

# MULTILEVEL SOURCE ITERATION ACCELERATORS FOR THE LINEAR TRANSPORT EQUATION IN SLAB GEOMETRY

C. T. KELLEY\*

**Abstract.** In this paper we show how classical error estimates for various discretizations of the source iteration map in slab geometry can be used to construct accurate approximate inverses in the context of fast multilevel methods. For discretizations that give strongly convergent collectively compact sequences of approximate source iteration maps, the Atkinson-Brakhage approximate inverse can be applied. For discretizations that give rise to norm convergent sequences, a more direct approach can be used. Our implementation of these ideas, based on use of GMRES iteration to solve the coarse mesh problems, gives the solution to an accuracy of fine mesh truncation error at a cost proportional to that of an evaluation of the fine mesh source iteration map. These methods require only the source iteration map and are hence easier to adapt to multiprocessor computers than methods that require solution of diffusion equations. We illustrate our results with a report on numerical experiments with both strongly and norm convergent source iteration maps using the Kendall Square KSR1 computer.

**Key words.** transport equation, source iteration, Atkinson-Brakhage preconditioner, Kendall Square KSR1 Computer

**AMS(MOS) subject classifications.** 45L10, 85A25, 65Y05, 65F10,

**1. Introduction.** In this paper we show how classical error estimates for various discretizations of the source iteration map in slab geometry can be used to construct accurate approximate inverses in the context of fast multilevel methods. For discretizations that give strongly convergent collectively compact sequences of approximate source iteration maps, the Atkinson-Brakhage [4], [6], approximate inverse can be applied. For discretizations that give rise to norm convergent sequences, a more direct approach can be used. Our implementation of these ideas, based on use of GMRES iteration to solve the coarse mesh problems, gives the solution to an accuracy of fine mesh truncation error at a cost proportional to that of an evaluation of the fine mesh source iteration map. These methods require only the source iteration map and are hence easier to adapt to multiprocessor computers than methods that require solution of diffusion equations. We illustrate our results with a report on numerical experiments with both strongly and norm convergent source iteration maps using the Kendall Square KSR1 computer. Our methods should extend to more general two and three dimensional geometries if appropriate source iteration maps can be constructed. The ideas in this paper have also been applied to integral equations [14].

---

\* North Carolina State University, Center for Research in Scientific Computation and Department of Mathematics, Box 8205, Raleigh, N. C. 27695-8205, USA. This research was supported by National Science Foundation grant #DMS-9024622 and Air Force Office of Scientific Research grant #AFOSR-FQ8671-9101094. Computing was supported by an allocation of time from the North Carolina Supercomputing Center.

Consider the monoenergetic transport equation in slab geometry with isotropic scattering written in the form

$$(1.1) \quad \mu \frac{\partial \psi}{\partial x}(x, \mu) + \psi(x, \mu) = \frac{c(x)}{2} \int_{-1}^1 \psi(x, \mu') d\mu' + q(x)$$

for  $x \in (0, \tau)$  with boundary conditions

$$(1.2) \quad \psi(0, \mu) = F_l(\mu), \mu > 0; \psi(\tau, \mu) = F_r(\mu), \mu < 0.$$

In (1.1) and (1.2) the intensity  $\psi$  is the unknown real valued function of  $x$  and  $\mu$ ,  $\tau < \infty$ ,  $c, q \in C([0, \tau])$ ,  $F_l$  and  $F_r$  are given continuous real valued functions of  $\mu$ , and  $k$  is a given continuous real valued function of  $x$ ,  $\mu$ , and  $\mu'$ . It is known [8] that the boundary value problem (1.1) – (1.2) has a unique solution if  $0 \leq c(x) < 1$ , say.

Our work will apply to the more general boundary conditions discussed in [22] and to multigroup and to certain anisotropic problems as well. The extension to the multigroup setting is trivial even if there is upscattering. The extension to anisotropic problems for which the phase function has a finite Legendre polynomial expansion was discussed in [26]. In § 4 we consider an anisotropic multigroup problem with upscattering as an example. Our results also extend to discontinuous  $c$  and  $q$  provided the spatial mesh points contain all points of discontinuity as was required in [22]. We discuss this point more completely in § 3. However, the important ideas can be completely illustrated by the simple problem in (1.1) – (1.2).

To establish notation and set the ideas we derive the continuous form of the source iteration map. Let

$$(1.3) \quad f(x) = \frac{1}{2} \int_{-1}^1 \psi(x, \mu') d\mu'.$$

If  $f$  is known, we can recover  $\psi$  by integration. For  $\mu > 0$  we integrate (1.1) forward in  $x$ , obtaining

$$\psi(x, \mu) = \frac{1}{\mu} \int_0^x \exp(-(x-y)/\mu)(cf+q)(y) dy + \exp(-x/\mu)F_l(\mu), \mu > 0.$$

Similarly, for  $\mu < 0$ , we integrate backwards to obtain,

$$\begin{aligned} \psi(x, \mu) &= -\frac{1}{\mu} \int_x^\tau \exp(-(x-y)/\mu)(cf+q)(y) dy \\ &+ \exp((\tau-x)/\mu)F_r(\mu) \\ &= \frac{1}{|\mu|} \int_x^\tau \exp(-|x-y|/|\mu|)(cf+q)(y) dy \\ &+ \exp(-|\tau-x|/|\mu|)F_r(\mu), \mu < 0. \end{aligned}$$

As is standard, [8], [9], the analysis above leads to an integral equation for  $f$ .

$$(1.4) \quad f(x) - \int_0^\tau k(x, y)f(y) dy = g(x)$$

where

$$k(x, y) = \frac{1}{2}E_1(|x - y|)c(y),$$

$$E_1(|x - y|) = \int_0^1 \exp(-|x - y|/\nu) \frac{d\nu}{\nu},$$

and

$$\begin{aligned} g(x) &= \frac{1}{2} \int_0^1 \exp(-x/\nu) F_l(\nu) d\nu \\ &+ \frac{1}{2} \int_0^1 \exp(-(\tau - x)/\nu) F_r(-\nu) d\nu + \frac{1}{2} \int_0^\tau E_1(|x - y|)q(y) dy \end{aligned}$$

Source iteration, in its continuous form, is simply summation of the Neumann series for (1.4):

$$(1.5) \quad f_{n+1} = \mathcal{K}(f_n) + g,$$

where

$$\mathcal{K}(f)(x) = \int_0^\tau k(x, y)f(y) dy.$$

Since  $I - \mathcal{K}$  is not invertible when  $\tau = \infty$  and  $c(x) = 1$  for all  $x$ , we should expect the iteration to be slowly convergent for large values of  $\tau$  and functions  $c(x)$  near 1.

Because of this slow convergence, preconditioning becomes important. In this paper we consider preconditioning by approximate inversion. Recall that an operator  $B$  on a Banach space  $X$  is an approximate inverse of  $A$  if

$$\|I - BA\|_{\mathcal{L}(X)} < 1.$$

Here  $\mathcal{L}(X)$  is the space of bounded operators on  $X$ . If  $B$  is an approximate inverse for  $I - \mathcal{K}$  then we may replace the unpreconditioned iteration (1.5) by

$$(1.6) \quad f_{n+1} = (I - B(I - \mathcal{K}))f_n + Bg.$$

The rate of convergence of the iteration (1.5) is the spectral radius of  $\mathcal{K}$  and that of (1.6) that of  $I - B(I - \mathcal{K})$ . Hence, a good approximate inverse can make a substantial difference in performance.

In § 2 we describe in general terms how approximate inverses can be used in multilevel methods. We then consider the norm and strongly convergent situations and describe the Atkinson-Brakhage approximate inverse. In source iteration, the discretized map returns the residual and not the approximate linear map. We indicate how both approaches can be adapted to this situation and how use of GMRES as a coarse mesh solver is useful in this case. In § 2 we also say something about other approaches to the solution of (1.5). In § 3 we discuss how the error analyses in [22] and [26] have direct implications for multilevel methods. In particular, we point out that the diamond difference discrete ordinate approximation can be implemented to give either norm convergent or strongly convergent collectively compact sequences of approximate source iteration maps. We point out the natural parallelism of both implementations. Finally, in § 4 we report on two computations using both implementations of the diamond difference discretization that illustrate the ideas.

**2. Approximate Inverses and Multilevel Methods.** In this section we will discuss how approximate inverses can be used in multilevel methods. We will restrict our attention to linear compact fixed point problems and will construct our approximate inverses from sequences of approximations to the fixed point map. In the case of norm convergent sequences of maps, it is clear, at least at the theoretical level, how to do this. If the sequence of approximating maps is strongly convergent and collectively compact, but not norm convergent, the Atkinson-Brakhage approximate inverse may be used and we sketch its construction. We put this discussion fully in the context of the transport equation in § 3, but will make remarks on the transport theoretic context throughout this section.

