LA-UR-05-4684

Approved for public release;
distribution is unlimited.

Title: A Diffusion Synthetic Acceleration Method for Block Adaptive Mesh Refinement

Author(s): Robert C. Ward
Randal S. Baker
Jim E. Morel

Intended for: Mathematics and Computation, Supercomputing, Reactor Physics and Nuclear and Biological Applications Conference, Palais des Papes, Avignon, France, September 12-15, 2005

Mathematics and Computation, Supercomputing, Reactor Physics and Nuclear and Biological Applications
Palais des Papes, Avignon, France, September 12-15, 2005, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2005)

A DIFFUSION SYNTHETIC ACCELERATION METHOD FOR BLOCK ADAPTIVE MESH REFINEMENT

Robert C. Ward and Randal S. Baker
Los Alamos National Laboratory
MS D409
P.O. Box 1663
Los Alamos, NM, 87545
wardr@lanl.gov; rsb@lanl.gov

Jim E. Morel
Los Alamos National Laboratory
MS D413
P.O. Box 1663
Los Alamos, NM, 87545
jim@lanl.gov

ABSTRACT

A prototype two-dimensional Diffusion Synthetic Acceleration (DSA) method on a Block-based Adaptive Mesh Refinement (BAMR) transport mesh has been developed. The Block-Adaptive Mesh Refinement Diffusion Synthetic Acceleration (BAMR-DSA) method was tested in the PARallel TIme-Dependent SN (PARTISN) deterministic transport code. The BAMR-DSA equations are derived by differencing the DSA equation using a vertex-centered diffusion discretization that is diamond-like and may be characterized as “partially” consistent. The derivation of a diffusion discretization that is fully consistent with diamond transport differencing on BAMR mesh does not appear to be possible. However, despite being partially consistent, the BAMR-DSA method is effective for many applications. The BAMR-DSA solver was implemented and tested in two dimensions for rectangular (XY) and cylindrical (RZ) geometries. Testing results confirm that a partially consistent BAMR-DSA method will introduce instabilities for extreme cases, e.g., scattering ratios approaching 1.0 with optically thick cells, but for most realistic problems the BAMR-DSA method provides effective acceleration. The initial use of a full matrix to store and LU-Decomposition to solve the BAMR-DSA equations has been extended to include Compressed Sparse Row (CSR) storage and a Conjugate Gradient (CG) solver. The CSR and CG methods provide significantly more efficient and faster storage and solution methods.

KEYWORDS: Block AMR, DSA, Conjugate Gradient, Compressed Sparse Row

1. INTRODUCTION

Orthogonal single-level mesh are often inefficient in multiple-material and multidimensional problems because of the necessity of having cells that are small enough to contend with optically thick regions on the mesh. As a result, a single-level mesh may produce cells that are too fine in

regions that do not need it, wasting valuable computer resources. The use of Adaptive Mesh Refinement (AMR) permits more efficient and more realistic mesh in the modeling of multidimensional transport simulations[1]. A Block-AMR method was implemented in the PARallel TIme Dependent SN (PARTISN) deterministic transport code but lacked any iterative acceleration other than Transport Synthetic Acceleration (TSA). TSA by itself is not considered to be sufficeintly efficient for many of the problems run with PARTISN. The Diffusion Synthetic Acceleration (DSA) method is the preferred transport acceleration method for most problems run with PARTISN. Without DSA, the use of a Block-AMR mesh was somewhat limited because of the relatively slow convergence rate of source iteration for many routine applications.

The original prototype Block-AMR DSA (BAMR-DSA) method[2] implemented in PARTISN utilized a dense or full matrix storage system and was solved with the LU-Decomposition (LUD)[3] method. The dense storage format is inefficent because the BAMR-DSA system matrix is filled with a large percentage of zeros. The LUD method performs an exact matrix inversion and was choosen in order to verify that any problems with the solution of the BAMR-DSA method was a result of the method and not the matrix inversion. Because LUD is slow, $O(N^3)$ operation, the dense matrix storage format is inefficient, and the solution of the BAMR-DSA system on large parallel computers is required, the Compressed Sparse Row (CSR) matrix storage format and the Conjugate Gradient (CG)[4] method has been implemented and tested for the BAMR-DSA system's solver.

