Chapter 1
Parallel Iterative and Direct Methods for Transport Calculations on Unstructured Grids *

Shawn P. Burns† Kent G. Budge‡ Mark A. Christon‡

Abstract

Two alternative approaches to the solution of discrete ordinate transport are considered in the context of general unstructured grids. The more traditional approach considered is the solution of a lower triangular system of equations for the spatial intensity distribution. Alternatively, the iterative solution of the system of equations arising from a more general differencing stencil is also considered. Advantages and disadvantages are considered for both sequential implementation and for spatial domain based parallelism.

1 Introduction

Numerical solution of the radiative transport equation is important in a number of applications including fusion energy and combustion. In both cases the radiative transport calculation is coupled with a solid or fluid mechanics but still requires the majority of the computational effort. The need for increased spatial resolution for complex engineering systems also increases the need for an efficient parallel implementation of the radiative transport calculation on an unstructured, spatially decomposed grid.

The principal objective of this work is to compare iterative and direct formulations for solving the radiative transport problem. In this work, the "iterative formulation" refers to the use of a Krylov iterative matrix solution to evaluate a linear system of equations for the radiative intensity for a given direction. The "direct formulation" refers to the use of a single ordered solution sweep to solve a lower triangular system of equations obtained using a one sided differencing stencil.

Although the ultimate interest of this work is the parallel implementation of the direct and iterative formulations, several questions remain regarding the parallel implementation of the direct formulation on general unstructured finite element grids. Therefore, in the following sections, the two formulations will be compared directly in serial for a challenging test case described by Adams [1]. Challenges regarding the parallel implementation of the direct formulation will be considered on a theoretical basis. The iterative formulation to be considered here has been previously applied in parallel by Burns [2] and Burns and Christon [3] and this work will also expand on these earlier efforts.

*This work was performed in part using the ASCI RED parallel supercomputer maintained by the Massively Parallel Computing Research Laboratory at Sandia National Laboratories in Albuquerque, New Mexico. Overall support was provided by the U.S. Department of Energy contract DE-AC04-94AL8500.

†Unsteady and Reactive Fluid Mechanics, Sandia National Laboratories, Albuquerque, NM

‡Computational Physics Research and Development, Sandia National Laboratories, Albuquerque, NM

1.1 Problem Description

To provide a consistent basis for comparison, consider the solution of the Boltzmann transport equation for a single time step and single energy band. The appropriate form of the Boltzmann transport equation for these conditions is given by Adams [1] and may be written in the form

$$\mathbf{\Omega} \cdot \nabla I + \mu_t I = \frac{\mu_t B}{\pi}, \tag{1}$$

where

$$B = \frac{1}{4\mu_t}[(\mu_t - \mu_r)G + 4\pi S], \tag{2}$$

and G is the scalar flux

$$G = \int_{4\pi} I d\Omega. \tag{3}$$

In Equations (1) and (2), μ_t, μ_r, and S are functions of position while the intensity, I, is a function of both position and the direction $\mathbf{\Omega}$. If $\mu_t \neq \mu_r$ then an outer source iteration loop is required to resolve the dependence of the source term B on the intensity.

The angular directions, Ω, in Equations (1) and (2) are discretized using the level weighted, even moment quadrature set derived by Lathrop and Carlson [4]. Spatial discretization is provided by an unstructured finite element grid which may is decomposed across processors in the parallel implementation. The details of the spatial discretization used for Equation (1)are provided in the next section.

2 Complexity Analysis

The following subsections provide a high level description of the iterative and direct formulations considered in this work. The important aspects of each formulation are considered in detail and a theoretical estimate for how the computational work scales is also provided. Finally, the parallel implementation of each formulation is also discussed.

2.1 Iterative Method

The iterative formulation considered in this work is described in detail by Burns [2] and the full development will only be summarized here to conserve space. The iterative formulation is obtained from a Galerkin finite element spatial discretization of Equation (1) with Petrov-Galerkin weighting [5] used for optically thin elements to preserve diagonal dominance of the operator. The resulting linear system of equations for each ordinate direction is then solved using a preconditioned Krylov iterative solution method [6].

