Preprint
UCRL-JC-148857

A Discrete Ordinates Algorithm for Radiation Transport Using Block-Structured Adaptive Mesh Refinement

L. H. Howell

This article was submitted to the Nuclear Explosives Code Developers' Conference, Monterey, CA, October 21–24, 2002

February 3, 2003

Proceedings of the NECDC 2002

A Discrete Ordinates Algorithm for Radiation Transport Using Block-Structured Adaptive Mesh Refinement (U)

L. H. Howell*

*Lawrence Livermore National Laboratory

The discrete ordinates method is well-suited to implementation with block-structured adaptive mesh refinement (AMR). AMR meshes group points into logically-rectangular patches, and provide the benefits of adaptivity without sacrificing the efficiency and geometric regularity of regular grids. In particular, these meshes preserve the directional ordering of points required for explicit ordinate transport sweeps. A number of algorithmic issues must be addressed to make such a method practical. These include sequencing of ordinates and grids for parallel execution, simultaneous solution of the transport equation on multiple levels of the grid hierarchy, implicit coupling to the fluid energy, and conservation of energy in a time-dependent context where different grid levels advance with different timesteps. I discuss these and other issues and present example calculations in two and three spatial dimensions. (U)

Introduction

In previous work *(Howell and Greenough, 1999 and 2003)* I have described the addition of gray radiation diffusion to an algorithm for multifluid Eulerian gasdynamics with adaptive mesh refinement. The scheme has its roots in the block-structured AMR algorithms developed for hyperbolic systems by Berger and Oliger (1984) and extended by Berger and Colella (1989) and by Bell et al. (1994). The implicit treatment of the parabolic diffusion equation is based on algorithms developed for the incompressible Navier-Stokes equations in *(Howell and Bell, 1997)* and *(Almgren et al., 1998).* Other examples of implicit processes coupled to explicit advection in AMR codes include self-gravitation in astrophysics *(Truelove et al., 1997),* and discrete ordinates for radiation in combustion applications *(Howell et al., 1999).*

The present work replaces the radiation diffusion equation with a discrete ordinates approximation to gray radiative transport. I will briefly present the implicit coupling between the radiation field and the fluid energy, showing that this introduces an effect resembling isotropic scattering, even when true scattering is absent. Having reduced the time-dependent problem to solution of the steady-state discretized transport equations, I will discuss the solution of these equations by parallel transport sweeps both on single refinement levels and on multiple levels of the adaptive mesh hierarchy. The final section

concerns convergence acceleration by conjugate gradient and a discussion of preconditioners.

AMR Essentials

A complete description of the AMR timestepping scheme can be found in many of the references cited in the previous section, so for this paper I will give only a very brief overview. The computational mesh consists of a hierarchy of refinement levels, each of which is locally uniform and organized as a union of rectangular patches. Cells on different levels are aligned, and refinement ratios between levels are typically 2 or 4. Edges of different levels may not be coincident except at the physical boundary—that is, level $\ell + 2$ may not directly border level ℓ without an intervening region at refinement level $\ell + 1$. For computational convenience coarse cells are defined under finer grids, but the finest solution in a given region takes precedence over coarser versions.

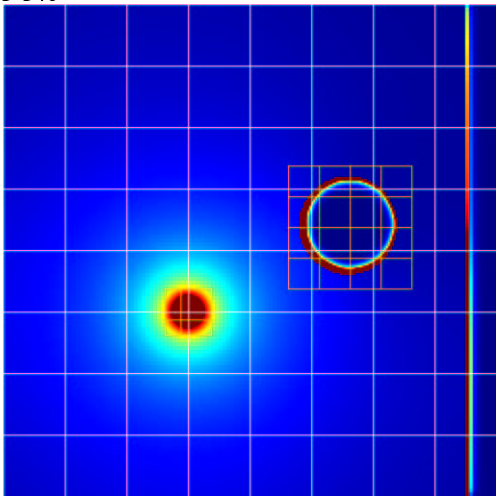
The scheme typically refines in time as well as space, both for efficiency and to help maintain an optimal Courant condition on all levels for the explicit hydrodynamics. The refinement ratios in time are the same as those in space—these are specified by the user along with the specific criteria for refinement. Timestepping is a recursive process, where a level ℓ is advanced first without reference to any finer levels, then level $\ell + 1$ is advanced a number of times to catch up (this in turn requires advancing levels $\ell + 2$ and higher). After the finer levels have reached the time corresponding to the end of the coarse timestep, synchronization operations must be performed to correct any flux mismatches that have occurred at the coarse-fine interface. Effects—including radiation—which are implicitly timestepped also require use of an implicit solver in the synchronization process.

Library support for this adaptive computational model, including parallel support through use of the grid layout on each refinement level for spatial domain decomposition, is provided by the BoxLib class library *(Rendleman et al., 2000).*

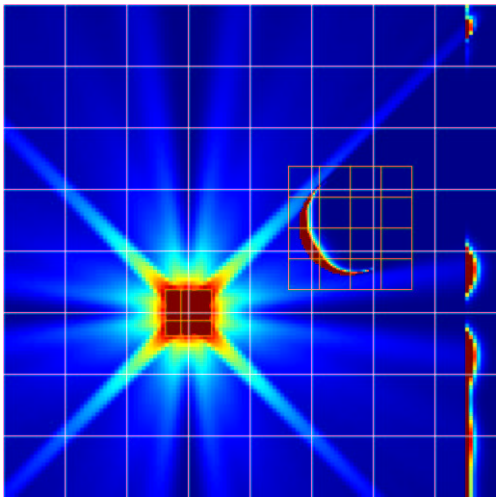
Fig. 1 shows the results of a simple time-dependent radiation calculation using AMR, and also illustrates the differences between diffusion and transport. This is a pure absorption problem set in the unit square and featuring two different fluids: an optically thin ($\kappa = 1$) background and an optically thick ($\kappa = 100$) material localized in a circular absorbing region and in a layer along the right wall. A hot spot in the thin material radiates energy, which is then absorbed weakly by the thin material and strongly by the thick material. Finer grids are placed around the hot spot and the circular absorbing region. Fluid energy is shown after 10 coarse (and 20 fine) timesteps. All three calculations conserve energy at the coarse-fine interfaces by using multilevel synchronization at the end of each coarse timestep, but nonreflecting boundary conditions allow energy to exit via the edges of the domain. The diffusion calculation uses the flux limiter presented in *(Levermore and Pomraning, 1981).* The transport calculations use the step characteristic discretization *(Mathews, 1999).*

The diffusion calculation produces a smooth solution that clearly shows one of the key weaknesses of diffusion in an optically thin medium. Energy is absorbed on the back side

Flux-limited Diffusion



S_{16} (144 ordinates)



144 equally-spaced ordinates

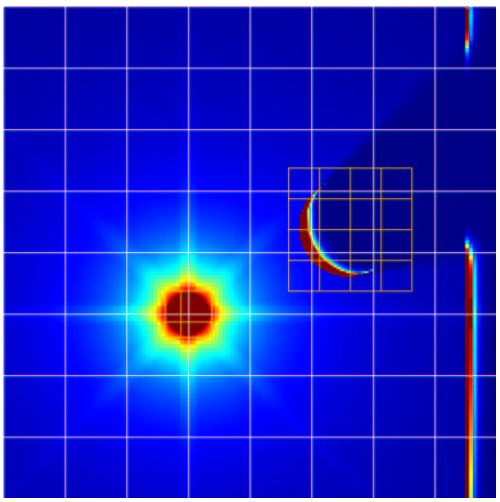


Fig. 1. Diffusion contrasted with discrete ordinates, with mesh refinement. Fluid energy is shown (overexposed to bring out detail).

of the circular cloud as well as the front, and no shadow is cast on the right wall. (Without the limiter the result is even worse—energy is spread through the thin region in a much more uniform manner (not shown in figure). The limiter at least allows for a more physically-reasonable decline in intensity with distance from the source.)