The BAMR-DSA method and its solution using the sparse CG linear systems solver is discussed. A background on Block-AMR is presented followed by a brief discussion of the BAMR-DSA method. The CSR and CG methods are then discussed with respect to the BAMR-DSA system. Finally, a discussion of the tests conducted and results is presented.

2. BLOCK-ADAPTIVE MESH REFINEMENT

In a multidimensional AMR mesh, the cells are not simply connected as they would be in a standard single-level mesh[1]. An AMR mesh is composed of orthogonal cells that are locally refined by some sort of error estimation so that the cell sizes along a given direction are no longer constrained by the need for a small cell in one section of the mesh. Thus, an AMR mesh may map more efficiently to the actual problem geometry than a simply connected grid[2].

The type of AMR method implemented in PARTISN is block-based[1]. In a block-based AMR mesh, the entire geometry is divided into blocks. Each block is assigned a level number that indicates how many cells it has. In two dimensions, a level zero block contains only one cell (1×1), a level one block has four cells (2×2), a level two block has sixteen cells (4×4), etc. The external source and cross sections are constant inside a given cell, but may vary between cells. The cells inside a block are simply connected, simplifying the calculations inside a block. The blocks themselves are also simply connected in order to make the sweep over the blocks straightforward even for parallel problems. However, cell interfaces between blocks may not be simply connected, and, as a result, require some special numerical techniques. The development of these techniques were discussed for the transport sweeps in Baker[1] and for DSA in Ward[2].

Any cell on a boundary of a given block in a Block-AMR mesh may contain from zero up to two

hanging nodes depending on how many neighboring blocks are at a higher level of refinement. Neighboring blocks may be no more than one level in difference from each other. Fig. 1 shows the first three cell-case types to illustrate the idea of hanging nodes. The first cell type is Case-0

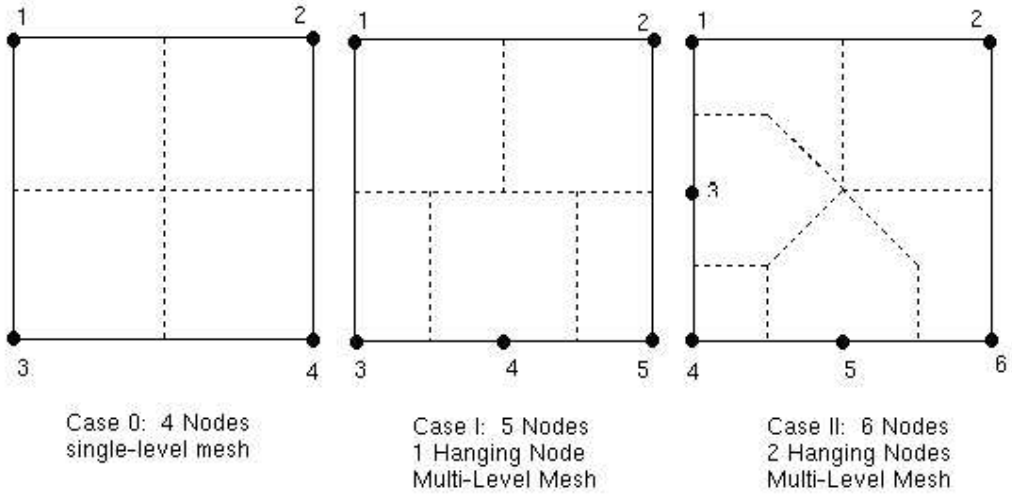

Figure 1. Three Block-AMR cell types (cases) and their median-mesh.

because it has no hanging nodes. Cells of the same or higher-refinement level than neighboring cells are considered Case-0. The second type of cell is Case-I and it has one hanging node. This case cell exists on the boundary of a block and is interfacing with a block that is at a higher level. The final cell case shown in Fig. 1 is a Case-II cell and it has two hanging nodes. This cell exists in a corner of a block that is interfacing with two blocks of a higher level.

