ANGULAR MULTIGRID ACCELERATION FOR PARALLEL S_N METHOD WITH APPLICATION TO SHIELDING PROBLEMS

Vefa Kucukboyaci and Alireza Haghighat
Mechanical and Nuclear Engineering Department
Penn State University
231 Sackett Building
University Park, PA 16802
E-mail: vnk102@psu.edu

ABSTRACT

New acceleration formulations, which are compatible with the adaptive differencing strategy and parallel environments, are required. For this purpose, we have developed the Simplified Angular Multigrid (SAM), Nested Iteration (NI), and V-Cycle algorithms, and their various combinations, and implemented them in the PENTRAN™ 3-D Parallel S_N code. These formulations have been examined for a test problem using a variety of parameters including different scattering ratios, coarse- and fine-grid convergence tolerances and quadrature orders. The effectiveness of the new multigrid formulations is compared to the standard Partial Current Rebalance (PCR) acceleration method. Thus far, the combinations of SAM and PCR or NI and PCR have proved to be very effective for a large range of c-ratios. We have achieved speedups as high as a factor of ~7.3 in the number of iterations, or a factor of ~4.0 in the CPU time. Preliminary studies indicate that other combinations such as SAM+V-cycle+PCR or NI+V-cycle+PCR can be even more effective in certain problem conditions.

1. INTRODUCTION

In problems with optically thick regions and high scattering ratios (c-ratio=σ_s/σ_t), particles that are making a large number of scattering collisions in a single energy group contribute significantly to scalar flux distribution. Consequently, the convergence of the source iteration (SI) method can become very slow. Several techniques such as Rebalance [*1*], diffusion synthetic acceleration (DSA) [*2*] and multigrid (MG) [*3*] methods have been devised to remedy the slow convergence of the SI method for both shielding and criticality problems.

Rebalance techniques (System Rebalance (SR) and Coarse Mesh Rebalance (CMR)) use the fact that the converged solution must satisfy the neutron conservation (or balance) equation. By imposing this balance condition on the unconverged solution over coarse regions of the problem domain, it is possible to obtain an iteration procedure that may result in faster convergence to the correct solution. Rebalance techniques are effective for deep-penetration problems, however two difficulties are associated with them: i) They suffer from convergence instability; ii) Selection of

an optimum coarse-mesh in the rebalance methods is usually difficult, especially in parallel computing environments where spatial domain decomposition is imposed. The Partial Current Rebalance (PCR) [*4*] method reduces this instability by introducing a damping parameter.

Unlike the rebalance methods, the DSA method works well in eigenvalue problems with high c-ratios. However it is not as effective in low c-ratio shielding problems. In the DSA method, transport solution is used to correct terms in the diffusion equation, and the diffusion solution is used to obtain an improved source for the transport equation. In this method, a diffusion formulation consistent with the S_N formulation is required. This means that if the differencing scheme of the transport equation is changed, a new formulation has to be derived for the diffusion solver. Derivation of consistent diffusion formulations becomes difficult especially in three-dimensional (3-D) geometries and with an adaptive differencing strategy [*5*].

In the remaining of this paper, we present: i) a discussion on the general multigrid methods, ii) the new angular multigrid methods implemented in the PENTRAN™ code [*6*], iii) numerical tests using a 3-D benchmark problem, and v) results and analysis.

2. GENERAL MULTIGRID METHODS

In multigrid methods, a sequence of *coarse* and *fine-grid*s is used to remove different modes of error from the estimate of the solution. The problem converges when the error remaining in the solution estimate is less than some predefined tolerance. We can express the solution (i.e. angular flux) and the associated error as a function of frequency rather than space or direction by applying the Fourier transform [*7*]. This representation facilitates the understanding of how the components of the error are removed by iterations. Assume a coarse and a fine discretization of the same domain as seen in Figure 1. The error modes behave differently on the two grids. The low-frequency error on the fine-grid becomes a high-frequency error on the coarse-grid. If we were to solve the transport equation on this coarser grid, then we would have a good approximation to the low-frequency components of fine-grid solution. If we can couple the coarse- and fine-grid solutions, then the convergence rate on the fine-grid will be governed only

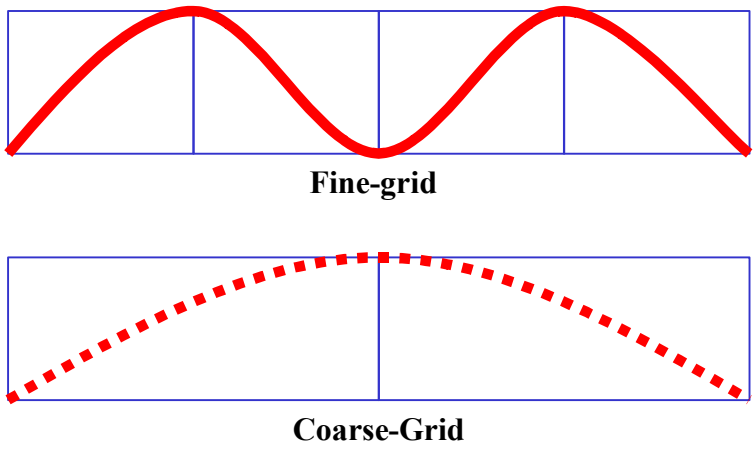

Figure 1: High- and low-frequency errors on fine- and coarse-grids

by the high-frequency errors. Since the high frequency errors are attenuated more rapidly than the low-frequency errors, we have effectively accelerated the overall convergence.