The transport solutions differ only in the choice of ordinate set (angular discretization), and both use 144 ordinates in 2D. The standard S_{16} set distributes ordinates in a roughly uniform manner throughout a hemisphere. Effort expended in discretizing the vertical direction (out of the plane) results in relatively poor resolution in the 2D plane of the calculation. As a result, ray effects are very obtrusive, causing false "shadows" as noticeable as the true shadow cast by the circular cloud. The final picture shows the results with a different ordinate set suggested to me by Britton Chang. All ordinates have the same direction cosine (0.577350269) with respect to the vertical axis, and are arranged uniformly with respect to the plane of the calculation. The first three moments of this distribution are correct, ray effects are minimized, and a clear shadow is cast by the circular cloud.

Equations for Radiation and Fluid Energy

Timestepping uses a split formulation where advection of all fluid state components takes place first, followed by conduction, and finally by the radiation transport step. The first two of these operations have been described in previous work, so I will summarize them as follows without further discussion:

$$(\rho E)^{-} = (\rho E)^{n} + \Delta t \left\{ -\left[\nabla \cdot (\boldsymbol{u}\rho E + \boldsymbol{u}p)\right]^{n+1/2} + \frac{1}{2}\nabla \cdot \left(\kappa_0 T^{5/2} \nabla T\right)^{n+1/2} \right\}. \qquad [1]$$

For the transport step I use the time-dependent, gray radiation transport equation with a fully-implicit time discretization. Though scattering may or may not be physically present, I include isotropic scattering in the equation to show the relationship between "true" scattering and a scattering-like term that arises from implicit coupling of radiation with the fluid energy (next section).

$$\frac{I^{n+1} - I^{n}}{c\Delta t} + (\Omega \cdot \nabla) I^{n+1} + \left(\kappa_a^{n+1} + \kappa_s^{n+1}\right) I^{n+1} = \kappa_a^{n+1} B^{n+1} + \kappa_s^{n+1} \phi^{n+1}, \qquad [2]$$

$$\phi^{n+1} = \frac{1}{4\pi} \int_{4\pi} I^{n+1} d\Omega, \qquad [3]$$

$$(\rho E)^{n+1} = (\rho E)^{-} - \Delta t \cdot 4\pi \kappa_a^{n+1} \left(B^{n+1} - \phi^{n+1}\right). \qquad [4]$$

In the actual code a multifluid formulation is used, with volume, mass, and energy fractions stored separately for each material present in a cell. Quantities derived from the fluid state, such as emission and absorption coefficients, are likewise computed separately for each material, but for radiation quantities only a single value is used for each cell. There is thus a single radiation equation, but a separate fluid energy equation for each material, as follows:

$$\frac{I^{n+1} - I^{n}}{c\Delta t} + (\Omega \cdot \nabla) I^{n+1} = \sum_{\alpha} f^{\alpha} \kappa_a^{\alpha,n+1} \left(B^{\alpha,n+1} - I^{n+1}\right) + \kappa_s^{n+1} (\phi^{n+1} - I^{n+1}), \qquad [5]$$

$$(f^\alpha \rho^\alpha E^\alpha)^{n+1} = (f^\alpha \rho^\alpha E^\alpha)^- - \Delta t \cdot 4\pi f^\alpha \kappa_a^{\alpha,n+1} \left(B^{\alpha,n+1} - \phi^{n+1} \right). \quad [6]$$

Quantities with α superscript are computed separately for each fluid. This form should be assumed throughout the remainder of the paper, though I will not need to write it out explicitly again.

Returning to the single-fluid form, the discrete ordinate method is a discretization of the angular dependence of I over a number of discrete directions (ordinates) Ω_m. Equations (2) and (3) thus become

$$\frac{I_m^{n+1} - I_m^n}{c\Delta t} + (\Omega_m \cdot \nabla) I_m^{n+1} + \left(\kappa_a^{n+1} + \kappa_s^{n+1} \right) I_m^{n+1} = \kappa_a^{n+1} B^{n+1} + \kappa_s^{n+1} \phi^{n+1}, \quad [7]$$

$$\phi^{n+1} = \frac{1}{4\pi} \sum_m w_m I_m^{n+1}. \quad [8]$$

An ordinate set is specified by the choices of ordinates Ω_m and weights w_m. Many different choices are possible, two of which are contrasted in the numerical example in the previous section. In Cartesian coordinates each I_m couples to other ordinate directions only through the boundary conditions and the scattering source. (More complex coupling appears in other coordinate systems. For a general introduction to this and other discrete ordinate concepts, see *(Lewis and Miller, 1993)).*

Solvers for (Eq. 7 and 8) are built using transport sweeps. For each ordinate m, computation begins at the "upstream" edges of the domain and proceeds "downstream" in the direction Ω_m. Various spatial discretizations are possible. The current code supports the basic step and diamond difference iterations, along with simple and upstream corner balance methods (SCB and UCB) *(Adams, 1997),* and step characteristic (SC) *(Mathews, 1999).* Given correct upstream boundary information and scattering source ϕ, values for I_m are obtained in a single sweep across the domain. The simplest solution algorithm, source iteration, consists of repeated transport sweeps with the sources recomputed after each sweep. Acceleration techniques can be applied to improve the convergence rate of this iteration, as I discuss later in the paper.

AMR Time Step

Finer levels of the adaptive mesh hierarchy are advanced using smaller timesteps than coarser levels. The timestepping algorithm used for the current code is very similar to those presented for discrete ordinates in *(Howell et al., 1999),* and for radiation diffusion in *(Howell and Greenough, 1999)* and *(Howell and Greenough, 2003).* Because of these similarities, I will not present the details of the scheme here but will instead give only a brief summary.

The algorithm is fundamentally recursive. A level first advances itself without reference to finer levels, then has the next finer level advance—using interpolated boundary conditions as needed—with finer timesteps, then performs synchronization operations to ensure conservation for fluxes crossing the interface to the next finer level, and finally returns control to the next coarser level.

The first of these steps, the single-level advance, is further broken down through the splitting procedure outlined in the previous section. Advection and conduction take place first, followed by radiation. The radiation step involves the solution of the discrete ordinate discretization of the transport equation (Eq. 7 and 8), coupled to the fluid energy through (Eq. 4). Close nonlinear coupling between the radiation and the fluid when κ_a is large requires an iterative, implicit solution technique.