3. DERIVATION OF THE BAMR-DSA METHOD IN TWO DIMENSIONS

The BAMR-DSA method is based on the DSA residual method for orthogonal single-level meshes. The single-level residual DSA method is derived from the source iteration form of the transport equation, which is, for one energy group and isotropic scattering[5],

$$\vec{\Omega}\cdot\vec{\nabla}\psi^{\ell+\frac{1}{2}}+\sigma_t\psi^{\ell+\frac{1}{2}}=\sigma_s\phi^{\ell}+q, \tag{1}$$

where ℓ is the iteration counter for source iteration, $\psi^{\ell+\frac{1}{2}}$ is the angular flux, ϕ^{ℓ} is the scalar flux from the previous iteration ℓ, q is the inhomogeneous source, σ_t is the total cross section and σ_s is the isotropic scattering cross section. The scalar flux $\phi^{\ell+\frac{1}{2}}$ is used along with ϕ^{ℓ}, from the previous iteration, to form the residual DSA equation,

$$-\vec{\nabla}\cdot D\vec{\nabla}\delta\phi^{\ell+\frac{1}{2}}+\sigma_a\delta\phi^{\ell+\frac{1}{2}}=\sigma_s(\phi^{\ell+\frac{1}{2}}-\phi^{\ell}), \tag{2}$$

where $D=\frac{1}{3\sigma_t}$ is the diffusion coefficient, $\sigma_a=\sigma_t-\sigma_s$ is the absorption cross section, and $\delta\phi^{\ell+\frac{1}{2}}$ is an estimate for the error in the scalar flux. To complete the DSA iteration, the scalar flux error $\delta\phi^{\ell+\frac{1}{2}}$ resulting from the solution of Eq. (2) is used to update the scalar flux at $\ell+\frac{1}{2}$ to produce

the final $\ell+1$ iteration scalar flux, i.e.,

$$\phi^{\ell+1} = \phi^{\ell+\frac{1}{2}} + \delta\phi^{\ell+\frac{1}{2}}. \tag{3}$$

The above Eqs. 1-3 are iterated upon until the maximum error at any one point from one iteration to the next is less than some tolerance.

The lack of simple connectivity in the Block-AMR mesh meant that the extension of DSA to Block-AMR was not straightforward[2]. Inversion of the left-hand-side transport operator is performed by sweeping along a face-centered mesh and using diamond-differencing to find the cell-centered angular flux values. DSA accelerates the source iterates on the right-hand side by solving a vertex-centered system of simultaneous algebraic equations for the scalar flux errors. The existence of hanging nodes in Block-AMR cells complicates this derivation of the vertex-centered system, which has cell-centered sources associated with the residuals derived from the cell-centered fluxes produced from sweeps.

In the Block AMR-DSA scheme, an equation is produced for each vertex scalar flux that represents a balance over the median-mesh of the cell associated with the vertex. The median-mesh in a cell is shown as the dashed lines in Fig. 1. The balance equations represent a diamond-like approximation to the surface integral form of the diffusion equation,

$$-\oint D\vec{\nabla}\phi\cdot\vec{n}\mathrm{d}A + \sigma_a\int_V \phi\mathrm{d}V = \int_V Q\mathrm{d}V, \tag{4}$$

where ϕ is the diffusion scalar flux, dV is the volume of the subcell defined by the median-mesh, and Q is the source or residual for the volume. The cross sections and source within the cells are assumed to be constant. The balance equations derived from Eq. 4 may be characterized as "partially" consistent. The derivation of a fully consistent[6] diffusion discretization with tranpsort diamond-differencing on AMR meshes[2] does not appear to be algebraically possible.