An important detail of the formulation descibed by Burns [2] is that the element assembly loop is performed once rather than for each ordinate direction. The element assembly loop constructs the global advection matrix, mass matrix, and load (cf. Equation(1)) prior to the ordinate loop. A rotation operation is performed within the ordinate loop to construct the system of equations for the intensity for each ordinate direction. This rotation operation requires only a small percentage of the time required for the element assembly loop at the expense of the memory required to store the global mass matrix, etc.

In parallel, for a spatially decomposed grid, each processor can construct the portion of the coefficient matrix corresponding to its sub-domain without any interprocessor communication. The work required for this element loop, as well as for the rotation operation described above, scales as the number of elements in the grid, N_e. Interprocessor communication occurs only during the vector inner products and matrix vector products required by the Krylov solver [7] [6].

In general the number of Krylov iterations required to evaluate the intensity for each ordinate direction will increase as N_e^b. It has been shown for a simple three dimensional case that $b = 0.26$ to 0.36 for a number of non symmetric Krylov solvers[1] using diagonal preconditioning [2]. As a result, for a fixed number of elements on each processor, the amount of work per processor will increase as the number of processors increase, e.g., the global problem size increases. This algorithmic inefficiency does not arise from the parallel implementation of the iterative formulation but rather from the nature of Krylov based solvers. Indeed, for reasonable sub-domain size, the interprocessor communication for the iterative formulation accounts for a few percent of the overall execution time [2]. Nevertheless, the inherent non scalability of Krylov methods is the principal drawback to the iterative formulation.

2.2 Direct Method

The Upwind Corner Balance (UCB) formulation described by Adams [1] is used as a typical example of a one sided differencing stencil for this work. As for the iterative formulation, the development of the direct formulation will only be summarized here to conserve space. Some details of the implementation of the UCB stencil peculiar to the present work will be emphasized.

The UCB algorithm evaluates the intensity at each corner of each element in the grid. Therefore, the UCB algorithm will involve N_v times as many unknowns as the iterative formulation for the same grid where N_v is the number of vertices per element, e.g., $N_v = 4$ for quadrilateral elements and $N_v = 8$ for hexahedral elements. The intensity at a given element corner depends only on the intensities which lie "upwind" of the corner relative to the ordinate direction. Therefore the intensities at each element corner may be evaluated by sweeping through the mesh along the ordinate direction without explicitly forming a system of equations and resorting to a Krylov solver.

To calculate the corner intensities for a given element a number of element areas and volumes need to be evaluated [1]. As the UCB algorithm is implemented here, these geometric quantities are evaluated "on the fly" rather than being precalculated and stored. Precalculating these quantities might accelerate the direct formulation but this approach would require the storage of 16 floating point values for each quadrilateral and 44 floating point values for each hexahedral element. It will be shown in the following sections that the calculation of the element geometry requires approximately 10% of the computation time for a two dimensional grid while gathering element data from the global storage takes as much as 40% of the compute time. Therefore, precalculating and storing the element geometry may not be cost effective.

To order the elements for each solution sweep it is necessary to evaluate the element-to-element connectivity graph for each ordinate. It is possible for cycles to occur in this connectivity graph for general unstructured finite element grids. This prevents the determination of unique upwind dependencies necessary for the solution sweep for an ordinate direction. Therefore, it is necessary to break these cycles when they occur, for example, by using intensities from a previous source iteration. In the current implementation, cycles are not considered and only cycle free grids are employed.

Two principal complications arise in the parallel implementation of the direct formulation using spatial domain based parallelism. The first of these is the inherently sequential

[1] Gauss-Seidel, stabilized biconjugate gradient, conjugate gradient squared

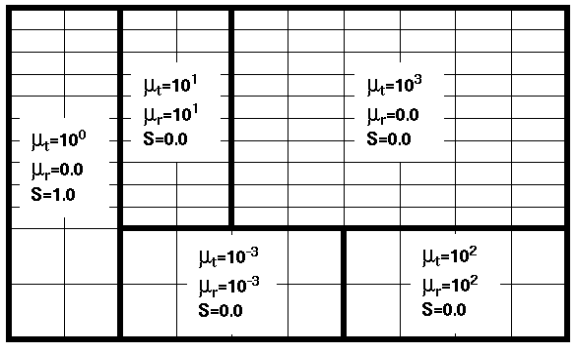


FIG. 1. *Coarse grid for two dimensional test problem. Five regions with four node quadrilateral elements throughout.*

nature of the solution sweep. Since information propagates through the mesh along the ordinate direction, downstream processors cannot do useful work until the sweep propagates through all upstream processor sub-domains. The second complication is the identification of cycles in a spatially decomposed element connectivity graph.