Suppose that  $\mathcal{K}$  is a compact operator on  $X$ , that  $(I - \mathcal{K})^{-1}$  is bounded and that we wish to solve

$$f - \mathcal{K}f = g.$$

We let  $\{\mathcal{K}_m\}$  be a sequence of approximating maps. We will assume that either  $\mathcal{K}_m \rightarrow \mathcal{K}$  in the operator norm or that family  $\{\mathcal{K}_m\}$  is strongly convergent to  $\mathcal{K}$  and collectively compact. Both situations are relevant to solution of the transport equation.

Recall that a sequence  $\{\mathcal{K}_n\}$  of bounded operators on  $X$  *converges strongly* to  $\mathcal{K} \in \mathcal{L}(X)$  if

$$\mathcal{K}_n x \rightarrow \mathcal{K}x$$

for each  $x \in X$ . We will denote strong convergence by

$$\mathcal{K}_n \xrightarrow{s} \mathcal{K}.$$

It is important to remember that strong convergence does not imply convergence in the operator norm. We say that a sequence of operators  $\{\mathcal{K}_n\}$  is *collectively compact* [2] if

$$\cup_n \mathcal{K}_n \mathcal{B}$$

is precompact in  $X$ . Here  $\mathcal{B}$  denotes the unit ball in  $X$ .

Note that if  $\mathcal{K}_m \rightarrow \mathcal{K}$  in norm, then the family  $\{\mathcal{K}_m\}$  is also strongly convergent to  $\mathcal{K}$  and collectively compact.

The following theorem from [2] states that strong convergence (consistency) and collective compactness (which, when added to strong convergence, implies stability) will imply convergence.

**THEOREM 2.1.** *Let  $\mathcal{K}$  be a compact operator on  $X$  and let  $\{\mathcal{K}_n\}$  be a sequence of collectively compact operators converging strongly to  $\mathcal{K}$ . Assume that  $(I - \mathcal{K})^{-1}$  is bounded. Then  $(I - \mathcal{K}_n)^{-1}$  is bounded for  $n$  sufficiently large and  $(I - \mathcal{K}_n) \xrightarrow{s} (I - \mathcal{K})^{-1}$ .*

If the spectral radius of  $\mathcal{K}$  (or  $\mathcal{K}_L$  for some  $L$  large enough to provide the desired accuracy) is small, Richardson iteration will converge rapidly to the solution. However if the spectral radius is large, Richardson iteration may converge slowly or not at all. Suppose, however, that  $I - \mathcal{K}_l$  is inexpensive to invert for small values of  $l$ . Then one might hope that  $(I - \mathcal{K}_l)^{-1}$  would be a good approximate inverse for  $(I - \mathcal{K})$  and that a preconditioned Richardson iteration based on  $B = (I - \mathcal{K}_l)^{-1}$  would work well. This is indeed the case if  $\mathcal{K}_m \rightarrow \mathcal{K}$  in norm. If, however, the convergence is only strong, one can still use collective compactness to construct an approximate inverse but at the expense of an additional evaluation of  $K$ . The operator  $B = I + (I - \mathcal{K}_l)^{-1}K$  is a good approximate inverse for  $l$  sufficiently large as Theorem 2.2 essentially from [4] asserts. For the purposes of this theorem, we let

$$\mathcal{K}_\infty = \mathcal{K}.$$

**THEOREM 2.2.** *Let  $\mathcal{K}$  be a compact operator on  $X$  and let  $\{\mathcal{K}_n\}$  be a sequence of collectively compact operators converging strongly to  $\mathcal{K}$ . Assume that  $(I - \mathcal{K})^{-1}$  is bounded. Then for all  $\rho > 0$  there is  $l$  such that for all  $l \leq L \leq \infty$ ,*

$$B_l^L = I + (I - \mathcal{K}_l)^{-1} \mathcal{K}_L$$

satisfies

$$(2.1) \quad \|I - B_l^L(I - \mathcal{K}_L)\|_{\mathcal{L}(X)} \leq \rho.$$

We will define our algorithms in terms of the approximate inverse. In the norm convergent case

$$(2.2) \quad B_l^L = (I - \mathcal{K}_l)^{-1}$$

and in the strongly convergent collectively compact case

$$(2.3) \quad B_l^L = I + (I - \mathcal{K}_l)^{-1} \mathcal{K}_L.$$

The original Atkinson-Brakhage method, as described in [4], computes a fine mesh solution using the approximate inverse in (2.3) and is referred to as a two-grid method. The method is the same if one can use (2.2). The two level algorithm to solve

$$(2.4) \quad f^L - \mathcal{K}_L f^L = g$$

is

ALGORITHM 2.1. *Algorithm grid2*( $g, f, \{\mathcal{K}_i\}, l, L, \epsilon$ )

While  $\|f - \mathcal{K}_L f - g\| > \epsilon$ :

$$(2.5) \quad f = (I - B_l^L(I - \mathcal{K}_L))f + B_l^L g$$

In the norm convergent case, it is clear that for sufficiently large  $l$  that  $B_l^L$  will be an approximate inverse and that Algorithm `grid2` will converge. This is also true in the strongly convergent collectively compact case by Theorem 2.2. In the norm convergent case, application of  $B_l^L$  requires only a coarse mesh solve. However in the strongly convergent case, each application of  $B_l^L$  requires an evaluation of  $\mathcal{K}_L$ , however  $B_l^L$  need only be applied once for each iterate because (2.5) can be written as

$$(2.6) \quad f = f + B_l^L(g - (I - \mathcal{K}_L)f).$$

Hence, in the strongly convergent case, each iterate in Algorithm `grid2` requires two fine mesh evaluations of the action of  $\mathcal{K}_L$  on a function, one for the evaluation of  $B_l^L$  and the other for the evaluation of  $(I - \mathcal{K}_L)f$ . Typically one initializes Algorithm `grid2` by solving  $f^l - \mathcal{K}_l f^l = g$  and setting  $f = f^l$ . The other inputs to Algorithm `grid2` are the right hand side, the sequence of maps (*i. e.* a subroutine that will compute them), the coarse and fine levels, and an estimate of truncation error.

A fully multilevel version, first advocated for the Atkinson-Brakhage approximate inverse in [15], fixes the coarse level, at  $l$ , say, and refines the fine level as the iteration progresses. Such an approach is call “nested iteration” in [11] and contributes greatly to efficiency. For solution of (2.4) the algorithm would be:

ALGORITHM 2.2. *Algorithm gridnest*( $g, f, \{\mathcal{K}_i\}, l, L, \{\epsilon_i\}$ )

1. Solve  $f^l - \mathcal{K}_l f^l = g$  on the coarse level.
2. Set  $f = f^l$ .
3. For  $m = l + 1, \dots, L$   
 $\text{grid2}(g, f, \{\mathcal{K}_i\}, l, m, \epsilon_m)$

4. Set  $f^L = f$ .

The inputs to Algorithm `gridnest` are the same as those to Algorithm `grid2` with the exception that a sequence of estimated truncation errors is needed. As discussed in [5], if the truncation errors decay geometrically and the cost of evaluation of  $\mathcal{K}_m$  increases geometrically, then the total cost of a solve using Algorithm `gridnest` is a small number of evaluations of  $\mathcal{K}_L$ . This type of cost analysis is standard for multilevel methods and we go into detail here to illustrate the difference between the strong and norm convergent cases and to express the cost estimates in transport theoretic terms.

Let  $C_m$  denote the cost of computing  $\mathcal{K}_m u$  and assume that there are  $M_C > 1$  and  $m_\epsilon < 1$  such that

$$C_m \approx M_C C_{m-1} \text{ and } \epsilon_m \approx m_\epsilon \epsilon_{m-1}.$$

If  $\rho < m_\epsilon$  then only one Richardson iterate will be necessary within each call to `grid2` in the for loop in Algorithm `gridnest`. If we denote the number of fine evaluations of  $\mathcal{K}_m$  required by each call to `grid2` by  $M_B$ , then  $M_B = 1$  in the norm convergent case and  $M_B = 2$  in the strong convergent situation. The total cost of a call to `grid2` is  $M_B$  evaluations of  $\mathcal{K}_m$ , whatever cost is associated with the coarse mesh solve, and some vector arithmetic. Now, the cost of the vector arithmetic is negligible. If we assume for the present that the cost of the coarse mesh work in the evaluation of  $B_l^m$  and the coarse mesh solve can be neglected (which may not be so), then the cost of the solve is dominated by the evaluations of  $\mathcal{K}_m$  in the for loop. This cost is

$$(2.7) \sum_{m=l+1}^L M_B C_m = M_B \sum_{m=l+1}^L M_C^{m-L} C_L = \frac{M_B C_L (1 - M_C^{l+1-L})}{1 - M_C^{-1}} \leq \frac{M_B C_L}{1 - M_C^{-1}}.$$

In § 3 we will see that  $C_L = O(N_L^A N_L^S)$  where  $N_L^A$  and  $N_L^S$  are the number of points in the angular and spatial meshes respectively.