The BAMR-DSA method has been implemented and tested in two dimensions for rectangular (XY) and cylindrical (RZ) geometries. The diamond-difference method is applied to Eq. 4 for each node in each cell to produce a series of difference equations. For the Case-0 cell, the four difference equations in two-dimensional generalized geometry are,

$$\frac{-D}{\Delta x_1}(\phi_2-\phi_1)\Delta A_{x_2} - \frac{D}{\Delta x_2}(\phi_3-\phi_1)\Delta A^l_{x_1} + \sigma_a\phi_1\frac{\Delta A^l_{x_1}\Delta x_2}{2} = Q\frac{\Delta A^l_{x_1}\Delta x_2}{2}, \tag{5a}$$

$$\frac{-D}{\Delta x_1}(\phi_1-\phi_2)\Delta A_{x_2} - \frac{D}{\Delta x_2}(\phi_4-\phi_2)\Delta A^r_{x_1} + \sigma_a\phi_2\frac{\Delta A^r_{x_1}\Delta x_2}{2} = Q\frac{\Delta A^r_{x_1}\Delta x_2}{2}, \tag{5b}$$

$$\frac{-D}{\Delta x_1}(\phi_4-\phi_3)\Delta A_{x_2} - \frac{D}{\Delta x_2}(\phi_1-\phi_3)\Delta A^l_{x_1} + \sigma_a\phi_3\frac{\Delta A^l_{x_1}\Delta x_2}{2} = Q\frac{\Delta A^l_{x_1}\Delta x_2}{2}, \tag{5c}$$

$$\frac{-D}{\Delta x_1}(\phi_3-\phi_4)\Delta A_{x_2} - \frac{D}{\Delta x_2}(\phi_2-\phi_4)\Delta A^r_{x_1} + \sigma_a\phi_4\frac{\Delta A^r_{x_1}\Delta x_2}{2} = Q\frac{\Delta A^r_{x_1}\Delta x_2}{2}, \tag{5d}$$

where the differential areas and cell widths are dependent on the type of two-dimensional

geometry used. The rectangular (XY) geometry differential areas are,

$$\Delta A_{x_1} = \frac{\Delta x}{2} * \Delta z, \tag{6a}$$

$$\Delta A_{x_2} = \frac{\Delta y}{2} * \Delta z, \tag{6b}$$

with Δz equal to unity because the geometry is two-dimensional. For cylindrical geometry (RZ) the surface areas are,

$$\Delta A^l_{x_1} = \pi(r_i^2 - r^2_{i-\frac{1}{2}}) \tag{7a}$$

$$\Delta A^r_{x_1} = \pi(r^2_{i+\frac{1}{2}} - r_i^2) \tag{7b}$$

$$\Delta A_{x_2} = 2\pi r_i \frac{\Delta z}{2}, \tag{7c}$$

where $r_{i-\frac{1}{2}}$ and $r_{i+\frac{1}{2}}$ are the radial boundaries, r_i is the center of the cylindrical cell, and Δz is the cell size along the cylindrical axis. The cell widths Δx_1 and Δx_2 are Δx and Δy for rectangular and Δr and Δz for cylindrical geometry.

The four Eqs. 5 are conservative and produce an Symmetric Positive Definite (SPD) linear system. The derivation of the generalized two-dimensional geometry difference balance equations for the Case-I and -II cells is similar to the Case-0 cell. The balance equations and diffusion matrices for these two cases are discussed in Ward[2] along with a discussion of the derivation of the boundary conditions, implementation details, stability issues, and testing of the BAMR-DSA method.

4. SOLUTION OF BAMR-DSA SYSTEM WITH CONJUGATE GRADIENT METHOD

The prototype BAMR-DSA method[2] implemented in PARTISN stored the diffusion matrix in a dense format, where all elements (including zeros) were stored, and was solved using the LU-Decomposition method[3]. The storage and solution methods of the prototype are inefficient and slow. They were choosen in order to verify that any problems with the prototype BAMR-DSA method were due to the method and not the linear system solution technique. Since the prototype BAMR-DSA method has been verified[2], the focus of advancing the method now is to enhance its viability for use in large scale problems. In order to achieve this, the storage of the diffusion matrix and the solution of the BAMR-DSA system must be efficient and work in parallel. Neither the original dense matrix storage format or the LUD solution method meet these criteria.

