A NEW ALGORITHM FOR GENERATING HIGHLY ACCURATE BENCHMARK SOLUTIONS TO TRANSPORT TEST PROBLEMS*

Y. Y. Azmy

Oak Ridge National Laboratory
Engineering Physics and Mathematics Division
P.O. Box 2008, Bldg. 6025
Oak Ridge, Tennessee 37831-6363

Full paper for presentation to the *International Conference on Mathematics and Computations, Reactor Physics, and Environmental Analyses*, April 30–May 4, 1995, Portland, Oregon.

*Research sponsored by the U.S. Department of Energy under contract No. DE-AC05-84OR21400 with Martin Marietta Energy Systems, Inc.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

A NEW ALGORITHM FOR GENERATING HIGHLY ACCURATE BENCHMARK SOLUTIONS TO TRANSPORT TEST PROBLEMS

Y. Y. Azmy

Oak Ridge National Laboratory
P.O. Box 2008, MS 6363
Oak Ridge, Tennessee, 37831-6363
(615) 574-8069, yya@ornl.gov

ABSTRACT

We present a new algorithm for solving the neutron transport equation in its discrete-variable form. The new algorithm is based on computing the full matrix relating the *scalar* flux spatial moments in all cells to the fixed neutron source spatial moments, foregoing the need to compute the angular flux spatial moments, and thereby eliminating the need for sweeping the spatial mesh in each discrete-angular direction. The matrix equation is solved exactly in test cases, producing a solution vector that is free from iteration convergence error, and subject only to truncation and roundoff errors. Our algorithm is designed to provide method developers with a quick and simple solution scheme to test their new methods on difficult test problems without the need to develop sophisticated solution, e.g. acceleration, algorithms before establishing the worthiness of their innovation. We demonstrate the utility of the new algorithm by applying it to the Arbitrarily High Order Transport Nodal (AHOT-N) method, and using it to solve Burre's Test Problem, a suite of benchmark problems that covers a large region in parameter space. Our results provide highly accurate benchmark solutions, that can be distributed electronically and used to verify the pointwise accuracy of other solution methods and algorithms.

I. INTRODUCTION

As computational resources advance, applications grow more sophisticated and demand deeper understanding of solution methods, and the development of new methods to overcome deficiencies in existing ones. This inadvertently leads computational scientists to explore new ideas whose legitimacy must be established by comparing the accuracy of the resulting solution to benchmark solutions for one or more test problems. Thus it is important for the benefit of enriching a given field to facilitate the process of solving the equations corresponding to the new method, and verifying its accuracy, expressed in several relevant ways, against highly accurate solutions to well established, challenging test problems. In this paper we attempt to provide the above described tools for computational scientists in the neutron transport area.

Neutron transport problems are notorious for their heavy demand on computational resources, namely memory size and execution time. The reason is evident from the fact that in N-dimensional geometry the phase space for the steady state case has dimension $2N$, implying a large number of dependent variables to be computed and stored. The classical Source Iteration (SI) algorithm commonly used for solving neutron transport problems eliminates the need for storing the angular flux spatial moments, instead iterating on, and storing, the scalar flux spatial moments thus reducing the demand on memory size. This is accomplished by computing the scattering source for every iteration using the previous iterate of the scalar flux moments (initial guess for the first iteration) then conducting mesh sweeps along each of the discrete ordinates and accumulating the contribution to the scalar flux spatial moments in an array. The simplicity of the SI algorithm is counterbalanced by its slow convergence in difficult problems, namely thick, highly scattering media, precisely the class of problems that poses the greatest challenge to method accuracy. Hence most production codes implement one or more acceleration schemes for the SI algorithm. In contrast, experimental codes and methods seldom implement acceleration schemes before their accuracy is carefully tested, because most acceleration methods are based on complicated formalisms and require additional programming. The computational scientist's dilemma is now evident: on the one hand establishing a new method's high accuracy in solving difficult problems via the SI is inadequate because it could consume a large number of iterations to converge, possibly falsely, to a solution that is contaminated by convergence errors. On the other hand implementing sophisticated acceleration schemes without *a* priori guarantee of the method's accuracy could prove disappointing and wasteful. Thus we propose a new solution algorithm that is easy to implement, and does not require performing iterations as an aid to the method developer.

Another difficulty facing method developers and users alike is concerned with establishing and verifying a new method's accuracy. Usually this is based on solving one or two test problems and comparing a few results, mostly global features of the solution such as the fundamental eigenvalue, total power, region-averaged flux, *etc,* to well established, highly accurate benchmark solutions. While such measures of accuracy may be sufficient in some applications, they are very *coarse* in nature, and do not satisfy the user interested in pointwise accuracy of the solution. In the past few years Arbitrarily High Order Transport (AHOT) methods of the Nodal (AHOT-N)[1] and Characteristic (AHOT-C)[2] types have been reported. These methods provide highly accurate solutions to simple test problems, where the solution consists of arbitrarily high order (as specified by the user at run time) spatial moments of the solution that can be used to reconstruct the "pointwise" solution over the entire problem domain. Clearly the computational resources required by such methods to solve practical problems are prohibitive; nevertheless, they provide a powerful tool for determining highly accurate, pointwise, solutions to relatively small, but potentially difficult, test problems. These pointwise solutions, represented by the computed spatial moments of the scalar flux, can be used as benchmarks to evaluate the accuracy of newly developed methods.