As  $L$  becomes large, the estimate above becomes sharper. Note that the algorithm is dominated only by evaluations of  $\mathcal{K}$  at various levels and that, as we shall see in § 4, attention to the routines that evaluate  $\mathcal{K}$  can result in very good performance. This is especially the case for source iteration which has a natural parallelism. Moreover, as we will see in § 4, the cost of the computation is dominated by fine mesh  $\mathcal{K}$  evaluations and hence the computational assumptions made in deriving (2.7) are valid. If satisfaction of the termination criterion is checked at each mesh level, as is done in [5], then the factor  $M_B$  in (2.7) becomes  $M_B + 1$ .

As we shall see in § 3 the discretized source iteration algorithm does not directly return  $\mathcal{K}_m f$  but rather an approximation to the affine fixed point map

$$(2.8) \quad \mathbf{K}_m(f) = g_m + \mathcal{K}_m f,$$

where  $g_m = \mathbf{K}_m(0) \rightarrow g$  as  $m \rightarrow \infty$ . The maps  $\{\mathbf{K}_m\}$  are collectively compact as nonlinear or affine maps and  $\mathcal{K}_m$  can be recovered from  $\mathbf{K}_m$  by

$$(2.9) \quad \mathcal{K}_m f = \mathbf{K}_m(f) - \mathbf{K}_m(0).$$

The implementation of the algorithm changes if  $\mathbf{K}_m$  is used instead of  $\mathcal{K}_m f$ . We replace (2.6) by

$$(2.10) \quad f = f + B_l^L(\mathbf{K}_L(f) - f).$$

The right hand side of (2.10) differs from that of (2.6) by  $B_l^L(g - g_L)$ , which tends to 0 as  $L \rightarrow \infty$ . Hence the difference in the algorithms and their implementation on the fine mesh is negligible. In the strongly convergent case, to compute  $B_l^L \mathbf{K}_L(f)$  we must compute

$$(I - \mathcal{K}_l)^{-1} \mathcal{K}_L(\mathbf{K}_L(f) - f).$$

If we set  $w = \mathbf{K}_L f - f$  then

$$\mathcal{K}_L w = \mathbf{K}_L(w + f) - \mathbf{K}_L(f)$$

can be evaluated without an evaluation of  $\mathbf{K}_L(0)$ . The norm convergent case is similar.

The coarse mesh equation is

$$(2.11) \quad (I - \mathcal{K}_l)v = p$$

where  $p = w$  in the norm convergent case and  $p = \mathcal{K}_L w$  in the strong convergent case. To solve (2.11) we use GMRES [27] to solve this linear compact fixed point problem with the zero function as the initial iterate. The results in [25] and the boundedness of the sequence  $\{\mathcal{K}_l\}$  (in both the norm and strong convergent cases) guarantee that the number of GMRES iterates required to solve (2.11) can be bounded independently of the level  $l$  or the right hand side  $p$ . To use GMRES, only computation of residuals and use of (2.9) are required [27]. Since  $\mathbf{K}_l(0)$  is computed on entry into the GMRES iteration, the use of (2.9) requires no extra work. Hence Algorithm `gridnest` can be used with minimal modification if  $\mathbf{K}_L(f)$  rather than  $\mathcal{K}_L f$  is returned. Use of an iterative method such as GMRES implies that the exact solution to the coarse mesh equations will not be returned. Instead of

$$v = (I - \mathcal{K}_l)^{-1} p$$

we compute  $\bar{v}$  such that

$$(2.12) \quad \|(I - \mathcal{K}_l)v - p\|_{L^2} \leq \eta \|p\|_{L^2},$$

for some  $\eta \in (0, 1)$ . By using GMRES instead of a direct factorization method we may attack problems with a very large number of coarse mesh unknowns. We report on such a problem in § 4. The use of GMRES in connection with the Atkinson-Brakhage iteration was first proposed in [13] in the context of nonlinear problems and has been further developed in [16]. The code used in [16] was used, without modification, to generate the results for the strong

convergent case reported in § 4. GMRES requires only a matrix-vector product evaluation subroutine and an inner product. For an inner product we used the trapezoid rule based on the spatial mesh.

Other approaches for improving the efficiency of source iteration include diffusion-synthetic acceleration, [20], [21], which constructs an approximate inverse using the diffusion approximation to the transport equation. It can be applied to anisotropic, [29], [24], and multigroup, [1] problems as well, and be used in connection with multigrid methods. Recent analysis, [10], [3], has precisely the regimes for which DSA will be most effective. Typically the spectral radius of  $I - B(I - \mathcal{K})$  is less than .25. The principal disadvantage of the DSA preconditioner is that a diffusion equation must be solved to apply the preconditioner. DSA, or any other preconditioner can also be applied to a Krylov subspace iterative method, such as conjugate gradient or GMRES, rather than the simple Richardson iteration in (1.6). In view of the fact that  $\mathcal{K}$  is compact, such iterative methods should be effective and we advocate (unpreconditioned) GMRES as a coarse mesh solver in this paper.

Another approach is the second kind multigrid method from [12]. Like the approximate inverse based methods that we advocate here, these methods are based on the expression of the transport equation as a compact fixed point problem. It is somewhat easier to verify the assumptions needed for a convergence proof for the approximate inverse based methods and, in our view, the approximate inverse based methods are much easier to code and implement. The cost of second kind multigrid and Algorithm `gridnest` for the strong convergent case are the same in terms of applications of the fine mesh source iteration method. If a norm convergent sequence of source iterations maps is available, Algorithm `gridnest` is more efficient. Implementation of other multigrid approaches have been discussed in [23] and [24].

**3. Applicability of the Diamond Difference Discretization.** In this section we indicate how the error analyses in [22] and [26] can be used to show who the diamond difference form of discrete ordinates can be used to construct approximate source iteration maps of both the norm and strongly convergent types. The difference in the two formulations is only in the way the maps are applied to data coming from finer meshes, *i. e.* the coarse to fine intergrid transfer.

We construct our sequence of approximate source iteration maps with the diamond difference discretization. We use a variation on the notation of [22] to express the scheme. At level  $m$  we have a spatial mesh  $\{x_i^m\}_{i=1}^{N_m^S}$  with

$$x_1^m = 0 < \dots < x_{N_m^S}^m = \tau,$$

and nodes  $\{\mu_j^m\}_{j=1}^{N_m^A}$  and weights  $\{w_i^m\}_{i=1}^{N_m^A}$  for a quadrature rule in angle.

Given  $f \in C[0, \tau]$  the diamond difference method first computes approximations  $\psi_i(x)$  to  $\psi(x, \mu_j^m)$  by computing  $\psi_j(x_i^m)$  for each  $i$  and extending by

piecewise linear interpolation. If we let  $h_i^m = x_{i+1}^m - x_i^m$  the discretized forward integration becomes

$$(3.1) \quad \begin{aligned} & \mu_j^m \frac{\psi_j(x_i^m) - \psi_j(x_{i-1}^m)}{h_i^m} + \frac{\psi_j(x_i^m) + \psi_j(x_{i-1}^m)}{2} \\ &= \frac{(cf)(x_i^m) + (cf)(x_{i-1}^m)}{2} + \frac{q(x_i^m) + q(x_{i-1}^m)}{2}, \mu_j^m > 0, \end{aligned}$$

with initial data  $\psi(x_1^m) = \psi_j(0) = F_l(\mu_j^m)$ . Similarly the discretized backward integration is

$$(3.2) \quad \begin{aligned} & \mu_j^m \frac{\psi_j(x_i^m) - \psi_j(x_{i+1}^m)}{h_i^m} + \frac{\psi_j(x_i^m) + \psi_j(x_{i+1}^m)}{2} \\ &= \frac{(cf)(x_i^m) + (cf)(x_{i+1}^m)}{2} + \frac{q(x_i^m) + q(x_{i+1}^m)}{2}, \mu_j^m < 0, \end{aligned}$$

with final data  $\psi_j(x_{N_S^m}^m) = \psi_j(\tau) = F_r(\mu_j^m)$ . If  $c$  is piecewise discontinuous, we assume that each discontinuity of  $c$  is a spatial mesh point and define the product  $cf$  at that point by the appropriate one-sided limit. Clearly, there is a natural parallelism in the two discrete forward and backward integrations in that they may be done for each  $\mu_j^m$  simultaneously.

Following the forward and backward integration we compute the affine map

$$(3.3) \quad \mathbf{K}_m(f)(x) = \frac{1}{2} \sum_{j=1}^{N_m^A} \psi_j(x) w_j^m.$$

Note that the right hand side of (1.4) is built into  $\psi_j$  by the way in which the integration is done. This is the map we use Algorithm `gridnest` in the way discussed in § 2.