Understanding the structure of the diffusion matrix is useful in the determination of storage and solution techniques. The diffusion matrix produced by the BAMR-DSA method is sparse, i.e. more zero elements than non-zero elements, and SPD. While the diffusion matrix is banded, its bandwidth and structure are unpredictable and mesh-dependent. As the number of nodes increases, the sparseness of the diffusion matrix becomes more pronounced. An example of a Block-AMR mesh used in the Fe-H_2O shielding test problem is shown in Fig. 2. This mesh has a minimum block-level of one and a maximum block-level of three. The structure of the diffusion

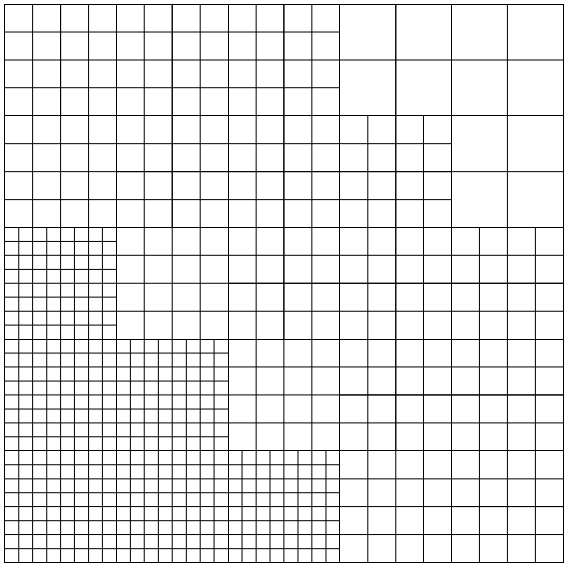

Figure 2. Example Block-AMR Mesh Used in the Fe-H_2O Shielding Problem

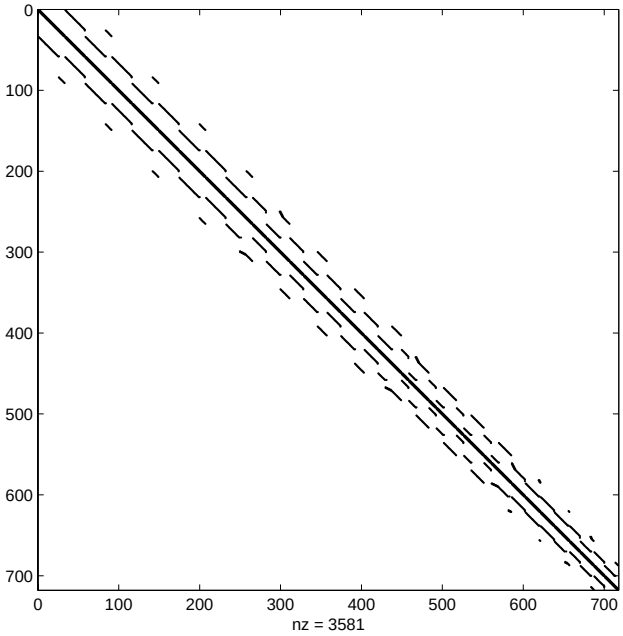

Figure 3. Structure of the example Block-AMR Mesh Used in the Fe-H_2O Shielding Problem

matrix for this mesh is shown in Fig. 3. The mesh has 569 nodes, producing a diffusion matrix with $569 \times 569 = 323761$ elements. Only 2881 elements in the diffusion matrix are non-zero, resulting in a matrix that is 99.1% zeros. This mesh is moderately sized compared to the problems that are expected to be run with the BAMR-DSA method.

The Compressed Sparse Row[4] (CSR) matrix storage method was chosen to reduce the diffusion matrix storage requiremenmts. The CSR storage method is implementable on a parallel computing environment and many linear system solvers have been adapted to solve sparse systems. The CSR method stores only the non-zero elements. These elements are stored in a single one-dimensional double precision array of size N_{nz}. The elements' matrix locations are stored in two integer arrays, one for the column location of each non-zero element and one that stores the number of columns with data for each matrix row. The size of these three arrays are, respectively, N_{nz}, N_{nz}, and $N_n + 1$, where N_n is the number of nodes in the Block-AMR mesh.

The original linear solver used for the solution of the prototype BAMR-DSA method was LU-Decomposition[2]. LUD is not an efficient linear solver becuase it requires $O(N^3)$ operations, so using it for solving large BAMR-DSA systems is impractical. As a result of this and the need to find an efficient parallel linear solver, a new linear system solution method was sought. The CG[4] method was chosen to solve the BAMR-DSA system because the diffusion matrix is SPD. The Conjugate Gradient (CG) method is an iterative solver that effectively solves sparse SPD linear systems[4]. CG is also parallelizable and works with many preconditioners.