In this paper we combine our new algorithm for solving *linear* neutron transport methods to the AHOT-N in particular, and we use the resulting approach to solve Burre's Test Problem[3] using up to quartic (fourth) order spatial expansions of the flux. Section II is dedicated to a brief review of the AHOT-N method, and its discrete-variable equations. In Sec. III we develop the new algorithm for solving linear transport methods by presenting two methods for constructing the full transport matrix that maps a fixed-neutron source distribution into the scalar flux spatial moments. This is followed in Sec. IV by a discussion of pointwise reconstruction of the solution over the problem domain and related problems. We bring all these ideas together in Sec. V where we describe the full suite of problems constituting Burre's Test Problem, and we use our new technique to study one of its cases.

II. THE AHOT-N METHOD

The AHOT-N method has been derived in detail elsewhere.[1] For the sake of completeness, and to introduce the notation and discrete-variable equations used in Sec. III we outline the method here. We note that while the original AHOT-N was derived for general dimensionality,[1] we restrict the following equations to the two-dimensional case. Extension to general dimensionality is straightforward but lengthy.

The set of discrete-variable unknowns solved for in the AHOT-N method are the Legendre spatial moments of the angular flux within, and on the edges of a computational cell, as defined by,

$$\psi_{k,i,j,m,n} \equiv \int_{-a_m}^{+a_m} \frac{dx}{2a_m} P_i(x) \int_{-b_n}^{+b_n} \frac{dy}{2b_n} P_j(y) \psi_k(x,y), \tag{1}$$

$$\psi_{k,x,j,m,n}(\pm a_m) \equiv \int_{-b_n}^{+b_n} \frac{dy}{2b_n} P_j(y) \psi_k(\pm a_m, y) \tag{2}$$

and an analogous definition for $\psi_{k,i,y,m,n}(\pm b_n)$, respectively. Here we use k, m, n to denote the angular, x-, and y-dimension indices, respectively; $P_i(x)$ is the i^{th} order Legendre polynomial, normalized over the interval $[-a_m, +a_m]$; $2a_m$ and $2b_n$ are the x-, and y-dimensions of cell (m,n), respectively; and $\psi_k(x,y)$ is the continuum angular flux in discrete direction k.

In two-dimensional geometry there are three set of discrete-variable equations constituting the Λ^{th}-order AHOT-N method. The first is a set of $(\Lambda+1)^2$ equations expressing the conservation of the angular flux spatial moments over a cell, and are obtained by taking the $(i,j)^{th}$-moment of the discrete ordinates equations, in the sense of Eq. (1), $i,j=0, \cdots, \Lambda$. These can be written in terms of the cell-moments of the flux, Eq. (1), and the transverse-moments evaluated on the incoming and outgoing edges of the cell, by taking into account the sign of the x- and y-components of the angular direction cosines, μ_k and η_k, respectively, in Eq. (2),

$$sg(\mu_k)^i\ \varepsilon^x_{k,m}\left\{\psi^o_{k,x,j,m,n}-(-1)^i\psi^i_{k,x,j,m,n}\right\}+sg(\eta_k)^j\ \varepsilon^y_{k,n}\left\{\psi^o_{k,i,y,m,n}-(-1)^j\psi^i_{k,i,y,m,n}\right\}$$
$$-2sg(\mu_k)\,\varepsilon^x_{k,m}\sum_{l=o(i)}^{i-1}{}^{*}\,(2l+1)\,\psi_{k,l,j,m,n}-2sg(\eta_k)\,\varepsilon^y_{k,n}\sum_{l=o(j)}^{j-1}{}^{*}\,(2l+1)\,\psi_{k,i,l,m,n}$$
$$+\sigma^T_{m,n}\,\psi_{k,i,j,m,n}=\sigma^S_{m,n}\,\phi_{i,j,m,n}+S_{i,j,m,n}, \tag{3}$$

where the o and i superscripts on the flux transverse-moments denote the outgoing and incoming edges, respectively; sg is the signum function; $i+1=o(i)(mod 2)$, and Σ^* is the summation operator with increment 2; $\sigma^T_{m,n}$ and $\sigma^S_{m,n}$ are the total and isotropic scattering cross sections in cell (m,n), respectively; $\phi_{i,j,m,n}$ and $S_{i,j,m,n}$ are the cell moments of the scalar flux and fixed neutron source, respectively; and

$$\varepsilon^x_{k,m}\equiv|\mu_k|\ /\ (2\,a_m),$$

with an analogous definition for $\varepsilon^y_{k,n}$. We note here that the summation terms are different from the formula appearing in Ref. (1), but it is trivial to show that the two forms are equivalent.

The second set of equations is a weighted difference formula relating the the cell-, and x-edge-moments of the angular flux,

$$\left[\frac{1+\alpha_{k,m,n}}{2}\right]\psi^o_{k,x,j,m,n}+\left[\frac{1-\alpha_{k,m,n}}{2}\right]\psi^i_{k,x,j,m,n}$$
$$=\sum_{l=0,even}^{\Lambda}{}^{*}\,(2l+1)\,\psi_{k,l,j,m,n}+sg(\mu_k)\,\alpha_{k,m,n}\sum_{l=1,odd}^{\Lambda}{}^{*}\,(2l+1)\,\psi_{k,l,j,m,n}, \tag{4}$$

where the spatial weights, $\alpha_{k,m,n}$, are expressed exclusively in terms of $\varepsilon^x_{k,m}/\sigma^T_{m,n}$ as shown in Ref. 1. The third set of equations analogously relates the cell-, and y-edge-moments of the flux. The weighted difference relations, e.g. Eq. (4), are obtained by taking the transverse-moments of the continuum transport equation separately in each dimension, then integrating the resulting ordinary differential equation exactly after locally expanding the scattering source in a Λ^{th}-order Legendre series.