We wish to refine the spatial and angular mesh simultaneously as  $m$  is increased. Very general conditions on the spatial and angular meshes that guarantee stability and convergence have been given in [26] for the constant  $c$  case of (1.1) in a variational setting. These results include the norm convergence of the source iteration operator. The results in [26] will be used to obtain the strong convergence and collective compactness results for the form of the diamond difference operator used in [22] as the spatial and angular meshes are refined simultaneously. If the angular mesh is held fixed, strong convergence and collective compactness of the approximate source iteration maps were verified in [22] for several spatial differencing schemes.

We make the assumption on the spatial mesh used in [22] and [26].

**ASSUMPTION 3.1.** *Let  $h_i^m = x_{i+1}^m - x_i^m$ . There is  $\gamma_M$  such that for all  $m$*

$$\bar{h}_m = \max_j h_j^m \leq \gamma \min_j h_j^m.$$

Our assumption on the angular mesh is the one used in the error estimates in [26].

ASSUMPTION 3.2. *The nodes  $\{\mu_j^m\}_{j=1}^{N_m^A}$  and weights  $\{w_i^m\}_{i=1}^{N_m^A}$  are either*

1. *Gaussian quadratures on  $[-1, 1]$  or*
2. *double Gaussian quadratures on  $[-1, 0] \cup [0, 1]$ .*

The results in [26] are for the case in which  $c$  is constant. To state the results, we define the affine map

$$\begin{aligned} \mathbf{T}(f)(x) &= \int_0^\tau E_1(x-y)(f+q)(y) dy + \int_0^1 \exp(-x/\nu) F_l(\nu) d\nu \\ &\quad + \int_0^1 \exp(-(\tau-x)/\nu) F_r(-\nu) d\nu \end{aligned}$$

and the approximation

$$\mathbf{T}_m(f)(x) = \sum_{j=1}^{N_m^A} \psi_j(x) w_j^m.$$

Note that for any  $u \in C[0, \tau]$ ,

$$\mathbf{K}(u) = \frac{1}{2} \mathbf{T}(cu) \text{ and } \mathbf{K}_m = \frac{1}{2} \mathbf{T}_m(cu).$$

Define linear operators

$$\mathbf{L}(f)(x) = \int_0^\tau E_1(x-y) f(y) dy = \mathbf{T}(f)(x) - \mathbf{T}(0)(x)$$

and

$$\mathbf{L}_m(f)(x) = \mathbf{T}_m(f)(x) - \mathbf{T}_m(0)(x).$$

Hence, in the notation above,

$$\mathcal{K}_m = \frac{c}{2} \mathbf{L}_m.$$

In [26] the spatial integrations are defined with a variational formulation.  $\mathbf{T}$  is approximated by  $\mathbf{T}_m^1$ , which is defined as follows. Let  $V^m$  be the space of piecewise linear functions with nodes at  $\{x_i^m\}_{i=1}^{N_m^S}$ . For  $j = 1, \dots, N_m^A$  compute  $\bar{\psi}_j \in V^m$  from the equations

$$(3.4) \quad \int_{\mathcal{I}_i} (\mu \bar{\psi}_j' + \bar{\psi}_j)(y) dy = \int_{\mathcal{I}_i} (f+q)(y) dy$$

for  $i = 1, \dots, N_m^S - 1$ , where  $\mathcal{I}_i = (x_i^m, x_{i+1}^m)$ . Then set

$$(3.5) \quad \mathbf{T}_m^1(f) = \sum_{j=1}^{N_m^A} \bar{\psi}_j(x) w_j^m.$$

The linear maps  $\mathbf{L}_m$  and  $\mathbf{L}_m^1$  defined by

$$\mathbf{L}_m^1(f) = \mathbf{T}_m^1(f) - \mathbf{T}_m^1(0)$$

are identical on  $V^m$ , however they differ in an important way elsewhere.  $\mathbf{L}_m(f)$  evaluates  $f$  at the spatial mesh points whereas  $\mathbf{L}_m^1(f)$  integrates  $f$  over intervals. The maps can be related by the formula

$$(3.6) \quad \mathbf{L}_m = \mathbf{L}_m^1 P_m$$

where  $P_m$  is the piecewise linear interpolation map from  $C[0, 1]$  to  $V^m$ . This is an important distinction in the two grid iteration, Algorithm `grid2`, since  $\mathbf{L}_l$  is applied to functions in  $V^L$  for  $L > l$  and therefore  $\mathbf{L}_l$  and  $\mathbf{L}_l^1$  lead to different restriction operators in the implementation of Algorithm `grid2`. Use of  $\mathbf{L}_l$  amounts to restriction by injection in the language of [11] and is easy and cheap to implement. Use of  $\mathbf{L}_l^1$  would require restriction maps that compute an average but  $\mathbf{L}_m^1 \rightarrow \mathbf{L}$  in norm and therefore the norm convergent version of Algorithm `gridnest` can be used. We define

$$\mathcal{K}_m^1 = \frac{c}{2} \mathbf{L}_m^1.$$

The operator convergence theorem in [26] can be stated as

**THEOREM 3.1.** *Assume that the spatial and angular meshes satisfy Assumptions 3.1 and 3.2. Assume, moreover, that*

$$(3.7) \quad \lim_{m \rightarrow \infty} \bar{h}_m |\log(\bar{\mu}_m)| = 0,$$

where

$$\bar{\mu}_m = \min_j |\mu_j^m|.$$

Then  $\mathbf{L}_m^1 \rightarrow \mathbf{L}$  and  $\mathcal{K}_m^1 \rightarrow \mathcal{K}$  in the operator norm in  $C[0, \tau]$ .

A consequence of Theorem 3.1 and (3.6) is the strong convergence result.

**THEOREM 3.2.** *Assume that the spatial and angular meshes satisfy Assumptions 3.1 and 3.2 and that (3.7) holds. Then the sequence  $\mathcal{K}_m$  is strongly convergent to  $\mathcal{K}$  and collectively compact. Moreover,*

$$g_m = \frac{1}{2} \mathbf{T}_m^1(0) \rightarrow g$$

as  $m \rightarrow \infty$ .

*Proof.* The uniform convergence of  $\{\mathbf{L}_m^1\}$  to the compact operator  $\mathbf{L}$  implies that the sequence  $\{\mathbf{L}_m^1\}$  is collectively compact. Since  $P_m$  converges strongly to the identity and has norm 1 in  $C[0, \tau]$ , the sequence

$$\mathbf{L}_m = \mathbf{L}_m^1 P_m$$

converges strongly to  $\mathbf{L}$  and is collectively compact because the sequence  $\{\mathbf{L}_m^1\}$  is. Since  $\mathcal{K}_m$  and  $\mathcal{K}$  differ from  $\mathbf{L}_m$  and  $\mathbf{L}$  only by multiplication by the continuous function  $c$ , the sequence  $\mathcal{K}_m$  is strongly convergent to  $\mathcal{K}$  and collectively compact as asserted.

The final assertion on convergence of  $g_m$  follows from properties of the quadrature rules and continuity of  $q$ ,  $F_l$ , and  $F_r$ .  $\square$

In [26] error estimates are derived in terms of the number of spatial and angular mesh points. Let  $\phi_m = (I - \mathcal{K}_m)^{-1}g_m$  and  $\phi^* = (I - \mathcal{K})^{-1}g$ . Then if  $c, q \in C^{4+\epsilon}$  for some  $\epsilon > 0$ , the hypotheses of Theorem 3.2 hold, and double Gaussian quadratures on  $[-1, 0] \cup [0, 1]$  are used we have

$$\|\phi_m - \phi^*\|_\infty = O((N_m^A)^{-2} + |\log \bar{h}_m(N_m^A)| \bar{h}_m^2(N_m^A)^2)$$

indicating, as pointed out in [26] that  $N_m^S \approx (N_m^A)^2$  is optimal if the log terms are neglected. If Gaussian quadratures on  $[-1, 1]$  are used and  $c, q \in C^{2+\epsilon}$  then

$$\|\phi_m - \phi^*\|_\infty = O((N_m^A)^{-1} + |\log \bar{h}_m(N_m^A)| \bar{h}_m^2 N_m^A)$$

leading to an optimal choice of  $N_m^S \approx N_m^A$ . If  $c$  and  $q$  are piecewise smooth, similar estimates hold provided any discontinuity of  $q$  or  $c$  is a spatial mesh point.

As we remarked above, if the angular mesh is not refined, the collective compactness results in [22] apply to the maps  $\mathcal{K}_m$  given by (2.9), and to certain higher order schemes as well, if the spatial mesh satisfies Assumption 3.1.