5. RESULTS AND DISCUSSION OF THE UTILIZATION OF CSR AND CG FOR THE BLOCK AMR-DSA METHOD

The BAMR-DSA method's acceleration and solutions for both multilevel meshes and single-level meshes is comparable to the original five-point DSA method implemented in PARTISN[2]. Table I shows results from a set of runs of the Fe-H_2O shielding[1] test problem for square cells

Table I. Comparison of Single-Level and AMR Two-Dimensional Iron-H_20 Runs for XY Geometry and Square Cells

Mesh	Acceleration?	Total # Cells	# Iterations	Right Leakage	Particle Balance
s.l.	no	$1{,}600(40\times40)$	1,147	5.054×10^{-2}	3.354×10^{-4}
s.l.	dsa	$1{,}600(40\times40)$	61	5.093×10^{-2}	3.032×10^{-8}
s.l.	dsa	$6{,}400(80\times80)$	60	5.182×10^{-2}	-2.35×10^{-8}
C0,L3	no	$1{,}600(40\times40)$	1,147	5.054×10^{-2}	3.354×10^{-4}
C0,L3	dsa	$1{,}600(40\times40)$	55	5.092×10^{-2}	6.748×10^{-9}
ml234	no	2,656	1,160	5.035×10^{-2}	3.084×10^{-4}
ml234	dsa	2,656	106	5.074×10^{-2}	-1.90×10^{-9}

in the XY geometry and a convergence tolerance of 1×10^{-4}. The single-level (s.l.) runs are solved with the original PARTISN single-level transport solver and source accelerators. The Case-0 BAMR-DSA (C0,L3) runs are done with a 5×5 block mesh that is equivalent to the single-level 40×40 mesh, so each block is level 3 (8×8). The multilevel Block-AMR mesh's lowest-level block is a 2 (4×4) and the highest-level block is a 4 (16×16). The single-level 80×80 mesh run is included to determine if grid convergence was achieved and for use in comparison of the level-4 blocks of the ml234 mesh. All of the BAMR-DSA problems were run with the LUD solver.

The scalar fluxes for the BAMR-DSA and original single-level runs are within convergence criteria. The integral right leakages for each set of runs also agree well with each other. The

particle balance comparison for each set of runs confirms that, as expected, the DSA method produces a solution with better balance than nonaccelerated runs. As expected, the DSA runs required fewer iterations than the unaccelerated run. The larger number of iterations for the multilevel mesh runs when compared to single-level mesh equivalent runs is because of the hanging nodes present in a true Block-AMR mesh. However, this is still a large improvement over no acceleration.

As the number of nodes in a Block-AMR mesh increases, the ratio of zero-elements to total number of elements increases, i.e. the diffusion matrix becomes more sparse. The sparseness of the mesh presented in Fig. 2 was shown in Fig. 3. Table II shows a comparison of storage requriements for the diffusion matrix for both dense and CSR methods for various B-AMR

Table II. Comparison of Storage Requirments for Dense and CSR Matrices

Mesh	Mesh Type	$N_n(N_n^2)$	N_{nz}	Zero(%)	Stor.(dense)	Stor.(CSR)	$\frac{Dense}{CSR}$
1	ml122	$333^2(110889)$	1677	98.5	6.768 Mb	0.164 Mb	41.3
2a	C0,L2	$441^2(194481)$	2121	98.9	11.87 Mb	0.208 Mb	57.2
2b	C0,L3	$1681^2(2825761)$	8241	99.7	172.5 Mb	0.806 Mb	214.1
2c	ml123	$717^2(514089)$	3581	99.3	31.4 Mb	0.350 Mb	89.8
3	ml1-6	$7537^2(5.7*10^7)$	38081	99.93	3467 Mb	3.72 Mb	932.9

meshes. The mesh in Table II with the same numbers are similar with the only differences related to the levels of the various blocks. The mesh labeled 1, 2c, and 3 are all multi-level. Mesh #2c is the one shown in Fig. 2. The remaining mesh are single-level equivalent Block-AMR mesh. The example mesh shown in Table II are on the small side compared ot the mesh that will be typically used with BAMR-DSA method. Thus, diffusion matrices with over 99% zero elements are expected for our applications. The $\gtrsim$ 99% savings in storage space for the CSR method when compared to the dense storage method is necessary in order for the BAMR-DSA method to be practical for large mesh.

