, Federal Republic of Germany (1981).

Hunter, M., and A. Haghighat, "Combined Spatial/Angular Domain Decomposition Sn Algorithms for Shared Memory Parallel Machines," *Proc Joint Int'l Conference on Mathematical Methods and Supercomputing in Nuclear Applications,* 2:112-123, Karlsruhe, Germany (1993).

James, M., G. Smith, and J. Wolford, *Applied Numerical Methods for Digital Computation with FORTRAN and CSMP (2nd Edition),* New York, Harper and Row, 1977.

Kallinderis, Y., “Numerical Treatment of Grid Interfaces for Viscous Flows,” *J. Comp Phys* , 98: 129-144 (1992).

Kobayashi, K., "A Proposal for 3-D Radiation Transport Benchmarks for Simple Geometries with Void Region," *Dept of Nuclear Engineering, Kyoto University, Yoshida, Sakyoku, Kyoto, Japan, presented as a Technical Memorandum submitted to the OECD/NEA,* winter 1997.

Lathrop, K., "Spatial Differencing of the Transport Equation: Positivity vs. Accuracy," *J. Comp Phys,* 4: 475-498 (1969).

Lee, C., "The Discrete Sn Approximations to Transport Theory," *LA-2595,* Los Alamos Scientific Laboratory, (1962).

Lewis, E. E. and W. F. Miller. *Computational Methods of Neutron Transport,* LaGrange Park, Illinois: American Nuclear Society, 1993.

Mathews, K., G. Sjoden, and B. Minor "Exponential Characteristic Spatial Quadrature for Discrete Ordinates Radiation Transport in Slab Geometry," *Nuclear Science and Engineering,* 118: 24-37 (1994).

Mattis, R., and A. Haghighat, "Parallel Sn Algorithms on Workstation Networks," *Proc 8th Int'l Conference on Radiation Shielding,* I: 558-563 (1994).

Miller, W., "Generalized Rebalance: A Common Framework for Transport Acceleration Methods," *Nuclear Science and Engineering,* 65:226-236 (1978).

Nakamura, S., *Computational Methods in Engineering and Science,* New York, Wiley and Sons, 1977.

Nowak, P., E. Larsen, and W. Martin, "Multigrid Methods for Sn Problems," *Transactions of the American Nuclear Society,* 55: 355-356 (1987).

O'Dell, R., F. Brinkley, D. Marr, and R. Alcouffe, *"DANTSYS - One, Two, and Three Dimensional Multigroup Discrete Ordinates Transport Code System,"* Los Alamos National Laboratory, (1995).

Petrovic, B., and A. Haghighat, "Boundary Conditions Induced Oscillations in the Sn Solutions of the Neutron Transport Equation," *Proc Int'l Conf on Mathematics and Computations, Reactor Physics, and Environmental Analyses,* Portland, OR, April 30-May 4 1995, Vol I: 382-391, La Grange Park, IL, American Nuclear Society (1995a).

Petrovic, B., and A. Haghighat, "Analysis of Inherent Oscillations in Multidimensional SN Solutions of the Neutron Transport Equation," *Nuclear Science and Engineering,* 124: 31-62, (1996).

Petrovic, B., and A. Haghighat, "New Directional Theta-Weighted Sn Differencing Scheme," *Transactions of the American Nuclear Society,* 73: 195-197 (1995b).

Petrovic, B., and A. Haghighat, "New Directional Theta-Weighted Sn Differencing Scheme and its Application to Pressure Vessel Fluence Calculations," *Proc 1996 Radiation Protection and Shielding Topical Meeting,* Falmouth, MA, Vol I, 3-10 (1996).

Petrovic, B., and A. Haghighat, "New Directional Theta-Weighted (DTW) Differencing Scheme and Reduction of Estimated Pressure Vessel Fluence Uncertainty," *Proc 9th International Symposium on Reactor Dosimetry,* 746-753, Prague, Czech Republic, September 1996.

Petrovic, B., A. Haghighat, M. Mahgerefteh, and J. Louma, "Validation of Sn Transport Calculations for Pressure Vessel Fluence Determination at Penn State," *Proc 8th Int'l Conference on Radiation Shielding,* I: 558-563 (1994).

Phillips, R., and F. Schmidt, "Multigrid Techniques for the Numerical Solution of the Diffusion Equation," *Numerical Heat Transfer,* 7, 251-268 (1984).

Reed, "The Effectiveness of Acceleration Techniques for Iterative Methods in Transport Theory," *Nuclear Science and Engineering,* 45:245-254 (1971).

Rhoades, W., "Improvements in Discrete-Ordinates Acceleration," *Transactions of the American Nuclear Society,* 39: 753-755 (1981).