Several different types of multigrid methods have been developed and tested in different disciplines. Few of the examples are *V-cycle*, *W-cycle*, *Nested iteration* and */-cycle*. Multigrid methods have been applied to transport calculations by using it directly for the S_N equations [*3, 8, 9*] and/or for the DSA equations [*10*]. Many of these applications have been limited to one or two-dimensional problems due to large memory requirements. Recently, an angular multigrid formulation has been developed for highly anisotropic scattering, especially for charged-particle transport problems, both in 1-D and 3-D geometries [*11, 12*]. Our method described in this paper is different in the sense that it is more general and can be applied to any shielding and criticality calculations.

3. ANGULAR MULTIGRID SCHEMES

Simplified Angular Multigrid (SAM)

The first angular multigrid scheme we have developed is a */-cycle*, which is called the Simplified Angular Multigrid (SAM) scheme. In the SAM scheme, a global approximate solution (i.e., angular fluxes) is obtained on a coarse angular grid (e.g., S_4/ P_0, P_1), and then this solution is projected onto a fine angular grid (e.g., S_{10}/ P_5) filtering out the low frequency error components. Effectively, the calculation on the coarse-grid provides preconditioning for the fine-grid iterations.

Using a two-grid approach, PENTRAN first performs a group source iteration/sweep over a coarse angular grid denoted by Ω^{2h}. The zeroth moment balance equation is given by:

$$\frac{|m_m|^{2h}}{\Delta x}(\Psi^{2h}_{out,x}-\Psi^{2h}_{in,x})+\frac{|h_m|^{2h}}{\Delta y}(\Psi^{2h}_{out,y}-\Psi^{2h}_{in,y})+\frac{|x_m|^{2h}}{\Delta z}(\Psi^{2h}_{out,z}-\Psi^{2h}_{in,z})+s\Psi^{2h}_{A}=q^{2h}_{A} \quad (1)$$

where q^{2h}_{A} includes scattering, external and fission sources. Inner iterations on Ω^{2h} are continued until convergence is achieved (Note that, convergence on the coarse-grid is less strict compared to fine-grid (Ω^{h})). Then, all the coarse angular fluxes are projected onto the fine angular grid:

$$\tilde{\Psi}^{h}=P^{2h\to h}\Psi^{2h} \quad (2)$$

Here, $P^{2h\to h}$ is the projection operator (coarse angular grid to fine angular grid). For this projection, we select the angular flux on a particular direction on Ω^{2h} that is closest to a direction on Ω^{h}. The angle between direction vectors of Ω^{2h} and Ω^{h} can be written in terms of direction cosines:

$$\cos(a^{2h\to h}_{m,n})=m^{2h}_{m}m^{h}_{n}+h^{2h}_{m}h^{h}_{n}+x^{2h}_{m}x^{h}_{n} \qquad \text{where} \quad \begin{matrix} n=1,...N(N+2)\in\Omega^{h} \\ m=1,...M(M+2)\in\Omega^{2h}, M<N \end{matrix} \quad (3)$$

By finding the minimum of these angles $a^{2h\to h}_{m,n}$, we determine the closest direction.

In case there is more than one minimum angle, the angular fluxes are determined by performing simple arithmetic mean of the fluxes in these directions.

In order to conserve particles, we must guarantee that the integral quantities (i.e., scalar fluxes) rendered on both coarse and fine angular grids are equal. In order to achieve balance we normalize the projected angular fluxes:

$$\Psi_n^h = \tilde{\Psi}_n^h \frac{\sum_{m=1}^{M(M+2)} w_m^{2h} \Psi_m^{2h}}{\sum_{n=1}^{N(N+2)} w_n^h \tilde{\Psi}_n^h} \quad (4)$$

Using the projected angular fluxes, the scattering source and the boundary angular fluxes (in case of spatial domain decomposition) on the fine-grid are updated. Then, the iterations/sweeps are continued on the fine angular grid (Ω^h) until convergence is achieved.

Nested Iteration (NI)

A variation of the SAM scheme is the *Nested Iteration*, in which we use successively refined multiple angular grids (e.g., Ω^{8h}, Ω^{6h}, Ω^{4h}, Ω^{2h}, Ω^h). We start on the coarsest angular grid (e.g. Ω^{8h}) and solve for angular fluxes within certain convergence tolerance. These angular fluxes are then used as the initial solution for the next finer grid. This process is continued until we converge on the finest grid (Ω^h). Note that the convergence tolerance for the coarser grids should not be as small as the finer grids. This issue is examined later in this paper.

Both SAM and Nested have the following features:

- *Efficiency*: The number of operations per mesh is significantly lower for the coarse angular grid compared to the fine angular grid.
- *Memory requirement*: SAM and Nested Iteration are not cyclic algorithms; all angular flux arrays are overwritten when iterations upgrade to a finer grid. Therefore, no extra memory is required.

V-Cycle

The V-cycle algorithm uses a two-grid scheme. We, first perform an iteration on the fine angular grid, and compute the difference between the previous and the current iteration scattering sources for each cell and direction. This difference is called the *residual*. Residuals are then expanded into moments to be used as source on the coarse angular grid. We then perform a sweep on the coarse angular grid to render the error terms. Using *the closest direction* approach, these error terms are projected back to the fine-grid to update the angular fluxes and the scattering source. We, then proceed to the next iteration with the updated source. We cycle between the two grids until a converged solution on the fine angular grid is obtained. The following algorithm summarizes the angular multigrid V-Cycle:

- Sweep $H^h \Psi^h = q^h$ on Ω^h with the initial guess Ψ^h
  - Compute residual $r^h = q^h - q^h_{old}$
    - Sweep $H^{2h} e^{2h} = P^{h \to 2h} q^h$ on Ω^h with the initial guess $e^h = 0$
  - Update fluxes $\tilde{\Psi}^h = \Psi^h + P^{2h \to h} e^{2h}$ and scattering source $q^h \to \tilde{q}^h$
- Sweep $H^h \Psi^h = \tilde{q}^h$ on Ω^h with the initial guess $\tilde{\Psi}^h$

where H is the transport operator, r is the residual, e is the error, P is the projection operator. h and $2h$ represent fine- and coarse-grids respectively, and *tilde* represents the updated values.