To alleviate the first of these complications, at least partially, the UCB algorithm implemented in this work is designed to sweep in all ordinate directions simultaneously. Thus all processors whose spatial sub-domain includes a portion of the global domain boundary will be able to perform useful work from the start while only the interior processors must wait for the sweeps to propagate through the grid.

3 Two Dimensional Serial Test Case

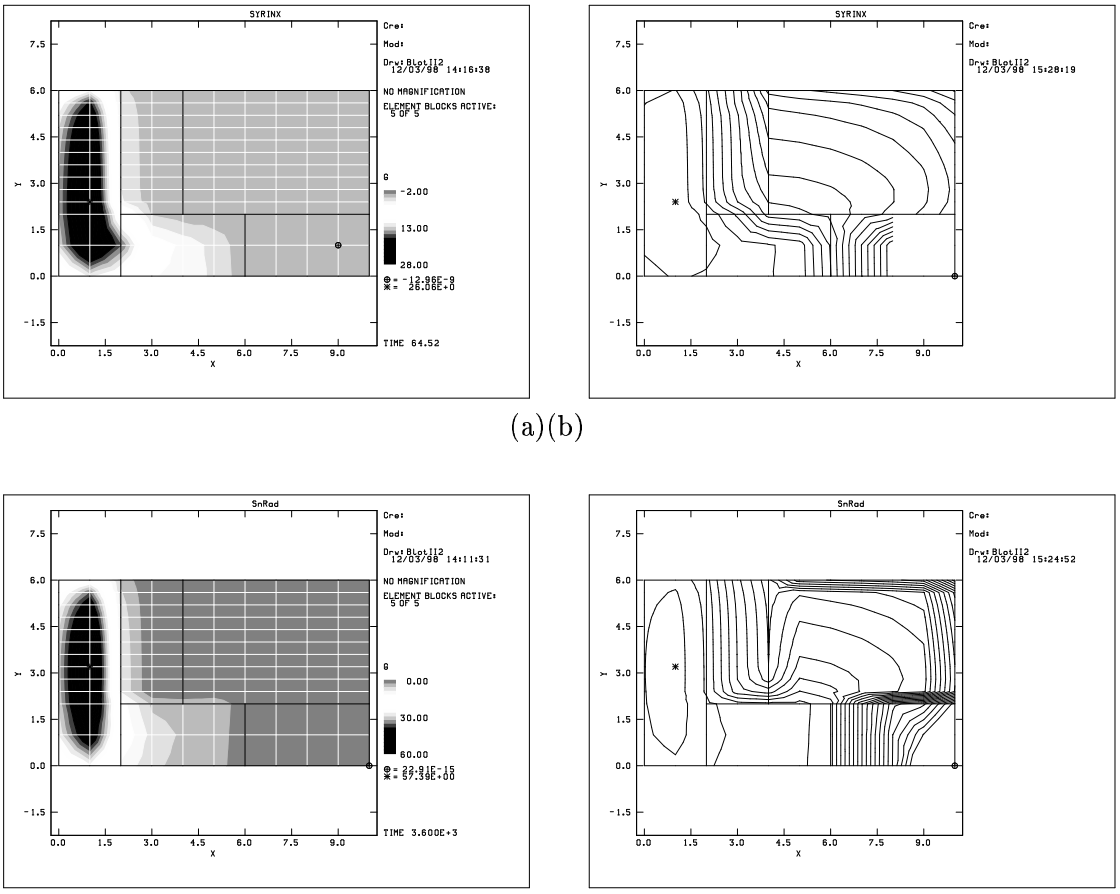
In this section results are presented for a challenging two dimensional test case described by Adams [1]. The test problem geometry is summarized in Figure 1. The geometry consists of a two dimensional domain with five regions. The left most region has a positive generation rate with a moderate optical thickness and isotropic scattering, i.e., $\mu_r \neq \mu_t$. The large region in the upper right corner is very optically thick and also scatters radiation isotropically. A streaming region connects the generating region and the large thick region resulting in a step change in absorptivity of six orders of magnitude at the exit of the streaming region. The remaining regions are strongly absorbing non scattering regions. Adams proposed the coarse grid shown to increase the difficulty of the test problem rather than to provide a mesh independent solution.