As in the diffusion algorithm presented in *(Howell and Greenough, 2003),* the derivation starts by extrapolating the emission term to the new time ($n+1$) temperature. (A star (*) superscript designates a previous approximation to time $n+1$ quantities.)

$$\kappa_a^{n+1} B^{n+1} = \kappa_a^* B^* + \frac{1}{c_v}(e^{n+1} - e^*)\frac{\partial(\kappa_a B)}{\partial T}. \qquad [9]$$

Substitution of this linearization into the radiation and fluid energy equations then yields the following form for the update iteration:

$$\frac{I^{n+1} - I^n}{c\Delta t} + (\Omega \cdot \nabla) I^{n+1} + (\kappa_a^* + \kappa_s^*)\, I^{n+1} =$$

$$(\kappa_s^* + \eta^* \kappa_a^*)\phi^{n+1} + (1-\eta^*)\kappa_a^* B^* - \frac{\eta^*}{4\pi\Delta t}\left((\rho E)^* - (\rho E)^-\right), \quad [10]$$

$$(\rho E)^{n+1} = \eta^*(\rho E)^* + (1-\eta^*)\left\{(\rho E)^- - \Delta t \cdot 4\pi\kappa_a^*\left(B^* - \phi^{n+1}\right)\right\}, \qquad [11]$$

where

$$\eta^* = \frac{\dfrac{\partial(\kappa_a B)^*}{\partial T}}{\dfrac{\rho^{n+1} c_v}{4\pi\Delta t} + \dfrac{\partial(\kappa_a B)^*}{\partial T}}. \qquad [12]$$

The key modification to the radiation equation (Eq. 10) is the the addition of a term that behaves like isotropic scattering, even when no true scattering is present. The other changes only alter coefficients. It is convenient then to gather terms of (Eq. 10) as follows, in order to obtain a simplified form for the derivation of acceleration schemes and the design of solver software.

$$(\Omega_m \cdot \nabla) I_m + \sigma_t I_m = \frac{1}{4\pi}\sigma_s \sum_{m'} w_{m'} I_{m'} + S_m, \qquad [13]$$

where

$$\sigma_t = \kappa_a^* + \kappa_s^* + \frac{1}{c\Delta t} \qquad [14]$$

$$\sigma_s = \kappa_s^* + \eta^* \kappa_a^* \qquad [15]$$

$$S_m = \frac{1}{c\Delta t} I_m^n + (1-\eta^*)\kappa_a^* B^* - \frac{\eta^*}{4\pi\Delta t}\left((\rho E)^* - (\rho E)^-\right). \qquad [16]$$

While the level advance operations require no communication with other levels beyond the use of interpolated boundary conditions, the synchronization operations match fluxes passing across coarse-fine interfaces in both directions and therefore require the participation of at least two levels at once. The solver I use for this is based on the one presented in *(Jessee et al., 1998)*, with two significant differences. First, in the current version the interface sources are explicitly located at the appropriate cell edges rather than at the centers of coarse cells adjacent to the interface. Second, the current version is implemented in parallel, with communication operations built up from BoxLib primitives. (Parallel communication between levels is necessary only at the coarse-fine interface, while parallel communication within each individual level is discussed in more detail in the next section.)

The form of the AMR timestep itself—which determines the source function for the multilevel synchronization operations—most closely resembles that given in *(Howell and Greenough, 1999)* for radiation diffusion. The version of the timestep algorithm presented for discrete ordinates in *(Howell et al., 1999)*, and for radiation diffusion in *(Howell and Greenough, 2003)* is more complicated, including the option of an additional multilevel solve at the beginning of each coarse timestep. This extra solve provides additional accuracy in cases where essential features of the global solution are not adequately reproduced on coarse level grids. I have not yet added this option to the current discrete ordinates code, but all of the necessary software capabilities are available to do so if it should be needed.

Parallel Transport Sweeps

The AMR timestep algorithm requires solutions to the transport equation both on single levels and on multiple levels at a time. The former can be solved by source iteration—that is, by repeated transport sweeps. The latter, as mentioned in the previous section, can be solved by a more complicated iteration that also breaks down into single-level transport sweeps *(Jessee et al., 1998)*.

In problems with significant scattering the source iteration may converge too slowly to be practical, and so in the next section I discuss acceleration techniques. Even with acceleration, though, transport sweeps remain the fundamental unit of the calculation, and the only part that is not directly parallelized by the BoxLib software framework. It is therefore appropriate to specify the parallelization techniques used, both for single and for multiple level problems.

Parallelization in BoxLib *(Rendleman et al., 2000)* is through spatial distribution of each level across the entire processor set. Grids are distributed whole—there is no provision for two or more processors to operate on the same grid. The grid generation parameters must therefore be chosen to ensure that there are enough grids on each level to keep all the processors busy. (Reasonable load balancing is obtained with three or more grids per processor.) This criterion favors large numbers of small grids, as does the issue of cache efficiency. On the other hand, the overheads associated with the adaptive grid layout, coarse-fine interfaces, and interprocessor communication favor smaller numbers of

large grids. I mention this here to point out that there is a tradeoff, but I will not explore the issue further in this paper. For this section it is sufficient to remember that each grid, its neighbors, and the coarser and finer grids it overlaps, may all be on different processors, and it is not too misleading to use the word "grid" as a euphemism for "processor".

The reason parallelization is not adequately provided by standard BoxLib primitives is that transport sweeps proceed across the domain following the direction of each ordinate. Grids farther "downstream" require boundary data from their upstream neighbors. These dependencies interfere with the natural parallelism of each grid doing the same thing at the same time.

The solution I'm working with is to have different grids working on different ordinates at the same time. At the first stage of the calculation, only grids that are farthest upstream with respect to some ordinate are active. At later stages boundary information becomes available for grids farther downstream, and the calculation moves inward from the corners of the domain. Figure 2 shows the situation for an 8×8 square arrangement of grids.

Even with a simple square grid layout, the situation becomes complicated when grids must decide which of several competing ordinates to work on next. To organize the calculation, consider first the progress of a single ordinate through the domain. The grid or grids furthest upstream, that depend on no other grids for boundary data, constitute the first "wave" for that ordinate. The grids which can sweep after the first wave is done are the second wave, and so on. (In Fig. 2 the waves are those groups of grids with exactly the same color.) The number of waves required to cross the domain can be considered the width of the domain, and this width plus the number of remaining ordinates in the same quadrant provides a lower bound on the number of stages required to complete the entire transport sweep for the entire ordinate set.

The complete steps into which the computation is broken are what I am calling "stages". During each stage, each ordinate will be swept through all the grids in its next wave, unless some of those grids are taken by an ordinate given higher priority. The current implementation treats this as an all-or-nothing proposition: no grid will sweep an ordinate unless all grids of the same wave are able to do so. (There may be some advantage, though possibly a small one, to allowing some grids to sweep ahead of the rest of their wave, and this may be explored in future implementations. Compare, for example, the demand-driven approach presented by Dorr and Still (1996).)