Unlike the SAM and Nested Iteration schemes, in the V-cycle scheme, the angular fluxes on the fine-grid are saved, since they are updated with the error terms computed on the coarse-grid. Saving the angular fluxes imposes extra memory requirement, which can be compensated by the possible increase in the rate of solution convergence. It is important to note that all the angular multigrid algorithms described are compatible with the parallel memory structure and the adaptive differencing strategy of the PENTRAN™ code.

It is also possible to combine the V-cycle with the SAM or NI schemes. SAM or NI effectively provides the initial solutions for the fine-grid. Figure 2 depicts the V-cycle and its combination with SAM or NI.

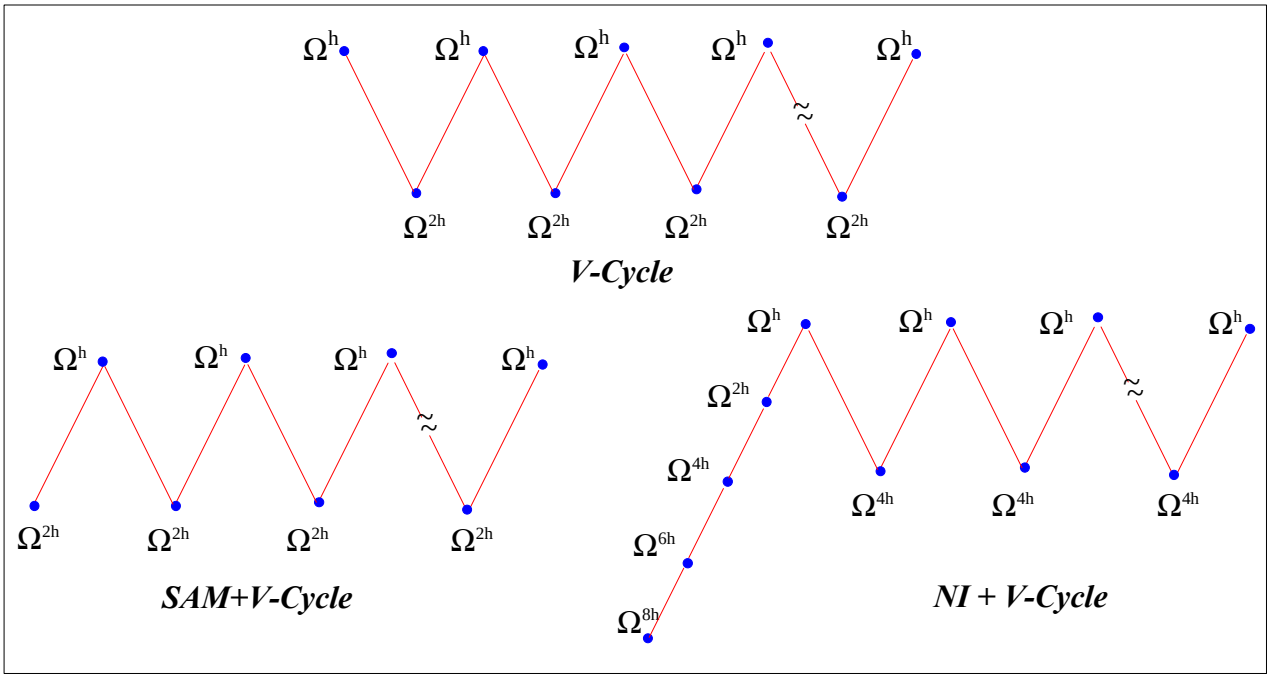

Figure 2: V-cycle and its combination with SAM or NI.

4. FEATURES OF THE PENTRAN™ CODE SYSTEM

PENTRAN™ is a 3-D parallel S_N code with complete, automatic phase space (angle, energy and space) decomposition for distributed memory and computing environments. It has been developed in Fortran-90 using MPI (message passing interface) library [*13*] and has been implemented on different platforms such as IBM SP2, SUN multi-processors, and PC clusters. Unique features and algorithms of PENTRAN™ include: Complete phase space decomposition, parallel I/O, partitioned memory structure, adaptive differencing strategy (including Directional-Theta-Weighted (DTW) and Exponential-Directional-Weighted (EDW) schemes), simplified angular and spatial multigrid acceleration schemes, and variable meshing with Taylor Projection Mesh Coupling (TPMC). Pre-and post-processing components included in the PENTRAN™ code system, such as PENMSH (automatic mesh and source generation), PENINP (automatic input generation), PENDATA, and PENPRL (flux data extraction and processing) [*14*] greatly facilitate the task of generating large 3-D models.

5. NUMERICAL TESTS

In this section, we measure the performance of the angular multigrid schemes for different problem parameters such as c-ratio, coarse and fine angular grid quadrature orders, and convergence tolerances. We utilize problem 1 of the Kobayashi 3-D deterministic transport benchmark problems [*15*]. Figure 3 shows the geometry of the problem. Reflective boundary condition is prescribed on planes x=0, y=0 and z=0, and vacuum boundary condition is prescribed on all other surfaces. A volumetric unit source is located in the region bounded by x=0-10cm, y=0-10cm, and z=0-10cm. For this study, we have used an S_{20} level-symmetric angular quadrature set. Scattering is isotropic, and we have analyzed cases with different c-ratios ranging from 0.6 to 0.99. Table 1 shows the total cross sections, fine mesh thickness, and