All of the mesh discussed in Table II were used on the Fe-H_2O shielding problem using both the LUD and CG methods on a single processor in order to determine the effectiveness of the CG method. The results of these runs are presented in Table III. The result marked with a (*),

Table III. Comparison of Conjgate Gradient versus LU-Decomposition for Various Block-AMR mesh in the Fe-H_2O Shielding Problem

Mesh	N_n	N_{nz}	N_{inners}^{LUD}	N_{inners}^{CG}	t_{LUD} (s)	t_{CG} (s)	$\frac{t_{LUD}}{t_{CG}}$
1	333	1677	76	76	7.32	0.73	10.0
2a	441	2121	64	64	21.01	0.9	23.3
2b	1681	8241	58	58	1352.9	8.78	154.1
2c	717	3581	61	63	103.4	1.89	54.7
3	7537	38081	3*	3*	51751	50.42	1026.4

mesh #3, was a diffusion only run which means that there was only one iteration per energy group.

In the process of testing the effectiveness of CSR-CG over dense-LUD, several Block-AMR mesh were developed in which, while both solver methods could solve each inner iteration's BAMR-DSA linear system, neither solver method would coverge. BAMR-DSA's failure to converge for certain Block-AMR mesh is shown in Fig. 4. This figure shows the maximum spatial flux change for each inner iteration of the third energy group for several Block-AMR

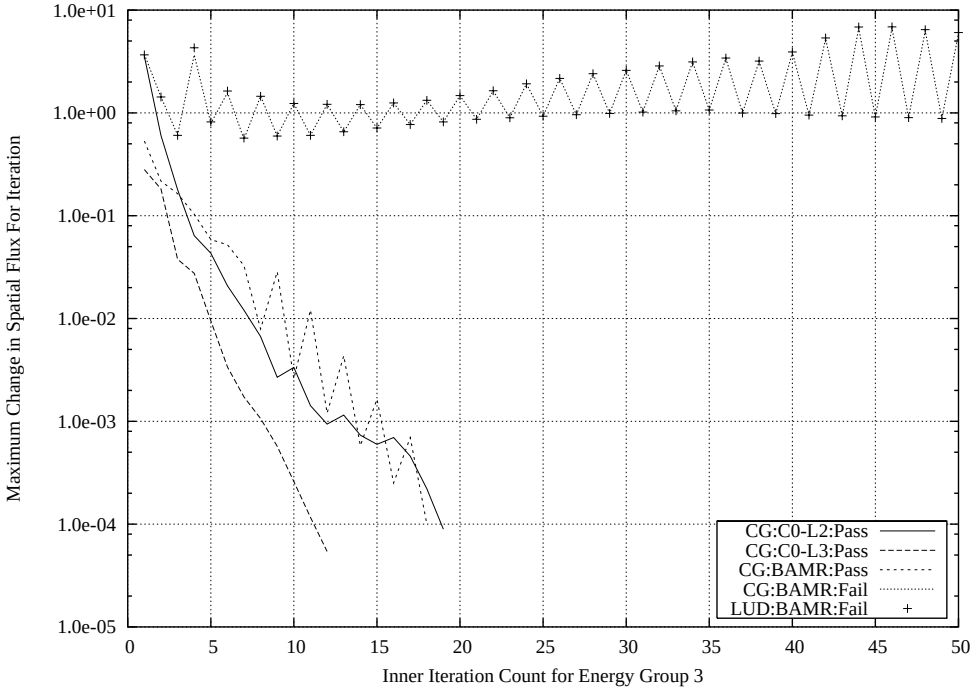

Figure 4. Convergence Comparison of Several Similar Block-AMR Meshes and the LUD and CG Solvers using the Block AMR-DSA Method on the Fe-H_2O Shielding Problem