A different issue is the question of which ordinates should be given higher priority when conflicts are encountered. In Fig. 2 the four quadrants receive equal amounts of work in the early stages, conflicts occur in the middle, but by late stages they are seen to be finishing roughly in step with each other. This is a desirable outcome. If the algorithm consistently favors ordinates in some quadrants over others in resolving conflicts, then those quadrants tend to finish first, and extra stages will be required to complete the sweeps for those ordinates in the "losing" quadrant or quadrants.

Table 1 gives the numbers of stages required for various sizes of ordinate sets and various grid arrays, with different algorithms for resolving conflicts. The runs with different conflict-resolution priorities at different stages gave consistently superior results.

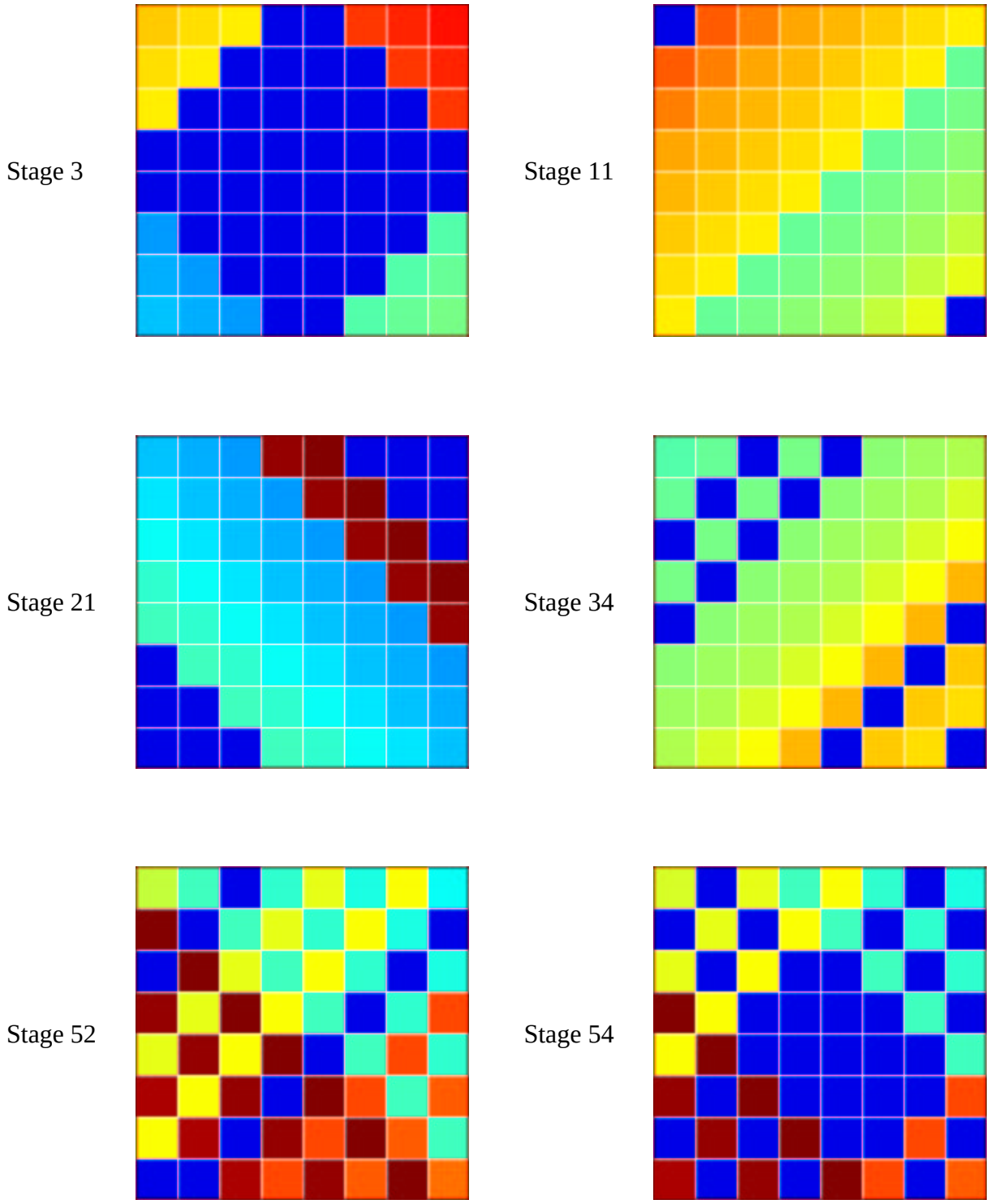


Fig. 2. An 8×8 square of grids required 58 Stages for 40 Ordinates (S_8). Color indicates the ordinate being processed by each grid, with idle grids shown as dark blue.

Note that the number of ordinates is a lower bound for the number of stages, and the "no overlap" column provides an easy upper bound that an optimal scheme should usually be able to improve upon.

Fig 3 shows the more complicated—and typical—case of a mesh level with an irregular grid layout due to adaptive mesh refinement. The collection of 43 grids has a width of 20 waves in each direction. 66 stages were required to sweep 40 ordinates through the grids on the refined level. Each stage therefore had roughly 40% of grids idle on the average, so there seems to be room for improvement. A larger S_{16} ordinate set required 178 stages to sweep 144 ordinates through these same grids, giving only 20% idleness. The same general rule applies to adaptive grid layouts as to the regular grid patterns of table 1: larger ordinate sets yield longer pipelines and more efficient use of any given set of grids.

Another difficulty arises in three dimensional problems. There are grid layouts in 3D that cannot be ordered—that is, there may be a grid A which must be swept before grid B,

Table 1. Number of stages required for transport sweeps in a variety of problem sizes.

size[a]	# of ordinates	width[b]	no overlap[c]	fixed order[d]	with rotations[d]
2×2 S_4	12	3	20	16	12
2×2 S_6	24	3	32	29	24
2×2 S_8	40	3	48	45	40
2×2 S_{12}	84	3	92	89	84
2×2 S_{16}	144	3	152	149	144
4×4 S_4	12	7	36	23	17
4×4 S_6	24	7	48	37	31
4×4 S_8	40	7	64	55	47
4×4 S_{12}	84	7	108	99	91
4×4 S_{16}	144	7	168	159	151
8×8 S_4	12	15	68	35	27
8×8 S_6	24	15	80	51	42
8×8 S_8	40	15	96	71	58
8×8 S_{12}	84	15	140	119	102
8×8 S_{16}	144	15	200	179	162

[a]Size of a square array of grids and the ordinate set used.
[b]The number of stages required to sweep a single ordinate.
[c]The number of stages if all ordinates in each quadrant are swept in pipeline fashion, but without overlap with other quadrants (this is $4 \times (n/4 + width - 1)$ for n ordinates).
[d]The last two columns give the number of stages for the methods discussed in the text—always giving priority to the same quadrants in resolving conflicts, or rotating priority with each stage.

B before C, and C before A. These situations can always be resolved by splitting one or more grids and doing the pieces in two different stages, but the determination of appropriate splittings adds still more complication to the sweeping algorithm. This splitting feature is not yet fully implemented in the code.