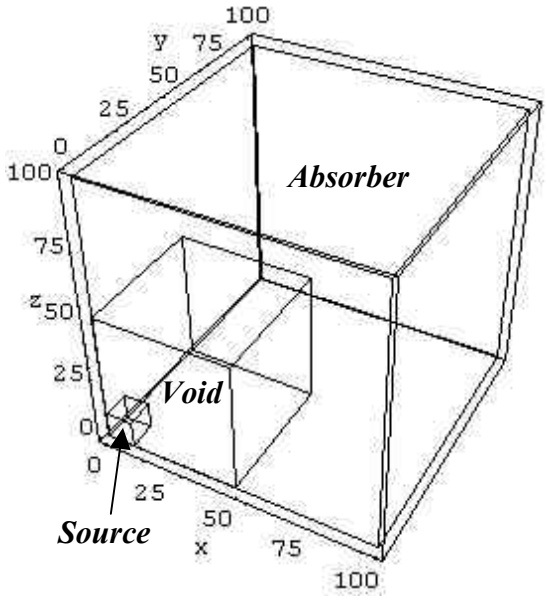

Figure 3: Kobayashi 3-D benchmark problem

differencing schemes used in different regions of the problem. Detailed studies [*16*] on this benchmark problem have demonstrated that the Directional Theta Weighted (DTW) differencing scheme is adequate for the source and void regions (small flux gradients), while the Exponential Directional Weighted (EDW) differencing scheme is adequate for the absorber regions (large flux gradients).

Table 1: Kobayashi benchmark problem specifications

Region	**s (cm^{-1})**	**Dx-Dy-Dz**	**Differencing Scheme**
Source	1.e-01	1.0cm	DTW
Void	1.e-04	10.0cm	DTW
Absorber	1.e-01	10.0cm	EDW

For parallel processing of this problem, we have partitioned the angular domain into four sub-domains (2 octants/ processor) and processed them on 4 processors of the LIONX parallel PC Cluster at Penn State University.

6. RESULTS AND ANALYSIS

Here, we investigate how convergence tolerances, c-ratio, and quadrature order on the coarse and fine-grids affect the performance of the SAM, NI, V-cycle and the combined algorithms. We also compare the effectiveness these new algorithms to that of PCR. We measure the performance by iteration and CPU speed-up. Note that

$$Iteration\ Speedup = \frac{Number\ of\ \ fine\ grid\ iterations\ without\ acceleration}{Number\ of\ \ fine\ grid\ iterations\ with\ acceleration},$$

while

$$CPU\ Speedup = \frac{CPU\ time\ without\ acceleration}{CPU\ time\ with\ acceleration}.$$

Effect of Coarse- and Fine-grid Tolerances

In this test, we have determined the effect of coarse- and fine-grid tolerances for the SAM scheme. For a fixed c-ratio of *0.9*, S_{20} for fine-grid and S_{10} for coarse-grid, we have varied the coarse-grid convergence tolerance in a range of *1.e-01* to *1.e-06* and the fine-grid tolerance in the range of *1.e-03* to *1.e-06*. In Fig. 4, we compare the iteration speed-up for the SAM scheme. For all fine-grid tolerances, we observe that SAM scheme becomes more effective with a tighter coarse-grid tolerance. However, note that for the coarse-grid tolerances below *1.e-04*, no further speed-up is obtained. We also observe that as the tolerance becomes tighter on the fine-grid, SAM becomes less effective, regardless of the coarse-grid tolerance. In Fig. 5, we compare the CPU speed-up. We see a relation between the coarse- and fine-grid tolerances, i.e., as we reduce the fine-grid tolerance, coarse-grid tolerance should also be reduced. Also note that, in order to

obtain the maximum CPU speedup, the coarse-grid tolerances should be in the range of *1.e-03 – 1.e-04*.

Effect of c-ratio

Using a fixed fine-grid convergence tolerance of *5.e-04*, S_{20} for the fine-grid and S_{10} for the coarse-grid, we have performed tests for SAM with different c-ratios ranging from *0.6* to *0.99*. The coarse-grid convergence tolerance is varied from *5.e-01* to *5.e-04*. As seen in Fig. 6, SAM becomes more effective with the increasing c-ratio, resulting in a significant acceleration as high as ~7.8. SAM outperforms PCR by a factor of *~2.6* in iteration speed-up, however as shown in Fig. 7, it performs similar to PCR for CPU speed-up.

Figure 8 compares the acceleration by PCR and the combination of SAM and PCR. Using PCR with SAM decreases the number of coarse-grid iterations, thereby increasing the overall efficiency of the angular scheme. This combination accelerates the calculation by a factor of ~4 while PCR alone achieves a speedup ~2.9 for c-ratio of *0.99* and the coarse-grid tolerance of *1.e-03*. We have repeated this test using the NI scheme combined with PCR (Fig. 9). In NI, we have started the calculations on S_4 grid, progressing to S_{10} and applied PCR on each grid. As seen in Fig. 10, we have obtained similar results as the SAM and PCR combination for large c-ratios, but the effectiveness decreases for small c-ratios and the small coarse-grid tolerances. Figure 10 also suggests that performance of NI is less sensitive to the coarse-grid tolerance.

Effect of Coarse- and Fine-grid Quadrature Orders

For a fixed c-ratio of *0.9* and S_{20} for the fine-grid, we have performed tests for the SAM scheme using a range of coarse-grid quadrature orders and convergence tolerances. Examining the iteration speed-up behavior, we observe that there is a relation between the iteration speed-up, coarse-grid quadrature order and tolerance. As we either increase the quadrature order or decrease the tolerance, we get better speedups. As seen in Fig. 10, behavior of the CPU speed-up is rather different. Beyond S_8 and tolerances below *5.e-03*, the efficiency of SAM decreases due to a higher computational effort on the coarse-grid.