A consequence of the norm convergence of  $\mathcal{K}_m^1$  and the strong convergence of the collectively compact sequence  $\{\mathcal{K}_m\}$  to  $\mathcal{K}$  is

**THEOREM 3.3.** *Assume that the spatial and angular meshes satisfy Assumptions 3.1 and 3.2 and that (3.7) holds. Then for any  $\rho \in (0, 1)$  there is  $l_0$  such that for all  $L$  and  $l$  such that  $L \geq l \geq l_0$*

$$\|I - (I - \mathcal{K}_l^1)^{-1}(I - \mathcal{K}_L)\| \leq \rho.$$

*In particular, if  $l \geq l_0$  and*

$$B_l^L = (I - \mathcal{K}_l^1)^{-1}$$

*algorithm grid2 will converge.*

If  $c$  or  $q$  have finitely many discontinuities and any discontinuity of either is a spatial mesh point, then the results above remain true if  $C[0, \tau]$  is replaced by the space of piecewise continuous functions that have discontinuities only at the discontinuities of  $q$ .

We close this section with a more detailed description of the differences between  $\mathcal{K}_m^1$  and  $\mathcal{K}_m$ . For constant  $c$  (or piecewise constant  $c$  if any discontinuity of  $c$  is a spatial mesh point of all grids), the two maps agree on  $V^m$ , the

space of piecewise linear functions with nodes at  $\{x_i^m\}$ . Hence the *fine mesh* maps (when  $m = L$ ) in either formulation are the same. The **coarse mesh** maps ( $m = l$ ), however, agree only on  $V^l$ . When  $\mathcal{K}_l$  is applied to a continuous function  $f \notin V^l$ , one evaluates  $f$  (and  $c$  and  $q$ , of course) at the coarse mesh points and applies formulae (3.1), (3.2), and (3.3) directly. The application of  $\mathcal{K}_l^1$ , however, to a continuous function  $f \notin V^l$ , replaces the means

$$\frac{(cf)(x_i^l) + (cf)(x_{i-1}^l)}{2} \text{ and } \frac{(cf)(x_i^l) + (cf)(x_{i+1}^l)}{2}$$

that arise in (3.1) and (3.2) with the integral averages

$$(3.8) \quad \frac{1}{m(\mathcal{I}_{i-1}^l)} \int_{\mathcal{I}_{i-1}^l} c(y)f(y) dy \text{ and } \frac{1}{m(\mathcal{I}_i^l)} \int_{\mathcal{I}_i^l} c(y)f(y) dy$$

respectively. Here  $m(I)$  denotes the length of the interval  $I$  and

$$\mathcal{I}_i^l = (x_i^l, x_{i+1}^l).$$

If  $c$  is not (piecewise) constant, evaluation of  $\mathcal{K}_m$  still uses the formulae (3.1), (3.2), and (3.3).  $\mathcal{K}_m(u)$  and  $\mathcal{K}_m^1(u)$  agree if  $cu \in V^m$ , which is not likely, however we only use  $\mathcal{K}_L$  on the fine mesh and use  $\mathcal{K}_l^1$  to form an approximate inverse. To evaluate  $\mathcal{K}_l^1$  on a function  $u \in C[0, \tau]$  we would use (3.1), (3.2), and (3.3) with the averages replaced by those from (3.8). A more convenient approach, which still preserves the norm convergence result in Theorem 3.3 is to replace the averages in (3.8) by

$$(3.9) \quad \frac{c(x_{i-1}^l) + c(x_i^l)}{2} \frac{1}{m(\mathcal{I}_{i-1}^l)} \int_{\mathcal{I}_{i-1}^l} f(y) dy \text{ and } \frac{c(x_{i+1}^l) + c(x_i^l)}{2} \frac{1}{m(\mathcal{I}_i^l)} \int_{\mathcal{I}_i^l} f(y) dy$$

We used (3.9) in the numerical results reported in § 4. Note that if (3.9) is used, the evaluation of the coarse mesh operator on an arbitrary continuous function requires evaluation of  $c$  and  $q$  at coarse mesh points only. Also note that use of (3.9) implies that the coarse mesh solution will be slightly different than if (3.8) is used. We see this in the numerical results in different convergence statistics in the example where  $c$  is not piecewise constant.

Having described the evaluation of the coarse mesh map, we turn to the solution of the coarse mesh equation. In Algorithm `grid2` with  $B_l^L$  given by (2.3) the application of the approximate inverse requires solution of equations of the form

$$(3.10) \quad v - \mathcal{K}_l v = r$$

where  $r \in V^L$ . To solve (3.10) we find  $\bar{v} \in V^l$  such that

$$(3.11) \quad \bar{v} - \mathcal{K}_l \bar{v} = I_l^l r$$

where  $I_L^l : V^L \rightarrow V^l$  is an intergrid transfer map, piecewise linear interpolation is typical and if the grids are nested, this is simple injection. The important properties of  $I_L^l$  are that

$$(I_L^l)^2 = I_L^l \text{ and } \mathcal{K}_l v = \mathcal{K}_l I_L^l v.$$

Having found  $\bar{v}$  we recover  $v$  by Nyström interpolation

$$(3.12) \quad v = r + \mathcal{K}_l \bar{v}.$$

If we use  $\mathcal{K}_l^1$  to form  $B_l^L$ , we must solve coarse mesh equations of the form,

$$(3.13) \quad v - \mathcal{K}_l^1 v = r.$$

The analog of (3.11) is

$$(3.14) \quad \bar{v} - I_l \mathcal{K}_l^1 \bar{v} = I_l r.$$

In (3.14),  $I_l$  is the projection of  $C[0, \tau]$  onto the space of functions  $PC^l$  that are constant on the intervals  $\{(x_i^l, x_{i+1}^l)\}$ .

$$(I_l u)(x) = \frac{1}{m(\mathcal{I}_i)} \int_{\mathcal{I}_i} u(y) dy \text{ for } x \in \mathcal{I}_i.$$

After we find  $\bar{v} \in PC^l$ , we recover  $v \in V^l$  via (3.12) as before. Note that if the grids are nested and  $v \in V^L$ , then the integral over  $\mathcal{I}_i$  can be computed exactly with the trapezoid rule.

**4. Computational Results.** In the examples reported in this section we tabulate the performance of Algorithm **gridnest** by giving, for each mesh level, the number of unknowns  $N$ , the iteration counter at that mesh level  $i$ , the number of GMRES iterates  $i_G$  needed to satisfy (2.12), the norm of the residual  $R_i$ , and, when  $i = 1$  the ratio of the norms of the residual on entry and on exit. In all cases only one iterate was required at each mesh level. We consider both the norm convergent and strongly convergent formulations of the diamond difference discrete ordinates method and use double Gauss quadratures for the angular mesh. While, as expected, the norm convergent version performs best, the strongly convergent version is of use in situations where a norm convergent family of approximations is not available.

We use a modification of the Brown-Hindmarsh GMRES code [7] with changes made in the inner product to allow the approximate  $L^2$  inner product to be used instead of the  $R^N$  inner product. The coarse mesh problem was solved with GMRES and the outer iteration was terminated at the coarse level when the norm of the residual was less than  $10^{-3}$ . The problem-method combinations use different values of the parameter  $\eta$ . We selected  $\eta$  and the coarse mesh width to be as large as possible and still require one outer iterate for each mesh level and obtain good accuracy in the final result.

The first example is (1.1) with

$$c(x) = \omega e^{-x/s}.$$

For values of  $\tau$  near  $\infty$ ,  $\omega$  near 1, and  $s$  near  $\infty$ , the spectral radius of the unpreconditioned source iteration operator is near one. Such problems are more difficult and we report on one such computation as well as an easier one. For all problems we used  $F_l(\mu) = 1, \mu > 0$   $F_r(\mu) = 0, \mu < 0$  as data. We used uniform spatial meshes. The meshes were arranged so that  $2^p + 1$  points were in each spatial mesh for some  $p$ , with  $p$  begin incremented by one as the meshes were refined. The parameter  $\eta$  in (2.12) was set to .1. The tolerances  $\{\epsilon_i\}$  were given by

$$(4.1) \quad \epsilon_0 = 10^{-3}, \epsilon_i = .25 * \|f - \mathbf{K}_{i-1}(f)\|$$

reflecting the expected [22] second order accuracy of the diamond differencing. We consider two problems, one with  $\tau = 5$ ,  $\omega = .5$ , and  $s = 1$ , which should be simple for most algorithms, and one with  $\tau = 100$ ,  $\omega = 1$ , and  $s = \infty$ , which is more difficult. A fixed angular mesh of double 10 point Gaussian quadratures was used.

We report on the performance of the results by tabulating, for each problem and algorithm combination, the number of unknowns  $N$  at each level, the iteration counter  $i$  for that level (which is 0 or 1 if the termination criterion Algorithm `grid2` is satisfied after the first iterate), the number of GMRES iterations  $i_g$  needed to satisfy (2.12) for the computation of the approximate inverse, the  $l^\infty$  norm of the residual,  $R_i$ , and the ratio  $R_i/R_{i-1}$  for the terminal iterate at each level. Note that there is very little variation in  $i_g$  and  $R_i/R_{i-1}$  as the mesh is refined.