A simple minded source iteration is employed here in which Equations (1) and (2) are solved iteratively until the maximum relative change in the nodal scalar flux satisfies the criteria

$$\max_{1 \leq i \leq N_n} \left[\frac{|G_i^n - G_i^{n-1}|}{\epsilon_1 G_i^n + \epsilon_2} \right] < 1, \tag{4}$$

where the superscript n is the source iteration counter, i is the node number, and N_n is the number of nodes in the mesh. No source iteration acceleration schemes are used for either the iterative or the direct formulation. The UCB corner values for the scalar flux were mapped onto the nodes using a volume average for the purposes of evaluating the convergence but the unmapped corner values were used in calculating the right hand side of Equation (1). For the convergence results discussed below $\epsilon_1 = 10^{-6}$ and $\epsilon_2 = 10^{-10}$.

Figure 2 shows the results obtained from both the iterative and direct formulation for



(a)(b)

(c)(d)

FIG. 2. *Coarse grid solution for two dimensional test problem - (a) iterative solution for scalar flux, (b) iterative solution for the log of the scalar flux, [-9.0, 2.0]×0.5 (c) direct solution for the scalar flux, (d) direct solution for the log of the scalar flux [-9.0, 2.0]×0.5*

the coarse grid shown in Figure 1. Figure 2 shows both the scalar flux values, G, and the log of the scalar flux predicted by both formulations. Both of the formulations predict the same qualitative result although the direct formulation predicts a maximum scalar flux more than twice that of the iterative formulation. This is to be expected given the differences between the two formulations and the extremely coarse grid used. The iterative formulation also allows the solution to go very slightly negative ($O(-10^{-9})$)in the lower right hand corner of the grid whereas the UCB direct formulation remains positive. Both formulations predict the minimum scalar flux in the strongly absorbing lower right hand region of the grid.

The iterative formulation results shown in Figure 2a and 2b were obtained using the row sum lumped form of the mass matrix resulting from the absorption term in Equation (1). While the source iteration converged using the consistent mass matrix, the resulting scalar flux solution demonstrated severe spatial oscillations in the optically thick region. The lumped mass matrix, however, provided a smooth and reasonable solution as shown in Figure (2).

As expected, the compute time per source iteration for the direct formulation scales as

the number of elements in the mesh while the compute time for the iterative formulation increases faster than the number of elements. Both formulations require approximately the same time per source iteration for $N_e = 120$ elements, i.e., the grid shown in Figure 1, but the iterative formulation requires approximately 13 times as long as the direct formulation for $N_e = 30,720$. It should not be assumed, however, that both formulations require the same number of source iterations. A detailed study of source iteration convergence and acceleration techniques, however, is beyond the scope of this work.

Approximately 40% of the compute time per source iteration for the direct formulation is required to gather the element data, e.g., vertex locations, intensities, etc., from the global storage and to scatter the updated element corner intensities back to global storage. Calculating the element geometry, e.g., interface areas and corner volumes, and updating the element corner intensities requires approximately 10% and 20% of the compute time respectively for the direct formulation. These fractions are independent of the number of elements in the global grid.

As described in the complexity analysis, the amount of time taken by the Krylov matrix solution for the iterative formulation increases as the number of elements increases. At $N_e = 120$ the Krylov matrix solve requires approximately 50% of the compute time per source iteration for the iterative formulation. This fraction increases to approximately 98% of the compute time for $N_e = 30,720$. The rate at which the compute time for the Krylov matrix solution increases is a function of the solver type and matrix preconditioning.

3.1 Effect of Krylov Solver and Preconditioning

The rate at which the number of Krylov iterations increased with the number of elements for the two dimensional test problem was determined by successively doubling the grid shown in Figure 1. Four successive grid doublings were employed to obtain the results described below. Similar data is also shown by Burns [2] using diagonal preconditioning for a three dimensional problem with spatial domain decomposition.