mesh and the LUD and CG solvers. The problems occur in the third energy group for mesh that have too many blocks where the cell sizes are too large with a high scattering ratio ($c = 0.994$). By increasing the level of many of these blocks the numerical instabilities disappear and the method converges. Shown in Fig. 4 as "CG:BAMR:Pass", mesh #2c, where most of the blocks are level 2 or 3, converges. However, a multi-level mesh similar to mesh #2c, but with at least one more level-1 block (CG:BAMR:Fail and LUD:BAMR:Fail), diverges. For runs where the BAMR-DSA method is not converging switching off DSA when the inner iteration results start to oscillate or diverge may allow convergance. In this case, turning off BAMR-DSA in the third energy group after inner iteration #10 and doing only sweeps to finish converging to $1 * 10^{-4}$ required a total of 391 iterations. While not a very good comparison to the 19 inner iterations for mesh #2c, that is still fewer than the 1052 inner iterations required for sweeps alone.

6. CONCLUSIONS

The results of the testing of the original prototype BAMR-DSA method was discussed in Ward[2]. This testing confirmed that a partially-consistent BAMR-DSA method will introduce instabilities for extreme cases, e.g., scattering ratios approaching 1.0 with optically thick cells,

but for most realistic problems the BAMR-DSA method provides a effective acceleration for transport on a Block-AMR mesh.

While the above convergence issues for several of the Block-AMR mesh is troubling, overall the BAMR-DSA method works very well for many problems. However, care must be taken when choosing the block-levels for problems with high scattering ratios. In order to be useful, BAMR-DSA must be used in conjunction with automatic mesh coarsening[7] and stability monitoring.

The diffusion matrix produced by the BAMR-DSA method is sparse and SPD. While the diffusion matrix is banded, its bandwidth and structure are unpredictable and mesh-dependent. As the number of nodes on a mesh increases, the matrix becomes more sparse. The CSR[4] matrix storage method was chosen to store the diffusion matrix because of the sparse nature of the diffusion matrix. Over 99% computer storage savings is achievable for large Block-AMR mesh. This savings is very important in order to run many of the problems which PARTISN is designed to solve. In addition, since PARTISN is a parallel computing code, the CSR storage method's adaptibility to parallel solving techniques will be important.

The CG[4] method was choosen to solve the BAMR-DSA linear system because the diffusion matrix is SPD. CG is implementable in parallel computing environments and works on sparse systems. The CG method was able to achieve a factor of 1000 speedup over LUD for solving a complete transport problem with DSA on a Block-AMR mesh. The faster solution time of CG combined with the smaller storage requirements of CSR are promising results for the BAMR-DSA method's full incorporation into PARTISN.

ACKNOWLEDGMENTS

This work was performed at the Los Alamos National Laboratory (LANL) under contract W-7405-ENG-36 with the U.S. Department of Energy.

REFERENCES

1. R. S. Baker, "A Block Adaptive Mesh Refinement Algorithm for the Neutral Particle Transport Equation," *Nucl. Sci. Eng.*, **141**, pp. 1–12 (2002).
2. R. C. Ward, R. S. Baker, J. E. Morel, "A Diffusion Synthetic Acceleration Method for Block Adaptive Mesh Refinement," *Nucl. Sci. Eng.* (2004).
3. W.H.Press, et al., *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, Cambridge, MA (1986).
4. Y. Saad, *Iterative Methods for Sparse Linear Systems 2nd ed.*, SIAM, Philadelphia (2003).
5. E. Lewis, J. W.F. Miller, *Computational Methods of Neutron Transport*, American Nuclear Society Inc., Lagrange Park, IL (1993).
6. E. E. Larsen, "Asymptotic Diffusion Limit of Discretized Transport Problems," *Nucl. Sci. Eng.*, **112**, pp. 336–346 (1992).
7. S. Turner, "Automatic Mesh Coarsening for Discrete Ordinates Codes," *Transactions of the Mathematics and Computation, Reactor Physics and Environmental Physics Conference*, Madrid, Spain, September 1999 (1999).