So far in this discussion I have been motivated by the needs of a parallel implementation, but I have not yet shown any runs actually performed on a parallel

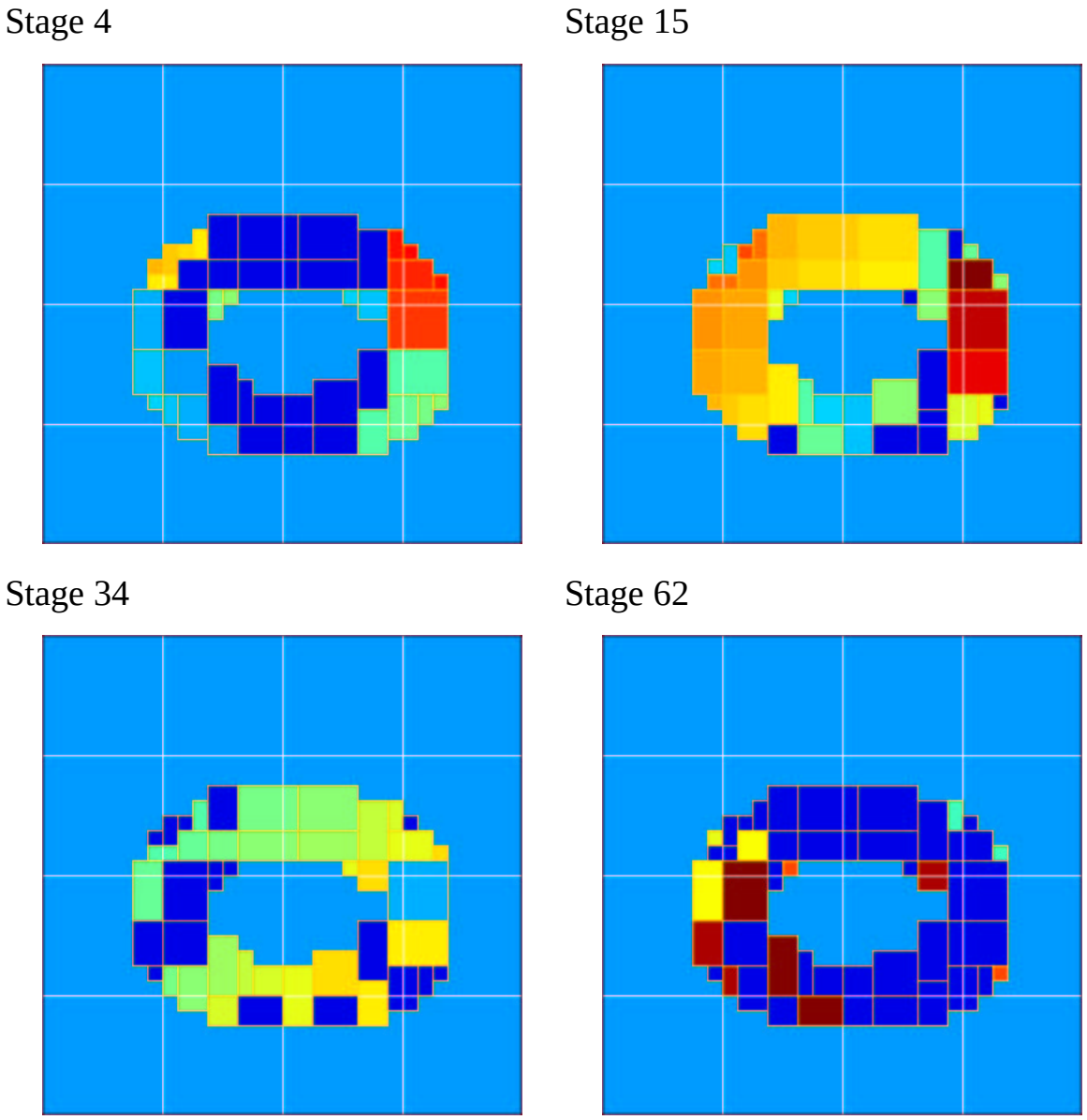


Fig. 3. For 43 AMR grids on a level, 66 stages are required to sweep an S_8 set containing 40 ordinates. Different colors show which ordinates are active on which grids, dark blue shows inactive grids, while light blue shows the region not refined at this level.

machine. I do not have a scaling study to present, but I can at least offer the sample results shown in Fig. 4. This figure examines the grid size tradeoff mentioned above, and shows efficiency improving as grids in an otherwise fixed arrangement become larger. The results suggest that—at least in this preliminary implementation of the algorithm—grid size must be at least 64×64 to keep overheads from dominating the calculation.

Conjugate Gradient Acceleration

In source iteration, three separate quantities are recomputed after every transport sweep. Reflecting boundaries and the AMR refluxing source converge rapidly with successive sweeps, but the scattering source may converge very slowly. Only this source requires convergence acceleration. So long as the scattering is isotropic the problem can be expressed as a symmetric positive definite linear system, so conjugate gradient (CG) acceleration is straightforward. Each step of the CG iteration involves solving, by transport sweeps, a reduced problem with the scattering source held constant while the reflection and AMR sources are iterated to convergence. A version of this algorithm without reflecting boundaries, AMR, or preconditioning appeared in *(Ramone, et al.,*

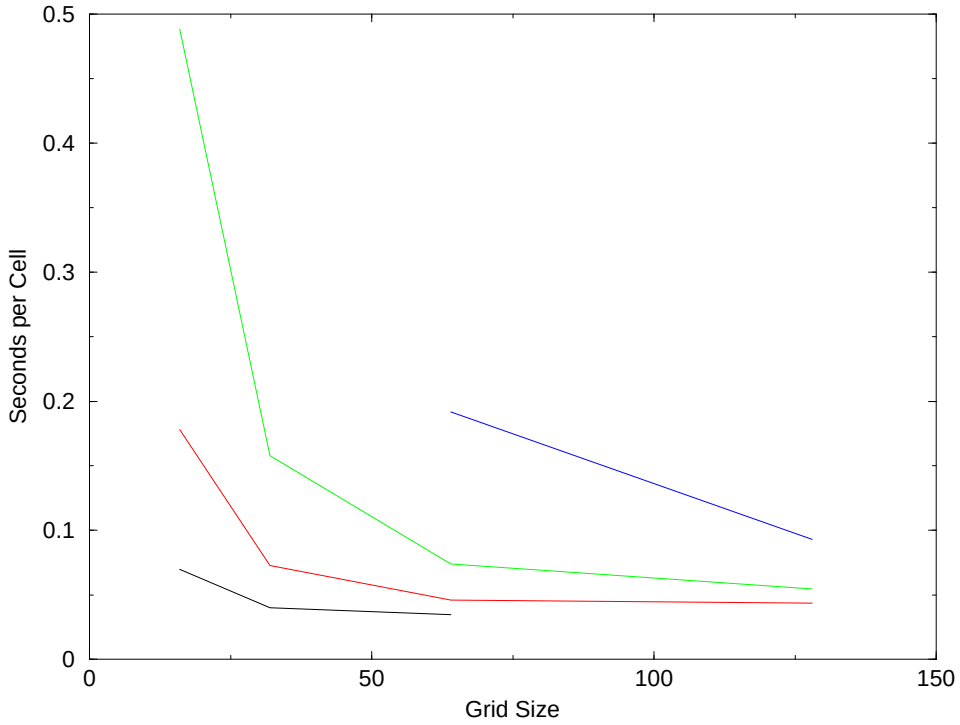


Fig. 4. Timings taken on ASCI Blue Pacific: 4 processors (black), 8 (red), 16 (green), 32 (blue). All runs used the same grid layout, with 64 coarse and 6 fine grids, all grids square with side length shown as the x-axis in the plot above. Times are CPU seconds per cell for initialization plus 10 coarse timesteps of a pure radiation problem using an S_8 ordinate set and step characteristic discretization (that is, 10 nonlinear implicit updates on the coarse level, 20 on the fine level, 1 multilevel initialization, and 10 multilevel synchronizations). All of these were iterated to convergence using transport sweeps.