Comparison between Tables 4.1 and 4.2 indicates that the performance of the two algorithms, in terms of the iteration counter  $i$ , is very similar. The much lower cost of the norm convergent method give it an advantage when it can be applied. Note also that, as we remarked in the previous section, the performance of the two methods on the start-up phase of the iteration on the coarsest mesh differs because  $c$  is not piecewise constant.

In the Tables 4.3 and 4.4 we report on the performance for a more difficult case. For the  $\tau = 100$ ,  $\omega = 1$  case the number of coarse mesh points was increased to 65 and  $\eta$  reduced to  $10^{-2}$  in order to keep  $i$  bounded by 1 for the finer meshes. Note that this is still an extremely coarse mesh since  $\tau = 100$ .

Our second example is a multi-region problem with  $\tau = 8$  and

$$c(x) = \begin{cases} .25 & 0 < x < 2 \\ 1 & 2 < x < 6 \\ .5 & 6 < x < 8 \end{cases}$$

We report our results in Tables 4.5 and 4.6 in the same format as in the previous example. We used  $F_l(\mu) = 1, \mu > 0$   $F_r(\mu) = 0, \mu < 0$  as data and the same meshes as in the previous example. We used  $\eta = .1$  in the computations.

TABLE 4.1  
 $\tau = 5, \omega = .5, s = 1$ , Norm Convergent Approximation.

$N$	$i$	$i_g$	$R_i$	$R_i/R_{i-1}$
33	0		0.50D+00	
	1	1	0.51D-01	0.10D+00
	2	1	0.26D-02	0.52D-01
	3	2	0.87D-05	0.33D-02
65	0		0.17D-01	
	1	1	0.25D-03	0.15D-01
129	0		0.92D-02	
	1	1	0.12D-03	0.14D-01
257	0		0.43D-02	
	1	1	0.33D-04	0.78D-02
513	0		0.17D-02	
	1	1	0.85D-05	0.49D-02
1025	0		0.60D-03	
	1	1	0.21D-05	0.35D-02
2049	0		0.18D-03	
	1	1	0.51D-06	0.28D-02
4097	0		0.51D-04	
	1	1	0.13D-06	0.25D-02
8193	0		0.14D-04	
	1	1	0.31D-07	0.23D-02

The computations for these first two examples were done on a SUN SPARC 1+ workstation running SUN OS version 4.1.1 and FORTRAN compiler f77 version 1.3.1.

The final example is a six group anisotropic problem taken from [28]. The transport equation is

$$(4.2) \quad \mu \frac{\partial \psi}{\partial x}(x, \mu) + \Sigma \psi(x, \mu) = \frac{1}{2} \int_{-1}^1 C(\mu, \mu') \psi(x, \mu') d\mu' + q(x, \mu).$$

In (4.2)  $\Sigma$  is a given  $6 \times 6$  diagonal matrix, the source term  $q(x, \mu)$  has an expansion

$$q(x, \mu) = \sum_{i=0}^3 P_i(\mu) q_i(x),$$

and  $C(\mu, \mu')$  is a  $6 \times 6$  matrix-valued function of the form

$$C(\mu, \mu') = \sum_{i=0}^3 P_i(\mu) P_i(\mu') C_i$$

where the  $6 \times 6$  matrix coefficients  $\{C_i\}_{i=0}^3$  are given. The unknown  $\psi$  is an  $R^6$  valued function. For this problem  $\tau = 45.156$ , a very large value and the

TABLE 4.2  
 $\tau = 5, \omega = .5, s = 1$ , *Strong Convergent Approximation.*

$N$	$i$	$i_g$	$R_i$	$R_i/R_{i-1}$
33	0		0.50D+00	
	1	1	0.51D-01	0.10D+00
	2	1	0.24D-02	0.47D-01
	3	2	0.10D-04	0.42D-02
65	0		0.18D-01	
	1	2	0.31D-03	0.17D-01
129	0		0.91D-02	
	1	1	0.88D-04	0.96D-02
257	0		0.42D-02	
	1	2	0.25D-04	0.59D-02
513	0		0.17D-02	
	1	2	0.53D-05	0.31D-02
1025	0		0.59D-03	
	1	2	0.13D-05	0.22D-02
2049	0		0.18D-03	
	1	2	0.32D-06	0.18D-02
4097	0		0.50D-04	
	1	2	0.81D-07	0.16D-02
8193	0		0.13D-04	
	1	2	0.20D-07	0.15D-02

matrices  $C_i$  are full, not triangular as is usually the case. This upscattering precludes solution as a triangular system and the full system of transport equations must be solved. Boundary conditions are

$$F_l(\mu) = (1, 0, 0, 0, 0, 0)^T, F_r(\mu) = (0, 0, 0, 0, 0, 0)^T.$$

The parameter  $\eta$  in (2.12) was set to  $10^{-5}$ . The tolerances  $\{\epsilon_i\}$  were given by (4.1) as before.

The implementation of source iteration for such a problem is an extension of that suggested in [26]. We take as our unknowns the four  $R^6$  valued functions

$$f_i(x) = \int_{-1}^1 P_i(\mu') \psi(x, \mu') d\mu'$$

resulting in a total of 24 unknown scalar functions. Treating the sequence  $\{f_i\}$  as given we may compute the right hand side of (4.2) and perform forward and backward integrations in each group. These integrations may be done independently since  $\Sigma$  is diagonal. This adds a second level of parallelism. On the coarse mesh there were 8 angular mesh points, corresponding to double 4 point Gaussian quadratures and 257 spatial mesh points leading to  $257 \times$

TABLE 4.3  
 $\tau = 100, \omega = 1, s = \infty$ , *Norm Convergent Approximation.*

$N$	$i$	$i_g$	$R_i$	$R_i/R_{i-1}$
65	0		0.50D+00	
	1	20	0.16D-02	0.33D-02
	2	117	0.11D-04	0.66D-02
129	0		0.59D-03	
	1	3	0.15D-03	0.26D+00
	2	40	0.55D-04	0.36D+00
257	0		0.41D-03	
	1	28	0.66D-04	0.16D+00
513	0		0.27D-03	
	1	39	0.38D-04	0.14D+00
1025	0		0.17D-03	
	1	27	0.16D-04	0.94D-01
2049	0		0.90D-04	
	1	34	0.60D-05	0.66D-01
4097	0		0.44D-04	
	1	34	0.20D-05	0.46D-01
8193	0		0.19D-04	
	1	35	0.80D-06	0.42D-01

TABLE 4.4  
 $\tau = 100, \omega = 1, s = \infty$ , *Strong Convergent Approximation.*

$N$	$i$	$i_g$	$R_i$	$R_i/R_{i-1}$
65	0		0.50D+00	
	1	20	0.16D-02	0.33D-02
	2	117	0.11D-04	0.66D-02
129	0		0.59D-03	
	1	64	0.10D-03	0.18D+00
257	0		0.39D-03	
	1	40	0.86D-04	0.22D+00
513	0		0.27D-03	
	1	69	0.27D-04	0.10D+00
1025	0		0.15D-03	
	1	83	0.17D-04	0.12D+00
2049	0		0.79D-04	
	1	67	0.95D-05	0.12D+00
4097	0		0.40D-04	
	1	58	0.54D-05	0.14D+00
8193	0		0.17D-04	
	1	69	0.30D-05	0.17D+00

TABLE 4.5  
*Multi-region Problem: Norm Convergent Approximation.*

$N$	$i$	$i_g$	$R_i$	$R_i/R_{i-1}$
33	0		0.50D+00	
	1	2	0.90D-02	0.18D-01
	2	5	0.19D-03	0.21D-01
65	0		0.34D-01	
	1	1	0.19D-02	0.55D-01
129	0		0.18D-01	
	1	4	0.16D-02	0.89D-01
257	0		0.89D-02	
	1	4	0.37D-03	0.41D-01
513	0		0.40D-02	
	1	5	0.27D-03	0.68D-01
1025	0		0.15D-02	
	1	4	0.77D-04	0.50D-01
2049	0		0.51D-03	
	1	5	0.70D-04	0.14D+00
4097	0		0.19D-03	
	1	3	0.15D-04	0.79D-01
8193	0		0.11D-03	
	1	5	0.21D-04	0.19D+00

24 = 6168 unknowns. The finest spatial mesh had 131,073 mesh points and 3,145,752 unknowns. All meshes after the first had 40 angular mesh points corresponding to a double 20 point Gauss rule. A uniform spatial mesh was used.

We computed the fluxes and currents (first angular moments of  $\psi$ ) at several values of  $x$  for comparison with the results obtained by other means in [28]. Our results are in Tables 4.7 and 4.8, we tabulate the group index and values of the fluxes and currents for  $x/\tau = 0, .25, .5, .75, 1$  to the same degree of precision that was reported in [28].