In the SI algorithm the scalar flux, and hence the scattering source, moments are considered known in each inner iteration in terms of the previous iterate. By sweeping the mesh starting from edges on which the incoming-edge flux moments are also known, Eqs. (1), (4), and its y-analogue are solved for the cell-, and outgoing-edge flux moments. This process can be formally represented by the matrix equation,

$$[\psi_{k,m,n}\ ,\ \psi^o_{k,x,m,n}\ ,\ \psi^o_{k,y,m,n}]^T=\Gamma_{k,m,n}\ [\sigma^S_{m,n}\phi^p_{m,n}+S_{m,n}\ ,\ \psi^i_{k,x,m,n}\ ,\ \psi^i_{k,y,m,n}]^T, \tag{5}$$

where the subvectors on both sides are composed from all computed moments, hence we suppress the moment order indices i and j, and Γ is a block matrix whose elements are computed from the balance and weighted difference equations. In Eq. (5) we added a p superscript to $\phi_{m,n}$ to denote that it is carried over from the previous iteration. The mesh sweep proceeds by accumulating the just-computed cell-moments in the new iterate array of the scalar flux spatial moments, denoted by a ν superscript, via,

$$\phi^\nu_{m,n}=\sum_{k=1}^{K}\omega_k\ \psi_{k,m,n}, \tag{6}$$

then setting the just-computed outgoing-edge flux moments to the incoming-edge flux moments in the down-flow-adjacent cells.

The SI algorithm has been implemented for the AHOT-N method in a test code, and used to compute highly accurate solutions to simple test problems.[1,2] However, these test problems were confined to thin, absorbing media in order to economically obtain a well-converged solution, thus limiting the accuracy evaluation to the less-difficult range in parameter space. Thick diffusive problems present the real challenge to method accuracy, and thus must be included in determining a new method's accuracy.

III. THE NEW SOLUTION ALGORITHM

The main idea in our new solution algorithm is to view the SI algorithm, i.e. Eqs. (5) and (6), as an iterative map of the scalar flux,

$$\phi^{\nu} = \mathrm{A}\,(\sigma^{S}\,\phi^{p} + S), \tag{7}$$

where we have suppressed the m,n cell indices to indicate the spatially global nature of this equation, and σ^S is a diagonal matrix whose elements correspond to the cell scattering cross section. It is possible to represent the SI scheme in this form because the discrete-variable equations and the solution algorithm are linear, and we assume vacuum boundary conditions. [If the incoming flux on the external boundaries is anisotropic Eq. (7) must be modified to take into account this additional "fixed source" of neutrons.] The SI converges to the solution ϕ^{∞} which satisfies,

$$\phi^{\infty} = (I - \sigma^{S}\,\mathrm{A})^{-1}\ \mathrm{A}S, \tag{8}$$

where I is the unit matrix. This matrix equation is of large dimension, $M \times N \times (\Lambda+1)^2$, and in most applications it is impractical to construct, let alone solve. Nevertheless, for simple test problems often used to evaluate the accuracy of a new method, or verify the correctness of a new code, the size of this matrix equation may be reasonably small thus providing an easy and economical alternative to the SI algorithm.

It is evident from the above discussion that our new algorithm is comprised of two steps: construct the A matrix and manipulate it in the form of Eq. (8), then solve the matrix equation. We will not dwell on the method of solving Eq. (8) since this can be accomplished easily using a standard mathematical library routine; in this case we employ LINPACK's SSIFA family of routines for real symmetric matrices.[4] Constructing the A matrix can be accomplished with one of two methods as described below.

A. Automatic Differentiation of a Mesh Sweep

The first approach recognizes that the A matrix is essentially the iteration Jacobian (divided by the scattering cross section matrix) of Eq. (7),

$$\mathrm{A}_{(i,j,m,n),(i',j',m',n')} = \frac{1}{\sigma^{S}_{m',n'}}\ \frac{\partial \phi^{\nu}_{i,j,m,n}}{\partial \phi^{p}_{i',j',m',n'}}\ . \tag{9}$$

Hence using an *Automatic Differentiation (AD) system*, such as GRESS,[5] we preprocess the AHOT code[2] declaring the old scalar flux spatial moments array as parameters, thus computing derivatives of all quantities with respect to its elements. Since the iterations are linear, the iteration Jacobian is constant across iterations so that only a single iteration need be performed. At the conclusion of this single iteration, i.e. mesh sweep in all angular directions, the derivative of the new scalar flux spatial moments array with respect to all declared parameters is retrieved and printed out. The so-obtained Jacobian matrix is then read into a small program that constructs the matrix equation, Eq. (8), and uses the SSIFA family of routines,[4] to solve for the solution vector.

The primary advantage of the AD approach is its simplicity as it requires minimal user knowledge of the particular transport method involved (as long as it is linear), and little additional programming. Given a transport code and a test problem, the user first generates a GRESS-ready version of the code,[5] essentially selecting options, specifying workspace, declaring parameters (old flux array), and retrieving and printing the derivatives. Second, the resulting code is preprocessed with GRESS to produce the *enhanced* source code, which is then compiled and linked with GRESS' library routines.[5] Third, a single iteration is performed on the test problem to produce the iteration Jacobian, and finally a small program constructs and solves the matrix equation, Eq. (8).