Table 1 summarizes the iteration scaling for three solvers - generalized minimal residual (GMRES), stabilized biconjugate gradient (BiCGStab), and conjugate gradient squared (CGS) - using diagonal, Neumann, and least squares preconditioning [2]. Assuming that the number of iterations scales as $N_{it} = CN_e^b$, Table 1 gives values for C and b for each solver/preconditioner combination. All of the data shown in Table 1 were obtained using the AZTEC Krylov matrix solver library developed at Sandia National Laboratories in Albuquerque, New Mexico.

The data in Table 1 shows that, of the three solvers considered , the BiCGStab solver requires the fewest number of iterations for a given preconditioner. BiCGStab also provides the slowest increase in the number of iterations, i.e., lowest value of b, for a given preconditioner. Per iteration, the GMRES algorithm is the most time consuming of the three solvers considered. The CGS and BiCGStab algorithms have equivalent compute times per iteration. Therefore, the BiCGStab algorithm provides the best performance in terms of the fastest compute time and slowest increase in the number of iterations with grid size. A similar result was obtained by Burns [2].

The data in Table 1 also shows that preconditioning has a beneficial effect on the BiCGStab algorithm in terms of the iteration count. The additional cost of the Neumann and least squares preconditioners, however, tend to negate this benefit and

[2]Using a Krylov subspace size of 30 for GMRES and a polynomial order of 2 for Neumann and least squares preconditioning

solver	preconditioner		
	none	Neumann	least squares
GMRES	C= 1.40 b= 0.48	C= 0.54 b= 0.48	C= 0.69 b= 0.46
BiCGStab	C= 1.04 b= 0.45	C= 0.49 b= 0.42	C= 0.68 b= 0.41
CGS	C= 1.37 b= 0.45	C= 0.52 b= 0.43	C= 0.62 b= 0.43

TABLE 1
Iteration scaling with grid size. $N_{it} = CN_e^b$

the overall compute time per Krylov iteration remains relatively unchanged regardless which preconditioner is used for the problem sizes considered. Neumann preconditioning provides a slightly faster compute time than diagonal and least squares preconditioning for $N_e > 30,720$.

A reasonable preconditioning approach may be to combine the iterative and direct formulations by using an ordered mesh sweep as a preconditioning step for the Krylov iterative solution. The next section describes a simple implementation of such an approach for a three dimensional problem running in parallel.

4 Parallel Upwind Preconditioning

The preconditioning formulations employed in the previous section did not take into account the hyperbolic nature of Equation (1) in the optically thin limit. In this section, a simple upwind preconditioning formulation is used to explore the possible benefits of a more intelligent preconditioning scheme on the performance of the iterative formulation. The preconditioning step used here employs a single ordered Gauss-Seidel sweep through the system of equations. The initial guess for the Gauss-Seidel sweep is the ratio of the load vector and the main diagonal of the coefficient matrix so that diagonal preconditioning is regained if the Gauss-Seidel preconditioning sweep is not employed.

The equation ordering used in the preconditioning sweep is determined for each ordinate direction based on the inner product of the nodal position corresponding to each equation and the ordinate direction. The index vector which sorts this list of inner products from minimum to maximum also represents a crude ordering of the nodes from upstream to downstream for a given ordinate direction. The hope is that this simple preconditioner will accelerate the rate at which information propagates through the grid in the ordinate direction and reduce the rate at which the number of Krylov iterations increases with the global grid size.

The three dimensional test problem described by Hsu and Farmer [8] and solved in parallel by Burns and Christon [2][3] is used to demonstrate the performance of the simple upwind preconditioner. This problem consists of a three dimensional cube with constant temperature and no scattering, e.g., $\mu_r = \mu_t$. The absorption coefficient varies trilinearly from a minimum of $\mu_t = 0.1$ at the edges of the cube to $\mu_t = 1.0$ at the center of the cube and only a single energy band is considered.

For the results presented here, the cube is discretized using a uniform, orthogonal grid of $N \times N \times N$ eight node hexahedral elements. The grid is decomposed into $M \times M \times M$ non

overlapping spatial sub-domains each of which are assigned to one of $P = M^3$ processors. All of the data presented below was obtained using the ASCI RED distributed memory, Multiple Instruction/Multiple Data, message passing super computer supported by the Massively Parallel Computing Research Laboratory at Sandia National Laboratories in Albuquerque, New Mexico. Although a structured grid is employed to provide these results, the formulation used for the iterative formulation described previously assumes a completely unstructured finite element grid.