Table 2 provides information on relation between the coarse- and the fine-grid quadrature orders in terms of the CPU speed-up. This test has been performed for a c-ratio of *0.6*, and coarse and fine-grid convergence tolerances of *5.e-02* and *5.e-04*, respectively. Table 2 indicates that for an effective acceleration for problems with fine-grid quadrature orders up to S_{10}, the coarse-grid quadrature order should be close to fine-grid quadrature order. Beyond S_{10} for the fine-grid, the coarse-grid quadrature orders should not be greater than S_8 or $S_{10.}$

Combinations of Angular Multigrid Formulations

In Table 3, we summarize various combinations of the angular multigrid formulations and the PCR acceleration. This test has been performed for a c-ratio of *0.6*, coarse and fine-grid tolerances and quadrature orders of *5.e-02/5.e-04*, and S_{10}/S_{20}, respectively. For the Nested Iteration (NI), we have started on S_4, gradually upgrading to S_{10}. Table 3 indicates that angular multigrid formulations combined with PCR become very effective. SAM combined with PCR reduces the CPU by a factor of *~3.43*, while PCR alone reduces by a factor of *~2.38*. The

combination of V-cycle, SAM and PCR can significantly reduce number of fine-grid iterations, however, because of the high cost of V-cycle, is not as effective in reducing the CPU time.

Table 2: Relation between coarse- and fine-grid quadrature orders for the SAM scheme [a]

Coarse-grid Quadrature Order	**Fine-grid Quadrature Order**							
	S_6	**S_8**	**S_{10}**	**S_{12}**	**S_{14}**	**S_{16}**	**S_{18}**	**S_{20}**
S_4	1.12	1.11	1.13	1.19	1.21	1.17	1.22	1.21
S_6	-	1.22	1.16	1.29	1.24	1.21	1.27	1.24
S_8	-	-	1.24	1.26	1.23	1.30	1.37	1.34
S_{10}	-	-	-	1.29	1.29	1.27	1.32	1.40
S_{12}	-	-	-	-	1.26	1.28	1.30	1.25
S_{14}	-	-	-	-	-	1.24	1.30	1.28
S_{16}	-	-	-	-	-	-	1.25	1.21
S_{18}	-	-	-	-	-	-	-	1.14

[a] *c-ratio=0.9,* coarse- and fine-grid convergence tolerances of *5.e-02* and *5.e-04*, respectively.

Table 3: Comparison of speedups obtained by combined formulations [a]

	ITERATION SPEED-UP	CPU SPEED-UP
NO ACCELERATION	1.00	1.00
PCR	2.43	2.38
SAM	2.76	1.52
SAM+PCR	5.41	3.24
NI	1.83	1.37
NI+PCR	5.62	3.39
V-cycle	1.74	1.28
V-cycle+PCR	4.87	3.43
V-cycle+SAM	3.24	1.57
V-cycle+SAM+PCR	7.30	3.39
V-cycle+NI	2.76	1.58
V-cycle+NI+PCR	6.95	2.45

[a] *c-ratio=0.6,* S_{10} for coarse-grid (for SAM and V-cycle), S_4 to S_{10} for the NI coarse-grids, S_{20} for fine-grid, and coarse- and fine-grid convergence tolerances of 5*.e-02* and *5.e-04*, respectively.

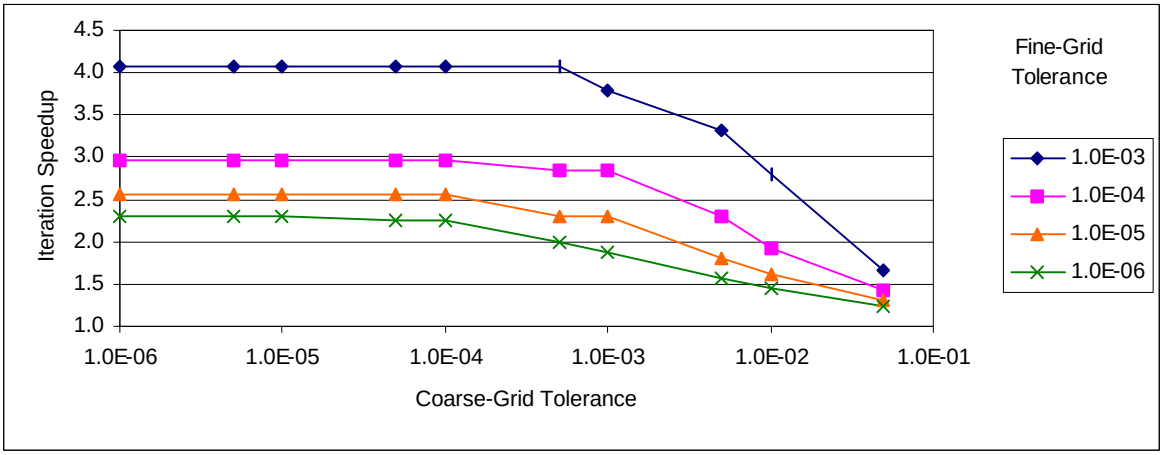

Figure 4: Iteration speedup with SAM for different coarse and fine-grid convergence tolerances [a]