1997).

We first form a correction equation. The original equation may have inhomogeneous boundary conditions, AMR (sync) sources, and an anisotropic source S_m. The correction equation has only an isotropic source. Let I_m^{old} be the starting point for the iteration—in a time-dependent problem, for example, this may be the solution at the previous timestep. We solve by sweeps for a quantity I_m^{new}, holding ϕ^{old} fixed:

$$(\Omega_m \cdot \nabla) I_m^{\text{new}} + \sigma_t I_m^{\text{new}} = \sigma_s \phi^{\text{old}} + S_m, \quad \phi = \frac{1}{4\pi} \sum_m w_m I_m. \qquad [17]$$

I_m^{new} is not the true solution we are looking for, since it is computed using the wrong scattering source. It does allow us, though, to form the isotropic correction equation (with homogeneous boundary conditions) for a quantity I_m^{corr} as follows:

$$(\Omega_m \cdot \nabla) I_m^{\text{corr}} + \sigma_t I_m^{\text{corr}} = \sigma_s \phi^{\text{corr}} + \sigma_s (\phi^{\text{new}} - \phi^{\text{old}}). \qquad [18]$$

Summing (Eq. 17 and 18) confirms that we can then apply this correction to obtain the true ϕ (and thus I):

$$\phi = \phi^{\text{new}} + \phi^{\text{corr}}. \qquad [19]$$

It is the correction equation (Eq. 18) that we will solve by conjugate gradient.

We next express the problem in matrix form. Let Ax be defined and computed by solving, through transport sweeps, the equation

$$(\Omega_m \cdot \nabla) \psi_m + \sigma_t \psi_m = \sigma_s x, \qquad [20]$$

with homogeneous boundary conditions, and then constructing

$$Ax = x - \frac{1}{4\pi} \sum_m w_m \psi_m. \qquad [21]$$

The equation we want to solve is

$$Ax = \phi^{\text{new}} - \phi^{\text{old}}, \qquad [22]$$

so that $\phi = \phi^{\text{old}} + x$. A is positive definite, and symmetric with an inner product

$$\langle u, v \rangle = \sum_{\text{cells}} \sigma_{si} u_i v_i \Delta x_i \Delta y_i. \qquad [23]$$

(Note that, in AMR problems, coarse cells under fine levels are not included in this inner product.)

Solution of (Eq. 22) is by preconditioned conjugate gradient, which I reproduce here in a form based on that presented by Golub and Van Loan (1983), where M is the preconditioner:

$$
\begin{aligned}
\alpha_k &= \langle z_{k-1}, r_{k-1}\rangle / \langle p_k, Ap_k\rangle \\
x_k &= x_{k-1} + \alpha_k p_k & x_0 &= 0 \\
r_k &= r_{k-1} - \alpha_k A p_k & r_0 &= \phi^{\text{new}} - \phi^{\text{old}} \\
z_k &= M^{-1} r_k & z_0 &= M^{-1} r_0 \\
\beta_{k+1} &= \langle z_k, r_k\rangle / \langle z_{k-1}, r_{k-1}\rangle \\
p_{k+1} &= z_k + \beta_{k+1} p_k & p_1 &= z_0
\end{aligned}
$$

I have experimented with two preconditioners for transport problems so far. A diagonal preconditioner of the form

$$M = \text{diag}\left(1 - \frac{\sigma_s \Delta x}{1 + \sigma_t \Delta x}\right) \qquad [24]$$

adds little cost to the calculation and typically reduces the number of iterations by a factor of two. When the ordinate set is large, solution of a smaller (S_2) system by diagonally preconditioned CG makes an effective preconditioner for the larger system. Note that while Ap_k must be computed accurately to avoid spoiling the residual update, $z_k = M^{-1} r_k$ can be computed to a much looser tolerance. I will explore the performance of these preconditioners in experiments in the next section.

Numerical Results

For numerical tests I use essentially the same 2D steady-state test problem here for discrete ordinates as I used for diffusion in *(Howell and Greenough, 2003).* In the diffusion problem sunlight (represented as an isotropic incoming flux of 400000 erg/cm^2/s) enters through the top of a square domain 1 km on a side by way of a Marshak boundary condition. This flux encounters a horizontal cloud layer across the middle of the square, where the gaps between clouds may not be resolved on coarse grids. Side boundaries are reflecting (Neumann), and the bottom is an absorbing (Marshak) boundary. There is no coupling with the fluid ($\kappa_a = 0$), so energy that does not make it through the cloud layer to the ground passes back out through the top of the domain.

To reproduce the same effects in a transport calculation, I use an isotropic incoming flux through the top of the domain and specular reflection at the side boundaries. The "clouds" are pure scattering regions, with $\kappa_s = 10^{-2}$ cm^{-1} within "clouds" and 10^{-6} cm^{-1} elsewhere. Each cloud is circular, gaps between the clouds are $1/10$ of a cloud diameter, and there are 8.5 clouds across the width of the domain. If refinement is used, only cells near the cloud layer are refined. Fig. 5 shows radiation energy density in a well-resolved AMR calculation of this test problem.

The purpose of the test problem for diffusion was to examine the behavior of the AMR algorithm in a case where a coarse-level feature of the solution (flux through the bottom of the domain) depended strongly on the resolution in a refined region (the cloud layer). This is also an interesting question for transport. In addition, since this is a strongly-scattering problem we can also observe the performance of the acceleration techniques discussed in the previous section.

Tables 2 and 3 show the average flux reaching the bottom of the domain for various grid resolutions (single grid and AMR) and ordinate sets, and for two different spatial discretizations. The S_2 runs used diagonally-preconditioned conjugate gradient, while larger ordinate sets used the S_2 solver as a preconditioner. Note first that for each discretization and ordinate set, the computed fluxes appear to be converging as the fine grid resolution is improved. These fluxes are nearly independent of the resolution in coarser regions, because in these regions the solution is relatively smooth. The timings for the larger AMR calculations are significantly better than for fully-refined problems with equivalent fine resolution. Together, these facts suggest that AMR is successfully delivering a performance improvement without damaging the computed solutions.