On the other hand the main drawback of the AD approach is the strict limit it imposes on the size of the test problem. This feature comprises one of the central research problems in AD techniques. However, since we propose our new algorithm as a means of producing highly accurate solutions for test problems only, in particular by employing coarse meshes and high approximation orders, this limitation is not too debilitating. Furthermore, if a test problem is too large to be differentiated in a single mesh sweep, i.e. memory size is not sufficient to declare all old scalar flux spatial moments as parameters, the Jacobian matrix can be constructed as follows. The derivative of all new scalar flux spatial moments with respect to all spatial moments of a particular cell old scalar flux comprises a $(\Lambda+1)^2$-block of rows in the

Jacobian matrix whose elements are independent of the derivatives with respect to other cells' old scalar flux moments. Hence the mesh can be divided into subdomains and the enhanced code executed once declaring the old flux spatial moments in each subdomain as parameters, resulting in stripes of rows of the Jacobian matrix, that are then stacked together to form the full iteration Jacobian matrix.

B. Computation of A from Γ in a Mesh Sweep

It is apparent that the reduced level of user involvement in the AD approach is costly in terms of its limitation of problem size, which results, at least in part, from the general purpose nature of GRESS, or for that matter any AD package. Alternatively, with more programming effort the Jacobian matrix can be accumulated explicitly via a single mesh sweep. In addition to serving the purpose of the work presented here as stated in the Introduction, this approach might be applicable to the development of new parallel algorithms of the neutron transport equation for massively parallel machines.[6]

Consider the formal relation, Eq. (5), for a given direction $1 \le k \le K$. [Note that any linear method based on the integro-differential representation of the transport equation can be written in this form.] The 3×3-block matrix Γ has $[(\Lambda+1)^2, (\Lambda+1), (\Lambda+1)]$ rows and columns comprising the submatrices $\gamma^{i,j}_{k,m,n}$ in the standard matrix notation. Let cell m,n be indexed so that for the present angular direction k, m=1=n denotes the cell at the mesh corner at which all incoming angular flux spatial moments are known from the boundary conditions; e.g. for $\mu_k, \eta_k > 0$ this would be the lower left corner cell. Since the mesh sweep in each angular direction proceeds completely independently from all other directions, we suppress reference to the k angular index, except where necessary, but the reader is reminded that all matrix elements depend on k explicitly (via the direction cosines), and implicitly (via the m,n cell indices). For arbitrary cell m,n suppose the matrices,

$$X_{(m,n),(m',n')} = \frac{\partial \psi^i_{x,m,n}}{\partial \phi^p_{m',n'}} \ , \ 1 \le m' \le m \ , \ 1 \le n' < n \ , \tag{10.a}$$

$$Y_{(m,n),(m',n')} = \frac{\partial \psi^i_{y,m,n}}{\partial \phi^p_{m',n'}} \ , \ 1 \le m' < m \ , \ 1 \le n' \le n \ , \tag{10.b}$$

are known. We derive a recursive relation for these two matrices that are updated throughout a mesh sweep, and used to compute the iteration Jacobian.

For the special case of vacuum boundary conditions considered here, and according to the cell-indexing scheme described above $\psi^i_{x,1,1}$and $\psi^i_{y,1,1} = 0$. It follows from Eq. (5) that,

$$X_{(1,2),(1,1)} = \sigma^S_{1,1}\ \gamma^{2,1}_{1,1}, \qquad Y_{(2,1),(1,1)} = \sigma^S_{1,1}\ \gamma^{3,1}_{1,1}, \tag{11}$$

where we have used the continuity condition across cell edges of the angular flux moments. Equations (11) serve as initialization formulas for the recursive relations derived below.

To compute the iteration Jacobian, or the A, matrix we accumulate the derivative of the angular flux throughout a mesh sweep, in exact analogy to the AD approach,

$$A_{(m,n),(m',n')} = \frac{1}{\sigma^S_{m',n'}} \sum_{k=1}^{K} \omega_k \frac{\partial \psi_{k,m,n}}{\partial \phi^p_{m',n'}} \ . \tag{12}$$

Equation (12) provides the (Λ+1)×(Λ+1) block located at $(m,n),(m',n')$ of the A matrix. Due to the hyperbolic nature of the transport operator the angular flux in a given cell depends only on the scattering source, hence the old scalar flux, in up-stream cells for each discrete direction,

$$\frac{\partial \psi_{m,n}}{\partial \phi^p_{m',n'}} = 0, \ m' > m, or \ n' > n. \tag{13}$$

Otherwise, the dependence of $\psi_{m,n}$ on $\phi^p_{m',n'}$ follows from its dependence on the incoming-edge flux moments as discussed shortly.

The contribution to the Jacobian diagonal block is given by,

$$\frac{\partial\psi_{m,n}}{\partial^{p}_{m,n}} = \sigma^{S}_{m,n}\ \gamma^{1,1}_{m,n} \tag{14}$$

The off-diagonal blocks are computed by differentiating Eq. (5) separately with respect to $\psi^{i}_{x,m,n}$ then $\psi^{i}_{y,m,n}$, and using the angular flux spatial moments continuity across cell edges,

$$\frac{\partial\psi_{m,n}}{\partial^{p}_{m',n'}} = \gamma^{1,2}_{m,n}\ X_{(m,n),(m',n')}\ +\ \gamma^{1,3}_{m,n}\ Y_{(m,n),(m',n')}\ \ ,\ 1 \le m' \le m\ ,\ 1 \le n' \le n\ . \tag{15}$$

Once this is computed, it is accumulated in the Jacobian matrix according to Eq. (12) for all valid m',n'.