[a] *c-ratio=0.9*, S_{10} for the coarse-grid, and S_{20} for the fine-grid

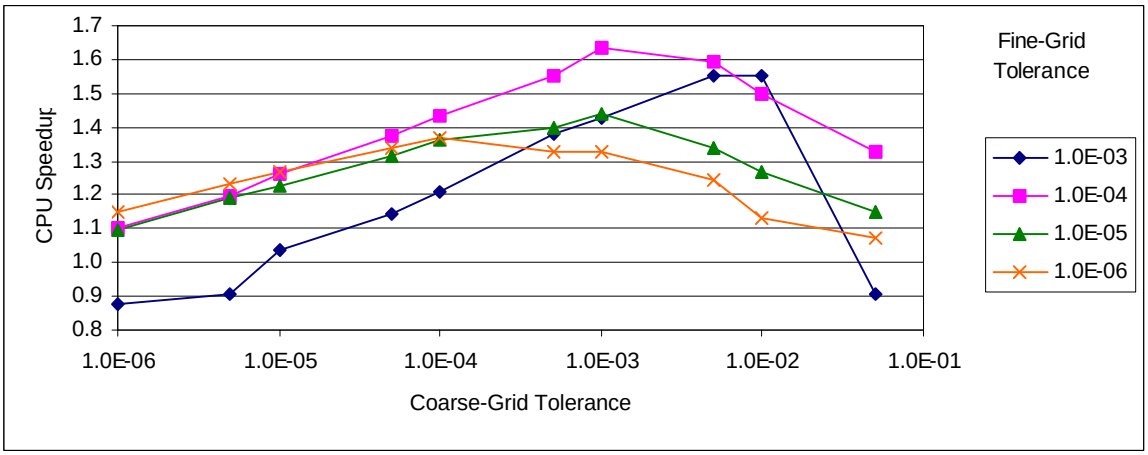

Figure 5: CPU speedup with SAM for different coarse and fine-grid convergence tolerances [a]

[a] *c-ratio=0.9*, S_{10} for the coarse-grid, and S_{20} for the fine-grid

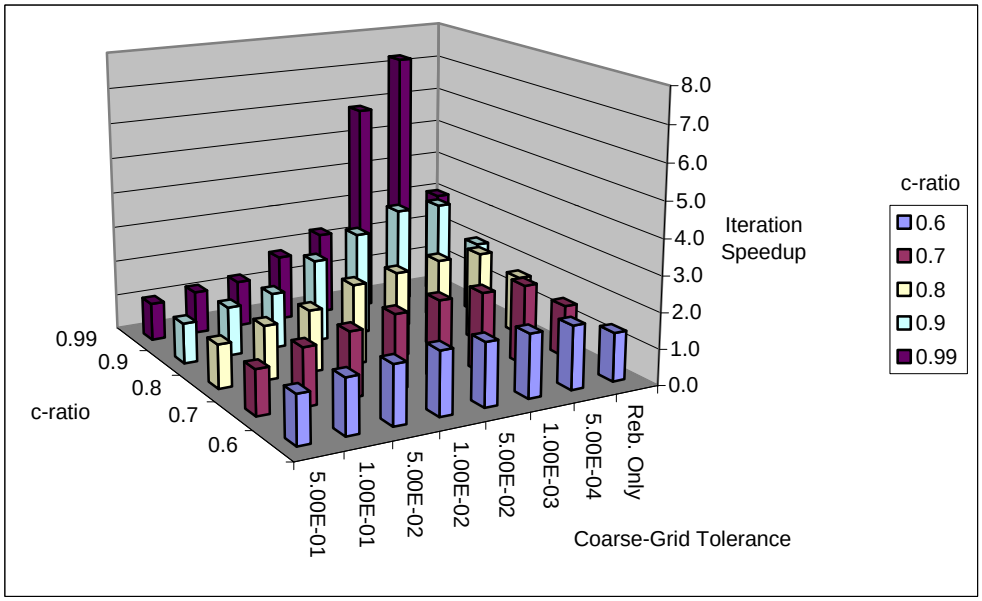

Figure 6: Iteration speedup with SAM compared to PCR for different c-ratios and coarse grid convergecne tolerances [a]

[a] S_{10} for the coarse-grid, S_{20} for the fine-grid, and fine-grid convergence tolerance of *5.e-04*

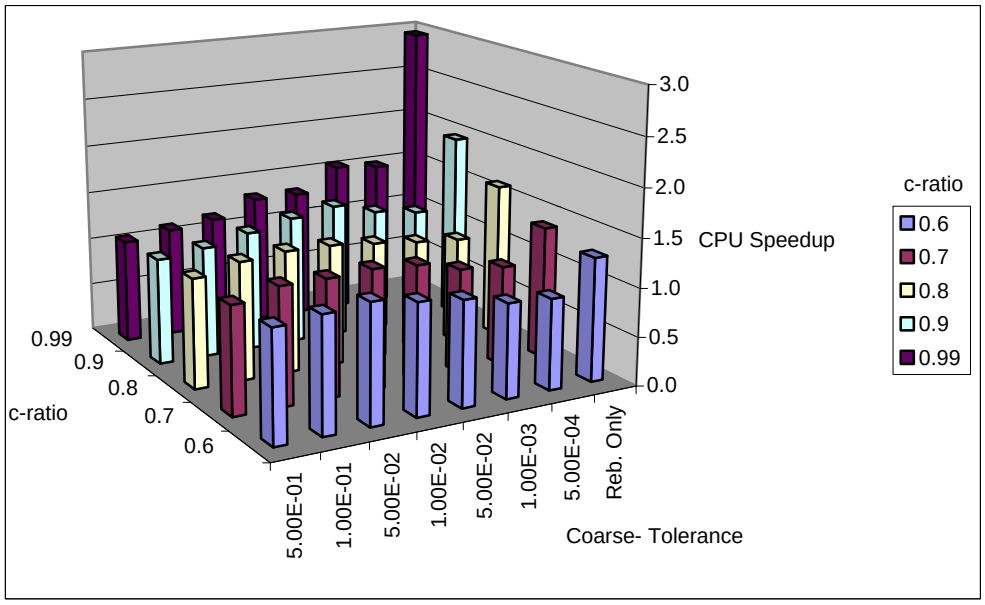

Figure 7: CPU speedup with SAM compared to PCR for different c-ratios and coarse grid convergence tolerances [a]

[a] S_{10} for the coarse-grid, S_{20} for the fine-grid, and fine-grid convergence tolerance of *5.e-04*