Is is also interesting to observe that the S_2 solutions consistently underestimate the flux penetrating the cloud layer. This is due in part to the fact that none of the rays in the

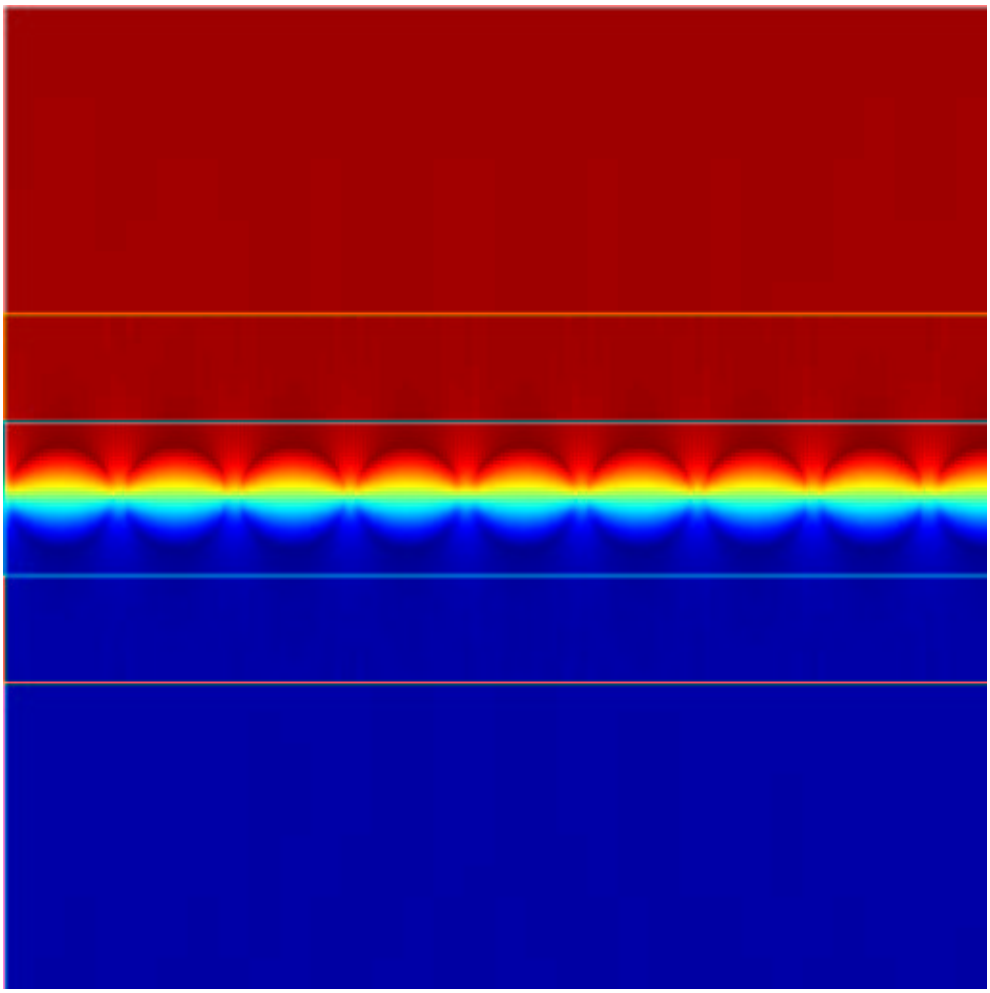


Fig. 5. "Clouds" AMR test problem (described in the text). Base grid is 32×32 with two levels of refinement, each by a factor of 4. The calculation used 40 equally-spaced ordinates and the simple corner balance discretization.

S_2 ordinate set can pass through a gap without hitting one of the clouds on either side. Comparing the two discretizations, we see that at coarse resolutions the step characteristic scheme overestimates the flux passing through the layer, while simple corner balance gives more consistent results. This effect is not related to the gaps at all—it persists even if the "clouds" are replaced by a uniform horizontal layer. Step characteristic can give excellent results for propagation through thin media (see Fig. 1), but for transport through thick scattering media it is at a disadvantage.

The fluxes shown in Tables 2 and 3 can be directly compared to those reported for diffusion in *(Howell and Greenough, 2003).* Diffusion greatly overestimates the flux passing through the cloud layer, even when a flux limiter is used. The timings, though, are not comparable, since different machines were used and different parts of the algorithm were being timed. The diffusion paper also went on to explore time-dependent effects, which I don't have the space to address here.

Finally, Table 4 gives performance data for the various acceleration options discussed in the previous section. It seems clear that conjugate gradient is a great improvement over source iteration, that the diagonal preconditioner is better than none at all, and that using S_2 as a preconditioner can significantly accelerate solutions for larger ordinate sets. The spatial discretizations shown in the table all give reasonable results with conjugate gradient acceleration. One discretization that I have implemented but not shown here, upstream corner balance (UCB), works with source iteration but becomes unstable with acceleration. Further investigation will be needed to correct this problem. Note also the

Table 2. Step characteristic (SC) results for "clouds" test problem.

		S_2		S_8		Equal-spaced	
	Total	4 ordinates		40 ordinates		40 ordinates	
Resolutions[a]	Cells[b]	Flux[c]	Time[d]	Flux[c]	Time[d]	Flux[c]	Time[d]
32	1024	79822	0.133	81872	0.800	82013	1.517
64	4096	55012	0.367	56897	2.350	57061	4.100
128	16384	38177	1.450	42445	9.817	43069	16.03
32,128	7168	38193	1.316	42446	8.733	43069	14.12
256	65536	28297	8.217	34675	56.15	35395	81.42
64,256	20480	28313	4.767	34678	31.33	35396	43.57
512	262144	23300	53.23	31403	339.0	31975	483.7
128,512	57344	23312	17.62	31408	121.0	31977	164.4
32,128,512	48128	23333	17.63	31410	127.8	31977	189.9
1024	1048576	21106	284.8				
256,1024	212992	21113	90.48	30398	629.7	30883	760.4
64,256,1024	167936	21132	82.47	30402	596.5	30884	733.3

[a]Number of cells across domain at each level of refinement.

[b]Total number of cells on all levels.

[c]Flux reaching the bottom of the domain in erg/cm^2/s.

[d]Time for a multilevel solve on a single 1 GHz Alpha EV6.8 processor.

preliminary 3D results at the bottom of the table.

All of the timings in this section have been for a single processor, since running the code in parallel introduces additional complications. One issue is raised by the S_2 preconditioner: while in serial calculations this acceleration scheme improves performance, the S_2 ordinate set gives poor parallel sweep efficiencies because of the shorter pipelines (see Table 1). Comparison of acceleration strategies in parallel calculations will be an important subject for future work.

Acknowledgment

This work was performed under the auspices of the U.S. Department of Energy by the University of California Lawrence Livermore National Laboratory under contract No. W-7405-ENG-48.

References

Adams, M. L., "Subcell Balance Methods for Radiative Transfer on Arbitrary Grids,", *Transp. Theory Stat. Phys.*, **26**, 385–431 (1997).