Figure 3 compares the performance of the diagonal and upwind preconditioner using a spatial sub-domain size of 8,000 elements. Figure 3 shows that the simple upwind preconditioner described above has a distinct impact on the growth of the number of BiCGStab iterations compared to diagonal preconditioning. A curve fit of the data in Figure 3a shows that the number of BiCGStab iterations grows as $N_{it} = 0.66N_e^{0.33}$ for diagonal preconditioning and $N_{it} = 1.34N_e^{0.23}$ for upwind preconditioning. As a result, the scaled parallel efficiency [3] at $P = 125$ processors ($N_e = 10^6$) rises from 29% for diagonal preconditioning to 51% for upwind preconditioning.

The most important performance metric, however, is the total solution time which is expressed as the total wall clock time for solution per ordinate per element in Figure 3b. Although the parallel performance is improved, the total grind time for the upwind preconditioner does not catch up to that of diagonal preconditioning for this test problem until the number of elements in the global grid exceeds 500,000.

Although this simple upwind preconditioner reduced the number of BiCGStab iterations for this three dimensional test problem, this performance may degrade for more challenging applications. Testing of the upwind preconditioner for the two dimensional test problem described previously showed that this preconditioning schemes could cause the solver to diverge as the grid refinement increases. Divergence may be avoided by employing more than one ordered Gauss-Seidel sweep during the preconditioning sweep. It is not possible to investigate this breakdown here but it is possible that the large diffusion region in the two dimensional test problem may be a contributing factor.

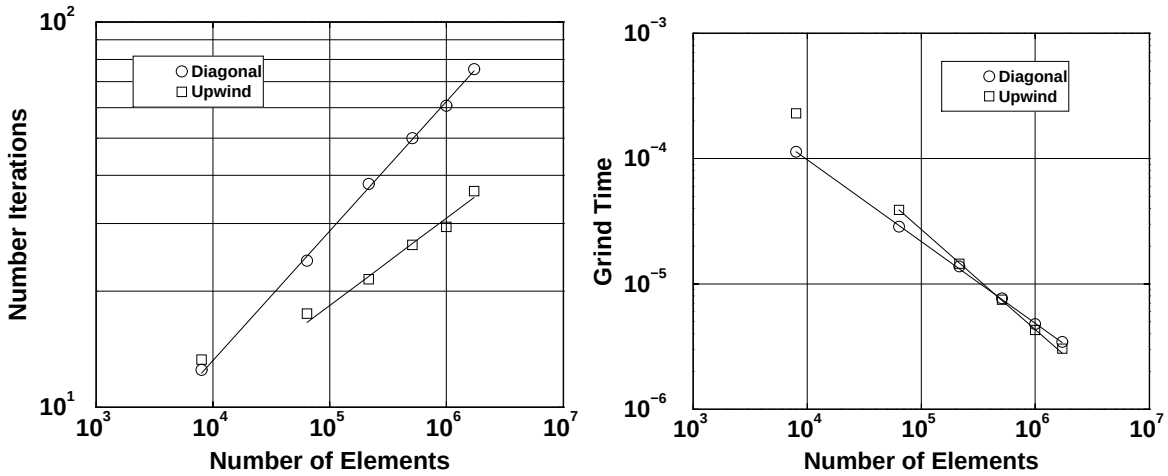
Nevertheless, the ability of such a simple minded preconditioning scheme to improve the parallel performance of the iterative formulation is promising. Poor cache performance caused by the Gauss-Seidel sweep ordering may account for a significant fraction of the preconditioner overhead and reordering the coefficient matrix prior to solution my be beneficial. Additional research along these lines is certainly warranted

5 Observations

This work compared iterative and direct solution formulations for the radiative transport formulation. The iterative formulation employs a Krylov iterative solver to evaluate a system of equations resulting from a general spatial discretization of the Boltzmann radiative transport equation for the directional radiative intensity. The direct formulation employs a one sided spatial differencing scheme and an ordered solution sweep through the grid to solve the resulting lower triangular system of equations.