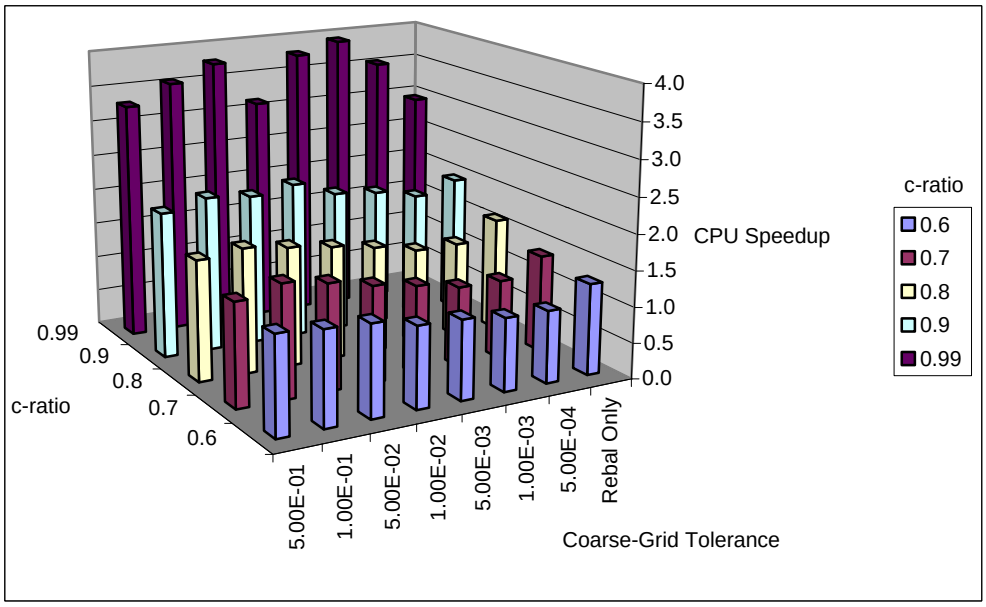

Figure 8: CPU speedup of SAM+PCR compared to PCR alone for different c-ratios and coarse grid convergecne tolerances [a]

[a] S_{10} for the coarse-grid, S_{20} for the fine-grid, and fine-grid convergence tolerance of *5.e-04*

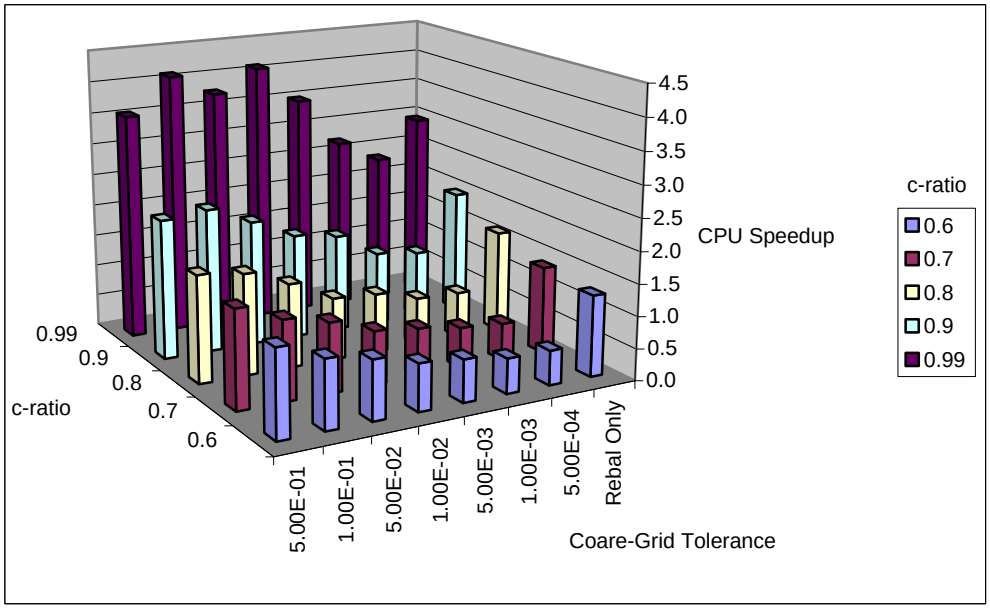

Figure 9: CPU speedup with NI+PCR compared to PCR alone for different c-ratios and coarse grid convergecne tolerances [a]

[a] S_4 to S_{10} for the NI coarse-grids, S_{20} for the fine-grid, and fine-grid convergence tolerance of *5.e-04*