Our results agree completely with those in [28] with the exception of the flux in group 1 at  $x/\tau = .5$ , where the result in [28] was  $.93942e - 13$  which differs from our result by one digit in the last place. When the spatial or angular mesh was coarsened or the value of  $\eta$  was increased, the results were not as good, but values of  $\eta$  as large as  $10^{-3}$  gave results which agreed with those in [28] to a relative accuracy, in terms of the norm of the flux vector, of  $10^{-5}$ , roughly what an examination of the  $i = 0$  entries of Tables 4.9 and 4.10 would predict. Larger values of  $\eta$  did not change the  $i = 0$  entries of Tables 4.9 and 4.10. The results also agree to one digit in the last place, even in entries that are several orders of magnitude smaller than the norm of the solution.

TABLE 4.6  
*Multi-region Problem: Strong Convergent Approximation.*

$N$	$i$	$i_g$	$R_i$	$R_i/R_{i-1}$
33	0		0.50D+00	
	1	2	0.90D-02	0.18D-01
	2	5	0.19D-03	0.21D-01
65	0		0.34D-01	
	1	5	0.94D-03	0.27D-01
129	0		0.18D-01	
	1	5	0.57D-03	0.32D-01
257	0		0.88D-02	
	1	5	0.32D-03	0.36D-01
513	0		0.39D-02	
	1	5	0.17D-03	0.44D-01
1025	0		0.15D-02	
	1	5	0.89D-04	0.59D-01
2049	0		0.51D-03	
	1	5	0.45D-04	0.89D-01
4097	0		0.21D-03	
	1	5	0.23D-04	0.11D+00
8193	0		0.11D-03	
	1	5	0.11D-04	0.11D+00

TABLE 4.7  
*Group Fluxes.*

Group	$x/\tau = 0$	$x/\tau = .25$	$x/\tau = .5$	$x/\tau = .75$	$x/\tau = 1$
1	.109e+1	.16205e-3	.48524e-7	.14567e-10	.4e-14
2	.230e+0	.37447e-1	.19639e-2	.10277e-3	.179e-5
3	.292e+0	.18547e+0	.97989e-2	.51278e-3	.437e-5
4	.306e-1	.26281e-1	.13884e-2	.72654e-4	.439e-6
5	.600e-3	.61078e-3	.32260e-4	.16882e-5	.791e-8
6	.731e-5	.72593e-5	.38325e-6	.20056e-7	.794e-10

TABLE 4.8  
*Group Currents.*

Group	$x/\tau = 0$	$x/\tau = .25$	$x/\tau = .5$	$x/\tau = .75$	$x/\tau = 1$
1	.47192e+0	.10441e-3	.31293e-7	.93943e-11	.3e-14
2	-.99291e-1	.59818e-2	.31144e-3	.16359e-4	.10985e-5
3	-.16297e+0	.87772e-2	.46933e-3	.24656e-4	.25198e-5
4	-.17026e-1	.59229e-3	.31693e-4	.16650e-5	.24654e-6
5	-.32783e-3	.67184e-5	.35924e-6	.18873e-7	.43167e-8
6	-.38625e-5	.28826e-7	.15337e-8	.80572e-10	.41041e-10

TABLE 4.9  
*Multi-group Problem, Norm Convergent Approach.*

$N$	$i$	$i_g$	$R_i$	$R_i/R_{i-1}$
6168	0		0.10e+01	
	1	66	0.16e-05	0.16e-05
12312	0		0.37e-01	
	1	15	0.15e-02	0.41e-01
24600	0		0.20e-01	
	1	28	0.56e-03	0.29e-01
49176	0		0.10e-01	
	1	29	0.20e-03	0.20e-01
98328	0		0.51e-02	
	1	32	0.61e-04	0.12e-01
196632	0		0.24e-02	
	1	32	0.18e-04	0.74e-02
393240	0		0.98e-03	
	1	31	0.49e-05	0.50e-02
786456	0		0.35e-03	
	1	31	0.13e-05	0.38e-02
1572888	0		0.11e-03	
	1	30	0.34e-06	0.32e-02
3145752	0		0.33e-04	
	1	30	0.84e-07	0.26e-02

The method of [28] takes advantage of the fact that  $\Sigma$  and  $C$  are independent of  $x$ , features not required by our approach, and provide accurate results that we used for testing. We point out that computations using coarser spatial or angular meshes did not agree as well with the results in [28]. This is why the very fine grids and the small value of  $\eta$  were used.

In the table we give the number of unknowns  $N$  at each level, the iteration counter  $i$  for that level (which is 0 or 1 as the termination criterion Algorithm `grid2` is satisfied after the first iterate), the number of GMRES iterations  $i_g$  needed to satisfy (2.12) for the computation of the approximate inverse, the  $l^\infty$  norm of the residual,  $R_i$ , and the ratio  $R_i/R_{i-1}$  for the terminal iterate at each level. Note that there is very little variation in  $i_g$  and  $R_i/R_{i-1}$  as the mesh is refined. GMRES was restarted after a Krylov space of dimension 50 had been built.

We conclude this section with a discussion of how we exploited the natural parallelism of the source iteration map in an implementation on the KSR1 computer. We concentrate on the third of the examples, as the computational effort for the monoenergetic computations in the first two examples was very low. We begin with a brief summary of the KSR1 architecture and software

TABLE 4.10  
*Multi-group Problem, Strong Convergent Approach.*

$N$	$i$	$i_g$	$R_i$	$R_i/R_{i-1}$
6168	0		0.10e+01	
	1	66	0.16e-05	0.16e-05
12312	0		0.37e-01	
	1	47	0.15e-02	0.41e-01
24600	0		0.20e-01	
	1	70	0.52e-03	0.27e-01
49176	0		0.99e-02	
	1	76	0.19e-03	0.19e-01
98328	0		0.50e-02	
	1	73	0.61e-04	0.12e-01
196632	0		0.24e-02	
	1	71	0.18e-04	0.77e-02
393240	0		0.97e-03	
	1	68	0.49e-05	0.51e-02
786456	0		0.34e-03	
	1	66	0.15e-05	0.42e-02
1572888	0		0.11e-03	
	1	66	0.41e-06	0.38e-02
3145752	0		0.32e-04	
	1	65	0.11e-06	0.34e-02

as described in [18], [19], and [17]. The KSR1 at the North Carolina Supercomputing Center is a ring with 32 cells, each cell having custom chips for processing units, connection units, a sub-cache with .25MB for data and .25MB for instructions, and a local cache of 32 megabytes of memory. The memory is entirely organized as cache, which allows one to treat the physically distributed memory as having a single address space and program using a shared memory model. Cache misses cause searches over the ring and transfer of copies of the missed pages to the local cache in the cell that requests the page. This memory organization is called ALLCACHE by KSR. The KSR architecture allows level 0 rings such as the one described above to be grouped in higher level 1 rings. A level 1 ring may contain up to 34 level 0 rings. A full level 0 ring has 1 GB of memory. The computations in this paper were done on the KSR1 at the North Carolina Supercomputing Center. This machine runs KSR OS R1.1.1 and KSR Fortran compiler version 1.0. All of the computations for the third example were done on this computer.

The KSR Fortran compiler has several directives which exploit parallelism. The one used in the computations reported in this paper was the `parallel region` construct. By using this one may send several copies of a block of code

FIG. 4.1. *Parallel Code Fragment*

```

c*ksr* parallel region (teamid=team_id,numthreads=loc_threads,
c*ksr*& private=(ipro))
    ipro=ipr_mid()
    if(mod(ipro,2).eq.0) then
        call bsweep(u,n,1+ipro/2)
    else
        call fsweep(u,n,1+(ipro-1)/2)
    endif
c*ksr* end parallel region

```

(a region) to different processors with different data. In our case, the block called a subroutine which performed spatial integration for different values of angle (and group in the multigroup case). Data that is to be shared between the different copies of the routine are passed through common blocks. When the code runs, a “team” of threads is created to execute the parallel region. Having done that the source iteration loop is the simple block in Figure 4.1.

The arguments to the `parallel region` command are an identifier for the team of threads, the number of processes or threads that will execute the block of code, and the names of private variables. Private variables are not shared among the processors. The private variable `ipro` is the number of the thread and is used to partition the spatial integrations in angle and group. In the block of code, `ipro` is set to the process number and either a backward (`call bsweep`) or forward (`call fsweep`) integration is done depending on the parity of `ipro`, this is the basic splitting of the computation into angles for which  $\mu < 0$  and for which  $\mu > 0$ . Within the forward and backward integrations, further splitting is done if the number of threads `loc_threads` is greater than two. in angle. For the multigroup case, Splittings in both group and angle were done for 1, 2, 4, 6, 12, 24 processors. For 2 or more processors the forward and backward integrations were done in parallel. For 24 processors the angular mesh was split again. For 4, 6, 12, and 24 processors the integrations for each group were done with as much parallelism as the number of processors and the splitting of the angular mesh would allow. For example, for 24 processors the angular mesh was split twice and all six groups were integrated in space simultaneously, thereby dividing the computation into 24 parts.