Before proceeding to the next cell in the mesh sweep we must update the X and Y matrices, via the continuity of the angular flux across cell edges, e.g.,

$$X_{(m,n+1),(m',n')} = \frac{\partial\psi^{i}_{m,n+1}}{\partial\phi_{m',n'}} = \frac{\partial\psi^{o}_{m,n}}{\partial\phi_{m',n'}}\ ,\ 1 \le m' \le m\ ,\ 1 \le n' \le n\ . \tag{16}$$

The update occurs in two ways. First, in each step of the mesh sweep a new block will be appended to each of these two matrices representing the present computational cell. Elements of these blocks are obtained by differentiating Eq. (5), e.g.,

$$X_{(m,n+1),(m,n)} = \sigma^{S}_{m,n}\ \gamma^{2,1}_{m,n}\ . \tag{17}$$

Second, for all other m',n' the update is accomplished by compounding the present X and Y matrices with the derivative of the outgoing-edge flux moments with respect to the incoming-edge flux moments as computed from Eq. (5), e.g.,

$$X_{(m,n+1),(m',n')} = \gamma^{2,2}_{m,n}\ X_{(m,n),(m',n')} + \gamma^{3,2}_{m,n}\ Y_{(m,n),(m',n')}\ ,\ \ 1 \le m' \le m\ ,\ 1 \le n' \le n\ . \tag{18}$$

Equation (18) effectively closes the recursive relation for the X and Y matrices.

It is common in transport calculations to sweep the mesh in the x-direction first, that is along rows, one row at a time. Therefore only one copy of the Y matrix needs to be stored and updated throughout the sweep because the outgoing, *x=const.*, edge at which Y is updated coincides with the incoming edge where it is used in precisely the next cell in the sweep. In contrast, the X matrix is computed on a surface but not used until the entire row plus m cells in the next row are swept, that is to say after M cells in the sweep procedure. Hence it is necessary to store M different X arrays, one for each $m=1,\cdots,M$, each of which is overwritten then used in the next, vertically adjacent computational cell. This highly taxes the memory, but in the case of small test problems, the focus of this work, is not unmanageable. One final remark concerns the γ-block submatrices; in general these are not easily determined analytically for the AHOT-N, and are computed numerically using LINPACK's SSIFA family of routines.[4] In other methods, e.g. AHOT-C, the discrete-variable equations, Eqs. (3) and (4), may be uncoupled enabling an analytic evaluation of Γ.

IV. POINTWISE SOLUTION RECONSTRUCTION

The solution of the AHOT-N method equations is comprised of three vectors representing the cell-moments of the scalar flux, $M\times N\times(\Lambda+1)^2$, the x-moments of the flux evaluated on *y=const.* edges, $M\times(N+1)\times(\Lambda+1)$, and the y-moments of the flux evaluated on *x=const.* edges, $(M+1)\times N\times(\Lambda+1)$. Once a solution is obtained, the pointwise flux can be reconstructed over the entire problem domain from the cell-moments on a pointwise basis,

$$\phi(x,y) = \sum_{i,j=0}^{\Lambda} \phi_{i,j,m,n}\ P_i(x)\,P_j(y)\ \ ,\ (x,y)\text{ in cell } m,n\ . \tag{19}$$

Similarly, the flux along x or *y=const.* surfaces can be reconstructed from the corresponding transverse moments, e.g.,

$$\phi(\pm a_m,y) = \sum_{j=0}^{\Lambda} \phi^{\pm}_{x,j,m,n}\ P_j(y)\ \ ,\ y\text{ in cell } n\ , \tag{20}$$

where $\phi^{\pm}_{x,j,m,n}$ are accumulated from $\psi^{i}_{x,j,m,n}$ and $\psi^{o}_{x,j,m,n}$ using the angular weights as usual.

Digital computations are "discrete" in nature, and the concept of pointwise solutions must be adjusted to conform with this property inherent in this work. First we note that a user interested in the solution at a specific set of points can reconstruct the solution at these points using Eqs. (19) and (20). More commonly though the reconstructed solution is desired over a range of points, for example a density plot of the flux over the problem domain, constituting a continuum of points. Hence we introduce a *pixel mesh* which is usually finer than the computational mesh (except for zero-order methods where the two are identical) such that the latter is a subset of the former. The pixel mesh is defined by the user to resolve as fine a scale as deemed adequate for a particular application, and the "pointwise" solution then is computed over the finite, discrete set of points constituting the pixel map. Since the solution reconstruction is performed *a* posteriori it can be performed using different pixel meshes using the same AHOT-N solution, for example if the first pixel mesh turns out to be inadequately coarse.

Notice that the continuity of the angular, hence scalar, flux spatial moments across cell edges guarantees the uniqueness of the reconstructed edge flux, Eq. (20), whether we use the expansion coefficients, $\phi^{\pm}_{x,j,m,n}$, for the cell to the right or left of that edge. However, since the cell-, x-, and y-moment expansion coefficients are obtained, through the AHOT-N formalism, via three independent expansions of the flux, they do not necessarily produce a unique value at common points. That is to say, the pointwise value of the flux along, for example, an x*=const.* surface obtained from the y-moments of the solution, Eq. (20), may not coincide with that obtained by taking the limit of the cell-moments-constructed flux, Eq. (19), to the same surface. This discrepancy should diminish with increasing method order, and our numerical results seem to confirm this conjecture. However, it would be very desirable in the future to develop a blending formula to reconstruct the pointwise flux from all three sets of expansion coefficients simultaneously in such a way that the pointwise reconstructed flux is unique everywhere, and exactly attains the computed edge values on cell boundaries.