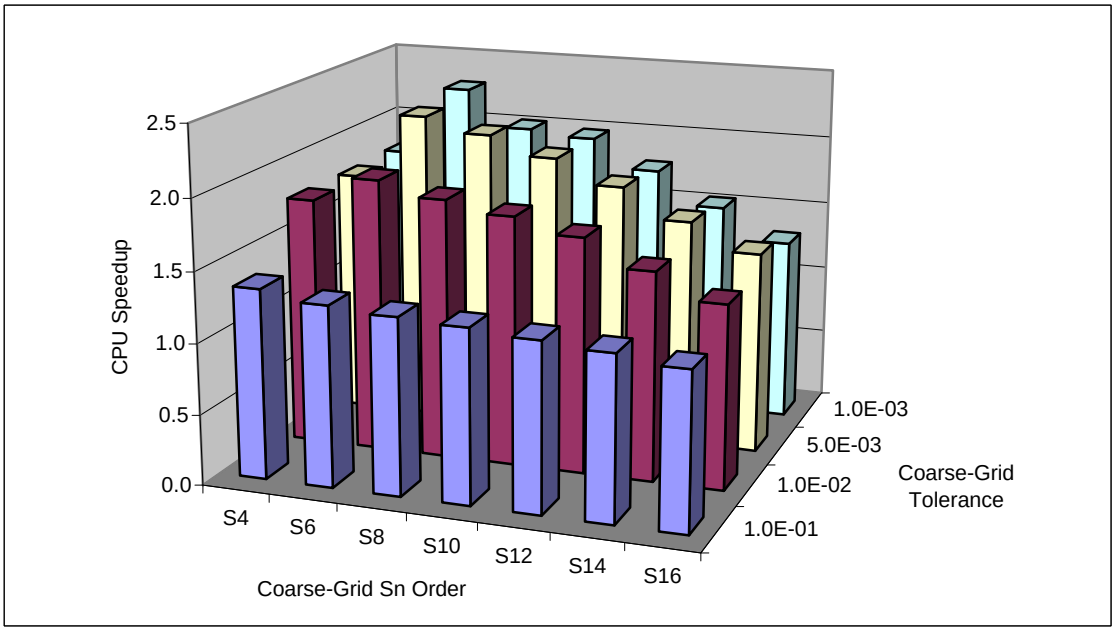

Figure 10: CPU speedup with SAM for different coarse grid quadrature orders and convergence tolerances[a]

[a] *c-ratio=0.9,* S_{20} for the fine grid and fine-grid convergence tolerance of *5.e-04*

SUMMARY AND CONCLUSION

We have developed the Simplified Angular Multigrid (SAM), Nested Iteration (NI), and V-Cycle angular multigrid algorithms that are compatible with the adaptive differencing strategy and parallel environment of the PENTRAN™ 3-D Parallel S_N code. These formulations have been examined for a test problem using a variety of parameters including scattering ratios, coarse and fine-grid convergence tolerances and quadrature orders. In comparison to the standard PCR formulation, combinations of PCR with SAM or NI have proved to be very effective for a large range of c-ratios. Preliminary studies indicate that other combinations with V-cycle can be effective even in low c-ratios. Further work is under way for testing the new angular multigrid formulations for criticality problems and real life problems such as the VENUS-3 benchmark and BWR core shroud problem.

ACKNOWLEDGEMENTS

The authors acknowledge that this work has been performed under a DOE NEER grant. Further, they express their appreciation for the CPU time at the Penn State LIONX Parallel PC-Cluster.

REFERENCES

1. Miller, W., "Generalized Rebalance: A Common Framework for Transport Acceleration Methods," ***Nuclear Science and Engineering***, 65: 226-236 (1978)
2. Alcouffe, R., " Diffusion Synthetic Acceleration Methods for the Diamond Differenced Discrete Ordinates Equation", ***Nuclear Science and Engineering***, 64: 344-355 (1977)
3. Barnett, A., J. Morel and D. Harris, "A Multigrid Acceleration Method for the One-Dimensional S_N Equations with Anisotropic Scattering", ***Nuclear Science and Engineering***, 102:1 (1989).
4. Rhoades, W., "Improvements in Discrete Ordinates Acceleration", ***Transactions of the American Nuclear Society***, 39, 753-755 (1981).
5. Sjoden, G. and A. Haghighat, "A New Adaptive Differencing Strategy in **PENTRANÔ** 3-D Parallel Code," ***Transactions of the American Nuclear Society***, 75, 148, (1996).
6. Sjoden, G. and A. Haghighat, "PENTRAN- Parallel Environment Neutral- particle TRANsport in 3-D Cartesian Geometry," ***Proceedings of Joint International Conference on Mathematical Models and Supercomputing for Nuclear Applications***, Saratoga Springs, New York, (1997)
7. Briggs, W., ***A Multigrid Tutorial,*** Philadelphia, Pennsylvania, Society for Industrial Applied Mathematics, Lancaster Press, 1987.
8. Nowak, P.F., E.W Larsen, and W.R Martin, "A Multigrid Method for S_N Calculations in x-y Geometry", ***Transactions of the American Nuclear Society***, 56,291 (1988)
9. Sjoden, G. and A. Haghighat, "A Simplified Multigrid Acceleration in the PENTRAN 3-D Parallel S_N code," ***Transactions of the American Nuclear Society***, 75, 152, (1996).
10. Alcouffe, R., " A Multigrid Solution of the Three- Dimensional DSA Equation: A Question of Efficiency for Three- Dimensional Transport Equations," ***Proceedings of Advances in Mathematics, Computations, and Reactor Physics,*** Pittsburgh, (1991)
11. Morel, J.E., and T.A. Manteuffel, "An Angular Multigrid Acceleration Technique for S_N Equations with Highly Forward-Peaked Scattering," ***Nuclear Science and Engineering***, 107: 330-342 (1991)
12. Pautz, S. D., J.E. Morel, and M. Adams, "An Angular Multigrid Acceleration Method for SN Equations with Highly Forward-Peaked Scattering," ***Proceedings of Mathematics***

and Computation, Reactor Physics and Environmental Analysis in Nuclear Applications, Madrid, Spain, (1999).

13. Gropp W., Lusk E., and Skjellum A., ***USING MPI- Portable Parallel Programming with Message Passing Interface,*** MIT Press, 1995.
14. Haghighat A., "PENMSH V3.1- A 3-D Cartesian Mesh Generator," Penn State Transport Theory Group, The Pennsylvania State University, 1998
15. Kobayashi, K., "A Proposal for 3-D Radiation Transport Benchmark for Simple Geometries with Void Region," ***OECD Proceedings of 3-D Deterministic Radiation Transport Computer Programs***, 403-413, Paris, France (1996)
16. Haghighat A. and G.E. Sjoden, "Significance of Adaptive Differencing, Variable Grid Density, and TPMC for S_N Methods" ***Proceedings of Mathematics and Computation, Reactor Physics and Environmental Analysis in Nuclear Applications,*** Madrid, Spain, (1999).