Other parts of the code were parallelized, but the effect of that effort was minimal as the evaluations of  $\mathbf{K}$  dominate the computational load.

Tables 4.11 and 4.12 in the norm and strong convergent cases illustrate the parallel performance and the degree to which evaluations of  $\mathbf{K}$  dominate the computation. We tabulate the number of processors  $N$ , total run times  $T$ , time spent in the  $\mathbf{K}$  evaluation routine  $T_K$ , the percentage of time spent in the  $\mathbf{K}$  routine  $P_K$ , the parallel percentage efficiency of the total code  $E_T$ , and

TABLE 4.11  
*Multi-group Problem, Norm Convergent Method.*

$N$	$T$	$T_K$	$P_K$	$E_T$	$E_K$
1	1226	1041	84.8	100.0	100.0
2	749	563	75.2	81.8	92.3
4	467	362	77.4	65.6	71.9
6	406	299	73.6	50.4	58.1
12	345	240	69.6	29.6	36.1
24	267	199	74.5	19.1	21.8

TABLE 4.12  
*Multi-group Problem, Strong Convergent Method.*

$N$	$T$	$T_K$	$P_K$	$E_T$	$E_K$
1	1770	1511	85.3	100.0	100.0
2	1093	834	76.2	81.0	90.6
4	732	561	76.5	60.4	67.4
6	634	460	72.5	46.5	54.7
12	540	367	68.0	27.3	34.3
24	453	318	70.2	16.3	19.8

the parallel percentage efficiency of  $\mathbf{K}$  routine. Here, for a given  $N$ ,

$$E_T = 100 \frac{\text{Total time for } N = 1}{NT} \quad \text{and} \quad E_K = 100 \frac{\text{Time for } \mathbf{K} \text{ for } N = 1}{NT_K}.$$

All times are given in seconds.

As one can see from Tables 4.11 and 4.12 the single processor time for the norm convergent method is about 2/3 that of the strong convergent method. Since we test the termination criterion before refining the mesh, this is what the cost analysis in § 2 would predict. The efficiency for 12 processors is roughly 30% for both methods, which we view as acceptable. As one would expect, the efficiency falls off as the number of processors increase. For finer angular meshes, this effect is less pronounced because the processors have more work to do. Based on the analysis in [26] and our observations, an 800 point Gauss rule should be expected to give roughly the same accuracy as the double 20 point rule that we used in the computations. We performed such a computation to see what the effect would be on efficiency. We obtained a 24 processor efficiency of  $E_T = 32\%$  for the strong convergent method, but, as Table 4.13 indicates, the run times were longer.

Even in the single processor runs, however, we use almost the full memory of the KSR1. The computations reported here require the storage of roughly 15 for the iteration and application of the approximate inverse and roughly one fine mesh vector per processor for the evaluation of the action of  $\mathcal{K}$  on a vector.

TABLE 4.13  
*Strong Convergent Method, 800 point Gauss rule.*

$N$	$T$	$T_K$	$P_K$	$E_T$	$E_K$
1	4396	4173	94.9	100.0	100.0
2	2450	2227	90.9	89.7	93.7
4	1384	1240	89.6	79.4	84.1
6	1059	914	86.3	69.2	76.1
12	774	628	81.1	47.3	55.4
24	558	445	79.8	32.8	39.1

**Acknowledgments.** The author would like to thank Greg Byrd and Eric Sills of the North Carolina Supercomputing Center for their help with the use of the Kendall Square KSR1 computer, and Chuck Siewert of North Carolina State University and Steve Wright of Argonne National Laboratory for many helpful conversations.

## REFERENCES

- [1] B. T. ADAMS AND J. E. MOREL, *An upscatter acceleration scheme for the multigroup  $S_n$  equations*. preprint.
- [2] P. M. ANSELONE, *Collectively Compact Operator Approximation Theory*, Prentice-Hall, Englewood Cliffs, NJ, 1971.
- [3] S. F. ASHBY, P. N. BROWN, M. C. DORR, AND A. C. HINDMARSH, *A linear algebraic analysis of diffusion synthetic acceleration for the Boltzmann transport equation*, Tech. Report UCRL-JC-110712, Lawrence Livermore National Laboratory, 1992.
- [4] K. E. ATKINSON, *Iterative variants of the Nyström method for the numerical solution of integral equations*, Numer. Math., 22 (1973), pp. 17–31.
- [5] ———, *A survey of numerical methods for solving nonlinear integral equations*, Journal of Integral Equations and Applications, 4 (1992), pp. 15–46.
- [6] H. BRAKHAGE, *Über die numerische Behandlung von Integralgleichungen nach der Quadraturformelmethode*, Numer. Math., 2 (1960), pp. 183–196.
- [7] P. N. BROWN AND A. C. HINDMARSH, *Reduced storage matrix methods in stiff ODE systems*, Tech. Report UCRL-95088, Lawrence Livermore National Laboratory, 1987.
- [8] I. W. BUSBRIDGE, *The Mathematics of Radiative Transfer*, no. 50 in Cambridge Tracts, Cambridge Univ. Press, Cambridge, 1960.
- [9] S. CHANDRASEKHAR, *Radiative Transfer*, Dover, New York, 1960.
- [10] V. FABER AND T. A. MANTEUFFEL, *A look at transport theory from the point of view of linear algebra*, in Transport Theory, Invariant Imbedding and Linear Algebra, P. N. et al., ed., New York, 1989, Marcel Dekker, pp. 37–61.
- [11] W. HACKBUSCH, *Multi-Grid Methods and Applications*, vol. 4 of Springer Series in Computational Mathematics, Springer-Verlag, New York, 1985.
- [12] ———, *Multigrid methods of the second kind*, in Multigrid Methods for Integral and Differential Equations, Oxford University Press, Oxford, 1985.
- [13] M. HEINKENSCHLOSS, C. T. KELLEY, AND H. T. TRAN, *Fast algorithms for nonsmooth compact fixed point problems*, SIAM J. Numer. Anal., 29 (1992), pp. 1769–1792.
- [14] C. T. KELLEY, *A fast multilevel algorithm for integral equations*. SIAM Journal on

- Numerical Analysis, to appear.
- [15] ———, *Operator prolongation methods for nonlinear equations*, in Computational Solution of Nonlinear Systems of Equations, E. L. Allgower and K. Georg, eds., vol. 26 of AMS Lectures in Applied Mathematics, American Mathematical Society, Providence, RI, 1990, pp. 359–388.
  - [16] C. T. KELLEY AND E. W. SACHS, *Multilevel algorithms for constrained compact fixed point problems*. SIAM J. on Sci. Comp., to appear.
  - [17] KENDALL SQUARE RESEARCH, *Kendall Square Research Technical Summary*, Waltham, MA, 1992.
  - [18] ———, *KSR Parallel Programming*, Waltham, MA, 1992.
  - [19] ———, *Porting Applications to the KSR1*, Waltham, MA, 1992.
  - [20] E. W. LARSEN, *Unconditionally stable diffusion-synthetic acceleration methods for the slab geometry discrete ordinates equations. part I: Theory*, Nucl. Sci. Eng., 82 (1982), pp. 47–63.
  - [21] ———, *Diffusion-synthetic acceleration methods for discrete-ordinates problems*, Trans. Th. Stat. Phys., 13 (1984), pp. 107–126.
  - [22] E. W. LARSEN AND P. NELSON, *Finite difference approximations and superconvergence for the discrete ordinate equations in slab geometry*, SIAM J. Numer. Anal., 19 (1982), pp. 334–348.
  - [23] T. MANTEUFFEL, S. MCCORMICK, J. MOREL, S. OLIVEIRA, AND G. YANG, *Parallel multilevel methods for transport equations*. proceedings of Copper Mountain Conference on Iterative Methods, Copper Mountain, Co., 1992.
  - [24] J. E. MOREL AND T. A. MANTEUFFEL, *An angular multigrid acceleration technique for  $S_n$  equations with highly forward-peaked scattering*, Nuclear Science and Engineering, 107 (1991), pp. 330–342.
  - [25] N. M. NACHTIGAL, S. C. REDDY, AND L. N. TREFETHEN, *How fast are nonsymmetric matrix iterations*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 778–795.
  - [26] J. PITKÄRANTA AND R. SCOTT, *Error estimates for the combined spatial and angular approximations of the transport equation in slab geometry*, SIAM J. Numer. Anal., 20 (1983), pp. 922–950.
  - [27] Y. SAAD AND M. SCHULTZ, *GMRES a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comp., 7 (1986), pp. 856–869.
  - [28] C. E. SIEWERT, *A spherical harmonics method for multi-group or non-grey radiation transport*, J. Quant. Spectrosc. Radiat. Transfer, 49 (1993), pp. 95–106.
  - [29] M. WILLIAMS, *The acceleration of anisotropic transport iterations*, Transport Theory Statist. Phys., (1984), pp. 127–140.