Almgren, A. S., Bell, J. B., Colella, P., Howell, L. H., and Welcome, M. L., "A Conservative Adaptive Projection Method for the Variable Density Incompressible Navier-Stokes Equations," *J. Comput. Phys.*, **142**, 1–46 (1998).

Bell, J., Berger, M., Saltzman, J., and Welcome, M., "Three-Dimensional Adaptive Mesh Refinement for Hyperbolic Conservation Laws," *SIAM J. Sci. Comput.*, **15**, 127–138

Table 3. Simple corner balance (SCB) results for "clouds" test problem.

		S_2		S_8		Equal-spaced	
	Total	4 ordinates		40 ordinates		40 ordinates	
Resolutions[a]	Cells[b]	Flux[c]	Time[d]	Flux[c]	Time[d]	Flux[c]	Time[d]
32	1024	17742	0.183	20115	0.950	20162	1.550
64	4096	13825	0.433	15842	1.783	15779	2.933
128	16384	18632	1.433	22677	6.583	22933	9.717
32,128	7168	18633	1.233	22678	6.433	22933	10.23
256	65536	19804	6.833	26568	33.87	27006	46.78
64,256	20480	19819	3.416	26571	19.00	27009	26.08
512	262144	20032	35.18	28644	209.2	29097	270.9
128,512	57344	20057	9.333	28651	67.00	29104	86.05
32,128,512	48128	20059	10.38	28654	70.55	29105	106.9
1024	1048576	19994	162.7	29651	1035.	30076	1264.
256,1024	212992	20014	45.87	29658	313.4	30084	367.3
64,256,1024	167936	20031	47.13	29664	286.3	30089	351.3

[a]Number of cells across domain at each level of refinement.

[b]Total number of cells on all levels.

[c]Flux reaching the bottom of the domain in erg/cm^2/s.

[d]Time for a multilevel solve on a single 1 GHz Alpha EV6.8 processor.

(1994).
Berger, M. J., and Colella, P., "Local Adaptive Mesh Refinement for Shock Hydrodynamics," *J. Comput. Phys.*, **82**, 64–84 (1989).
Berger, M. J., and Oliger, J., "Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations," *J. Comput. Phys.*, **53**, 484–512 (1984).
Dorr, M. R., and Still, C. H., "Concurrent Source Iteration in the Solution of

Table 4. Performance of the conjugate gradient acceleration schemes compared against pure source iteration (SI). A variety of spatial discretizations, grid resolutions, and ordinate sets are shown. At the bottom of the table are 3D runs for the same problem in a cubical domain, using the step approximation.

Scheme	Res	Set	Accel[a]	Iter[b]	Sweeps[b]	PreCon[c]	Time[d]
SCB	128	S_2	SI	18472	18472		58.12
			CG	290	876		3.283
			DPCG	112	342		1.433
SCB	128	S_8	SI	18674	18674		560.3
			CG	290	1752		52.88
			DPCG	111	678		20.92
			S_2 PCG	12	84	836	6.583
SCB	128	S_{16}	CG	290	2615		319.4
			DPCG	111	1017		125.2
			S_2 PCG	12	126	828	19.98
SCB	128,512	S_2	SI	19398	19398		227.4
			CG	260	1053		14.03
			DPCG	168	683		9.333
SCB	128,512	S_8	CG	263	2119		264.1
			DPCG	164	1327		166.4
			S_2 PCG	16	143	3353	67.00
StepChar	128,512	S_8	CG	197	1392		398.9
			DPCG	158	1119		323.0
			S_2 PCG	15	118	2892	121.0
SCB	128,512	S_{16}	CG	263	2891		1304.
			DPCG	163	1809		824.3
			S_2 PCG	17	208	3570	144.9
Step	128,512	S_{16}	S_2 PCG	11	129	1889	106.0
Diamond	128,512	S_{16}	S_2 PCG	19	274	5244	237.7
StepChar	128,512	S_{16}	S_2 PCG	16	163	3108	265.3
3D Step	32	S_2	DPCG	38	205		4.617
3D Step	64	S_2	DPCG	54	280		61.38

[a]CG is pure conjugate gradient, DPCG is diagonally preconditioned, etc.
[b]Iterations and transport sweeps required for the outer CG iteration.
[c]Separate S_2 transport sweeps required for preconditioner.
[d]Time for a multilevel solve on a single 1 GHz Alpha EV6.8 processor.

Three-Dimensional, Multigroup Discrete Ordinates Neutron Transport Equations," *Nuc. Sci. Eng.*, **122**, 287–308 (1996).

Golub, G. H., and Van Loan, C. F., *Matrix Computations*, (Johns Hopkins University Press, Baltimore, 1983).

Howell, L. H., and Bell, J. B., "An Adaptive Mesh Projection Method for Viscous Incompressible Flow," *SIAM J. Sci. Comput.*, **18**, 996–1013 (1997).

Howell, L. H., and Greenough, J. A., "A Block-Structured Adaptive Mesh Refinement Algorithm for Diffusion Radiation," Proceedings of the 10th Nuclear Explosives Code Development Conference, Las Vegas, Nevada, October 25–30, 1998 (1999).

Howell, L. H., and Greenough, J. A., "Radiation Diffusion for Multi-Fluid Eulerian Hydrodynamics with Adaptive Mesh Refinement," *J. Comput. Phys.*, **184**, 53–78 (2003).

Howell, L. H., Pember, R. B., Colella, P., Jessee, J. P., and Fiveland, W. A., "A Conservative Adaptive Mesh Algorithm for Unsteady, Combined-Mode Heat Transfer Using the Discrete Ordinates Method," *Num. Heat Transfer B*, **35**, 407–430 (1999).

Jessee, J. P., Fiveland, W. A., Howell, L. H., Colella, P., and Pember, R. B., "An Adaptive Mesh Refinement Algorithm for the Radiative Transport Equation," *J. Comput. Phys.*, **139**, 380–398 (1998).

Levermore, C. D., and Pomraning, G. C., "A Flux-Limited Diffusion Theory," *Astrophys. J.*, **248**, 321–334 (1981).

Lewis, E. E., and Miller, W. F. Jr., *Computational Methods of Neutron Transport*, (American Nuclear Society, La Grange Park, IL, 1993).

Mathews, K. A., "On the Propagation of Rays in Discrete Ordinates," *Nucl. Sci. Eng.*, **132**, 155–180 (1999).

Ramone, G. L., Adams, M. L., and Nowak, P. F., "A Transport Synthetic Acceleration Method for Transport Iterations," *Nucl. Sci. Eng.*, **125**, 257–283 (1997).

Rendleman, C. A., Beckner, V. E., Lijewski, M., Crutchfield, W., and Bell, J. B., "Parallelization of Structured, Hierarchical Adaptive Mesh Refinement Algorithms," *Comput. Visual. Sci.* **3**, 147–157 (2000).

Truelove, J. K., Klein, R. I., McKee, C. F., Holliman, J. H. II, Howell, L. H., and Greenough, J. A., "The Jeans Condition: A New Constraint on Spatial Resolution in Simulations of Isothermal Self-Gravitational Hydrodynamics," *Astrophys. J.*, **489**, L179–L183 (1997).