Parallel implementation of the iterative formulation is straight forward and only requires interprocessor communication to complete the matrix-vector and vector-vector products required at each iteration of the Krylov solver. The principal drawback of this formulation is that the number of Krylov iterations required for convergence grows as the number of elements in the global mesh resulting in a degradation in the scaled parallel performance.

In contrast, the direct formulation requires only a single sweep through the grid to



(a) (b)

FIG. 3. *Upwind versus diagonal preconditioning for three dimensional parallel test problem using the stabilized biconjugate gradient solver and non-overlapping spatial sub-domains of 8,000 elements. Comparison of (a) average number of BiCGStab iterations per ordinate direction and (b) grind time per element per ordinate direction as a function of the number of elements in the global mesh*

obtain the directional intensity for any grid size. Parallel implementation of the direct formulation is more problematic given the inherently sequential nature of the solution sweep through a spatially decomposed grid. Also, the direct formulation requires the specification of an element connectivity graph for each ordinate direction. Cycles may occur in the connectivity graph for unstructured grids and must be identified for the solution sweep to proceed. Identifying cycles in a spatially distributed graph poses an additional challenge for the direct formulation.

Several general observations may be made from the data presented in this work:

The general spatial discretization used for the iterative formulation provides a reasonable solution when compared to the upwind corner balance formulation for a challenging two dimensional problem with sharp changes in absorptivity and a coarse spatial grid.

A stabilized biconjugate gradient solver with diagonal preconditioning provided the best performance of the solvers and preconditioners considered in terms of the overall run time and increase in the number of iterations with grid size.

A curve fit of the data showed that the average number of iterations per ordinate direction for the BiCGStab solver increased as $N_{it} = 0.49N_e^{0.42}$ for the two dimensional problem and $N_{it} = 0.66N_e^{0.33}$ for a less challenging three dimensional problem.

A simple minded upwind preconditioning formulation showed that a preconditioning method which takes advantage of the hyperbolic nature of the Boltzmann radiative transport equation may significantly improve the performance of the iterative formulation. Additional development is required to provide a robust preconditioning method.

References

[1] M. L. Adams, *Subcell balance formulations for radiative transfer on arbitrary grids*, Transport Theory and Statistical Physics, 26 (1997), pp. 385-431.

[2] S. P.Burns, *Application of Spatial and Angular Domain Based Parallelism to a Discrete Ordinates Formulation with Unstructured Spatial Discretization*, Proceedings of the Second International Symposium on Radiation Transfer, (1997), Kusadasi, Turkey, pp. 173-193.

[3] S. P.Burns, M. A.Christon, *Spatial Domain-Based Parallelism in Large Scale, Participating-Media, Radiative Transport Applications*, Numerical Heat Transfer, Part B: Fundamentals, 31 (1997)

[4] K. D.Lathrop, B. G.Carlson, *Discrete Ordinates Angular Quadrature of the Neutron Transport Equation*, Los Alamos Scientific Laboratory Report, LA-3186, Los Alamos, New Mexico, (1965).

[5] O. C.Zienkiewicz, R. L.Taylor, *The Finite Element Method, Volume 2, Solid and Fluid Mechanics Dynamics and Non-Linearity, Fourth Edition*, McGraw-Hill Book Company, London, England, (1991).

[6] S. A.Hutchinson, L. V.Prevost, J. N.Shadid, R. S.Tuminaro, *Aztec User's Guide, Version 2.0 Beta*, Sandia National Laboratories Report SAND95-1559, (1998), Albuquerque, NM, ($http://www.cs.sandia.gov/CRF/Aztec_pubs.html$)

[7] R.Barrett, M.Berry, T.Chan, J.Demmel, J.Donato, J.Dongarra, V.Eijkhout, R.Pozo, C.Romine, H.van der Vorst, em/Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, Society for Industrial and Applied Mathematics, (1994), ($http://netlib2.cs.utk.edu/linalg/html_templates/report.html$)

[8] P.Hsu, J.T.Farmer, *Benchmark Solutions of Radiative Heat Transfer Within Nonhomogeneous Participating Media Using the Monte Carlo and YIX Methods* Proceedings of the 30th National Heat Transfer Conference, Portland, OR, ASME HTD-vol. 315, (1995), pp. 29-36