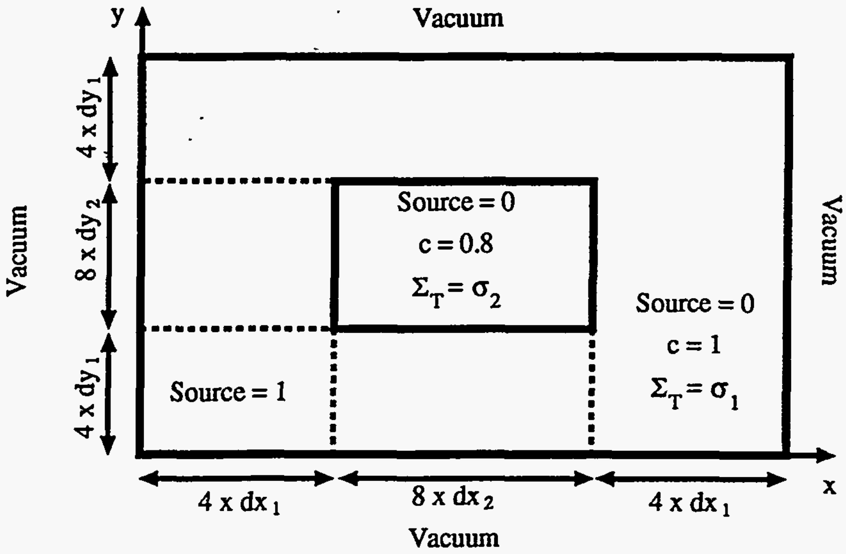
Our new solution algorithm deals primarily with the scalar flux, and does not involve the angular flux explicitly. Nevertheless, for tests that require knowledge of the angular flux, or related quantities, such as leakage, the cell- and transverse-moments of the angular flux can be computed directly from the solution. This is accomplished by sweeping the mesh once in the discrete directions of interest with Eq. (5) using the scalar flux spatial moments obtained as described above for $\phi^{p}_{m,n}$. Since the solution computed with our new algorithm is mathematically equivalent to the SI converged solution, the angular flux spatial moments also are equal, within arithmetic and convergence precision. Clearly once the angular flux moments are available, the pointwise angular flux can be reconstructed over the entire problem domain, or on selected surfaces, using formulas analogous to Eqs. (19) and (20).

Another utility of the work presented in this paper is performing numerical error analysis on multidimensional, complicated test problems. As observed earlier,[7] the generally discontinuous first derivative of the angular flux has impeded the performance of analytical error analysis in multidimensional geometry, even for the simplest methods. The next best option after an analytical error analysis is a numerical error analysis with respect to high accuracy solutions to a set of challenging test problems. Such highly accurate solutions can be obtained by reconstructing the solution over a fine pixel mesh using three different method orders, e.g. Λ, Λ+1, and Λ+2, then extrapolating the solution at each pixel to *infinite* order. This produces a reference solution that is more accurate than each of the three solutions, and against which solutions to new methods can be compared in order to verify or estimate their accuracy on a "pointwise" level.

V. APPLICATION TO BURRE'S TEST PROBLEM

We demonstrate the ideas presented above by applying them to the solution of two of the cases embodied in Burre's Test Problem,[3] a suite of difficult test problems originally proposed as a challenge to acceleration methods. The problem configuration is depicted in Fig. 1. In its original form, the cross sections for the two materials, σ_1^T and σ_2^T are varied independently over the set of values .01, .1, 1, and 10. Also the height of the cells denoted by dy_1 and dy_2 are varied independently over the set of values .01, .1, and 1, thus producing a total of 144 test problems that cover a wide range in parameter space. The original mesh proposed for this suite of problems is 16×16;[3] this mesh is inadequate for the purposes of the present study, though, due to the high order methods adopted here. Hence, in the examples presented in this section we use a 4×4 mesh for all high order methods, i.e. $\Lambda > 0$, and in order to retain spatial resolution for the zero