Rhoades, W., and R. Childs, *"TORT/DORT: Two- and Three-Dimensional Discrete Ordinates Transport," CCC-543,* ORNL Radiation Shielding Information Center, Oak Ridge, TN (1991).

Rhoades, W., and W. Engle, "A New Weighted Difference Formulation for Discrete Ordinates Calculations," *Transactions of the American Nuclear Society,* 27: 776-777 (1977).

Sewell, G. *The Numerical Solution of Ordinary and Partial Differential Equations,* New York, Academic Press, 1988.

Sjoden, G., "Exponential Characteristic Spatial quadrature for Discrete Ordinates Neutral Particle Transport in Slab Geometry," *M.S. Thesis, AFIT/GNE/ENP/92-M10,* Air Force Institute of Technology (1992).

Sjoden, G. and A. Haghighat, "The Exponential Directional Weighted (EDW) Differencing Scheme in 3-D Cartesian Geometry," *Proceedings of the Joint International Conference on Mathematics and Supercomputing for Nuclear Applications, Saratoga Springs,* New York, Vol II: 1267-1276, October 1997.

Sjoden, G., and A. Haghighat, "*PENTRAN,* A 3-D Scalable Transport Code with Complete Phase Space Decomposition," *Transactions of the American Nuclear Society,* 74, 181-183 (1996).

Sjoden, G. E., "*PENTRAN*: A Parallel 3-D SN Transport Code with Complete Phase Space Decomposition, Adaptive Differencing, and Iterative Solution Methods," *Ph.D. Thesis in Nuclear Engineering,* The Pennsylvania State University, May 1997.

Sjoden, G., and A. Haghighat, "Taylor Projection Mesh Coupling Between 3-D Discontinuous Grids for Sn," *Transactions of the American Nuclear Society,* 74, 178-179 (1996).

Sjoden, G., and A. Haghighat, "A New Adaptive Differencing Strategy in the *PENTRAN* 3-D Parallel Sn Code," *Transactions of the American Nuclear Society,* 75, (1996).

Sjoden, G., and A. Haghighat, "A Simplified Multigrid Acceleration in the *PENTRAN* 3-D Parallel Code ," *Transactions of the American Nuclear Society,* 75, (1996).

Stamm'ler, R. and M. Abbate, *Methods of Steady State Reactor Physics in Nuclear Design,* New York, Academic Press, 1983.

Wagner, J. and A. Haghighat, "Application of the Discrete Ordinates Adjoint Function to Accelerating Monte Carlo Reactor Cavity Dosimetry Calculations," *Proc 1996 Radiation Protection and Shielding Topical Meeting,* Falmouth, MA, Vol I, 345-352 (1996).

Walters, W., T. Wareing, and D. Marr, "The Non-Linear Characteristic Scheme for X-Y Geometry Transport Problems," *Proc Int'l Conf on Mathematics and Computations, Reactor Physics, and Environmental Analyses,* Vol I, American Nuclear Society, Portland, OR, April 30-May 4 1995.

Wareing, T. and R. Alcouffe, "An Exponential Discontinuous Scheme for x-y-z Geometry Transport Problems," *Proc 1996 Radiation Protection and Shielding Topical Meeting*, Vol II: 597-604, American Nuclear Society, Falmouth, MA, 1996.

Werner, W., "Application of Parallel Computing Systems to Nuclear Engineering Problems," *Proc Int'l Topical Meeting on Advances in Mathematical Methods for the Solution of Nuclear Engineering Problems,* II:509-519, Munich, Federal Republic of Germany (1981).

Wienke, B., R. Hiromoto, and R. Brickner, "Parallel Discrete Ordinates Algorithms on Distributed and Common Memory Systems," *Transactions of the American Nuclear Society,* 55: 321-322 (1987).

Wienke, B., and R. Hiromoto, "Parallel Sn Iteration Schemes," *Nuclear Science and Engineering,* 90: 116-123 (1985).

Yavuz, M. and C. Aykanat, "Spatial Domain Decomposition Applied to Linear Discontinuous Sn Methods," *Transactions of the American Nuclear Society,* 66: 274 (1992).

Yavuz, M. and E. Larsen, "Iterative Methods for Solving x-y Geometry Sn Problems on Parallel Architecture Computers," *Nuclear Science and Engineering,* 112: 32-42 (1992).

Yavuz, M. and E. Larsen, "Spatial Domain Decomposition Methods for Discrete Ordinates Problems," *Proc 1989 American Nuclear Society Mathematics and Computation Conference,* 69, Santa Fe, New Mexico (1989).