Fig. 1. Configuration of Burre's Test Problem



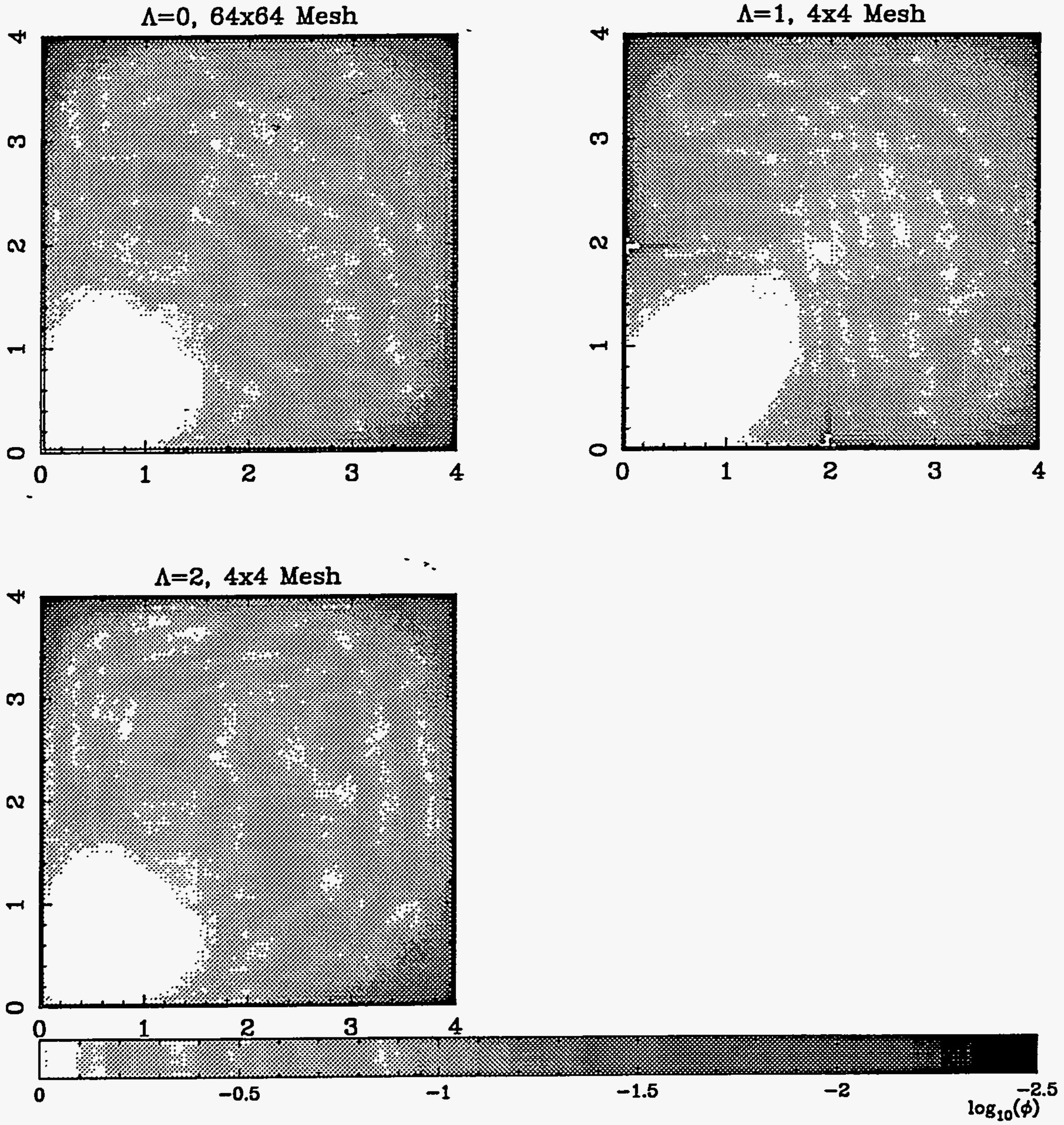
order method we use a 64×64 mesh for this case.

The usefulness of Burre's Test Problem in its entirety to verify new method accuracy involves generating and processing a large volume of data. Even then, communicating the results is not an easy task, perhaps demanding the development of an interactive postprocessor that reads in the reference solutions to the entire suite, then allows the user to manipulate them to view, and compare various features of the solution across methods. However, since such tool is not available yet, we focus on two cases of interest.

The first case is characterized by $dy_1 = 1 = dy_2$, $\sigma_1 = 10$, and $\sigma_2 = .01$. This problem involves an optically thick region surrounding an *almost* void, so that the outer region is approaching the thick diffusion limit, and the neutrons stream through very sharp discontinuities. We solve this problem using $\Lambda = 0$, 1,and 2, and then reconstruct the scalar flux over the entire problem domain on a 64×64 pixel mesh for the zero order method, and an 80×80 pixel mesh for all higher order methods. Density plots of the resulting scalar flux are presented in Fig. 2. The solution for this problem displays a smooth two and a half orders decrease in the flux from the source region to the right and top boundaries, with a *squarish* mostly leveled area over Region 2 due to its small optical thickness. The linear method, $\Lambda = 1$, flux is significantly different from the fine-mesh, zero order method solution, even in the source region where one would expect the solution to be most accurate. The *white pixels* at $x = 2 = y$ indicate negative scalar flux values as they occur outside the source region and cannot have larger values than points closer to the source region. In contrast, the quadratic method, $\Lambda = 2$, flux displays all the salient features of the fine-mesh solution, near and far from the source region, and does not produce negative flux values.

The second case is characterized by $dy_1 = 1 = dy_2$, $\sigma_1 = 1$, and $\sigma_2 = 10$. This case is difficult because it involves the streaming of neutrons in the relatively thin Region 1, turning the lower-right and upper-left corners around the very thick, slightly absorbing, Region 2. This causes the upper-right corner to be better illuminated than the upper-right corner of Region 2, even though the latter is closer to the source region. This feature is clearly displayed in Fig. 3 which depicts the density plots of the scalar flux for this case using the same spatial and pixel meshes as before, with orders $\Lambda = 0$, 2, 6, and 7. Here also white pixels outside the source region imply negative scalar flux values. Evidently there are more negative scalar fluxes computed in this problem due to the sharp gradients in the scalar flux surface, over four orders of magnitude. However, it is clear that the the number of white pixels decreases significantly as the method order increases. Also, in spite of the negative scalar flux values, the solution possesses the proper general features of the fine-mesh solution, namely the sharp decrease in magnitude in the upper-right corner of Region 2, and the illumination of Region 1 behind Region 2, with some shadowing effect of the former by the latter. Perhaps the most interesting feature

Fig. 2. Density Plots of the Reconstructed Scalar Flux for Burre's Test Problem with $dy_1 = 1 = dy_2$, $\sigma_1 = 10$, and $\sigma_2 = .01$.



Λ=0, 64x64 Mesh

Λ=1, 4x4 Mesh

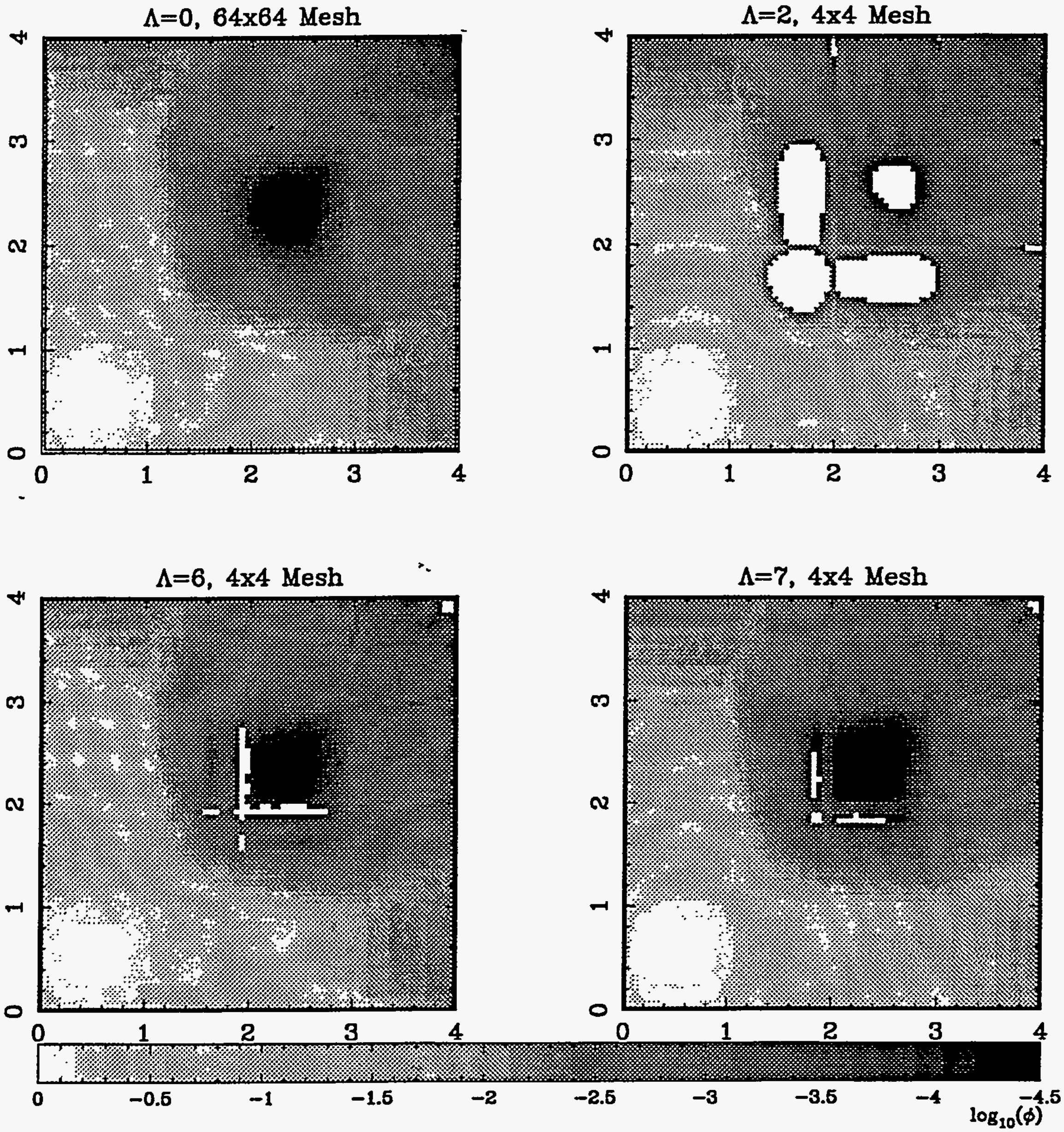
Λ=2, 4x4 Mesh

of this case is the resolution of the shadow of Region 2 on the part of Region 1 behind it because it demonstrates that discontinuous solutions can be represented by a finite series of continuous functions, Lĕgendre polynomials in this case, thus justifying the AHOT approach.

REFERENCES

1. Y. Y. Azmy, *Nucl. Sci. Eng.* **98,** 29 (1988).
2. Yousry Y. Azmy, *Ann. Nucl. Energy* **19,** 593 (1992).
3. C. A. Burre, Knolls Atomic Power Laboratory, personal communication (1994).

Fig. 3. Density Plots of the Reconstructed Scalar Flux for Burre's Test Problem with $dy_1 = 1 = dy_2$, $\sigma_1 = 1$, and $\sigma_2 = 10$.



Λ=0, 64x64 Mesh

Λ=2, 4x4 Mesh

Λ=6, 4x4 Mesh

Λ=7, 4x4 Mesh

4. J. J. Dongarra, J. R. Bunch, C. B. Moler, and G. W. Stewart, *LINPACK Users' Guide*, SIAM, Philadelphia, PA (1979).

5. J. E. Horwedel, "GRESS: A Preprocessor for Sensitivity Studies on Fortran Programs," in *Automatic Differentiation of Algorithms: Theory, Implementation, and Application,* A. Griewank and G. F. Corliss, Eds., p. 243, SIAM, Philadelphia, PA (1991).

6. Y. Y. Azmy, research in progress (1994).

7. Edward W. Larsen and Warren F. Miller, Jr., *Nucl. Sci. Eng.* 73, 76 (1980).