Nuclear Mathematical and Computational Sciences: A Century in Review, A Century Anew
Gatlinburg, Tennessee, April 6-11, 2003, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2003)

UNCOLLIDED-FLUX PRECONDITIONING OF THE CONJUGATE GRADIENTS SOLUTION OF THE TRANSPORT EQUATION

Clif Drumm and Wesley Fan*
Sandia National Laboratories
MS 1166, PO Box 5800, Albuquerque, NM 87185
crdrumm@sandia.gov; wcfan@sandia.gov

ABSTRACT

The CEPTRE code is being developed as a part of the ASCI program at Sandia to model photon-electron transport for radiation effects applications. A second-order transport equation is solved by a unique approach: simultaneous space-direction solution by the method of conjugate gradients (CG). This approach has proven to be very efficient for electron transport. However, for photon transport, convergence of the CG iterations may be unacceptably slow because the space-direction matrix is ill-conditioned for problems containing optically-thin regions, which is often the case for ~100-keV photon transport in miniature electronics. We investigate the use of the uncollided-flux solution as a preconditioner for the transport equation and find that the CG convergence is dramatically improved, demonstrating an order of magnitude or more speedup for typical applications.

Key Words: preconditioning, conjugate gradients, even-odd parity transport

1. INTRODUCTION

The CEPTRE code (**C**oupled **E**lectron-**P**hoton **T**ransport for **R**adiation **E**ffects)[1] [2] is being developed as a part of the ASCI program at Sandia[3]. CEPTRE's primary application is to compute energy- and charge-deposition distributions for radiation effects analysis, such as cable SGEMP (System Generated ElectroMagnetic Pulse). Major features of the code include:

- multigroup energy discretization,
- angular discretization with discrete ordinates, with arbitrary order of anisotropic scattering,
- continuous linear or quadratic finite-element approximations on unstructured mesh,
- second-order forms of the transport equation: even-odd parity flux or Self-Adjoint Angular Flux (SAAF)[4]
- parallel implementation with a spatial domain decomposition,
- object-oriented program architecture to allow ease of maintenance and extension[5].

*Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

The solution method incorporated in CEPTRE is based on a novel approach in which the space-direction dependence is solved simultaneously by using the conjugate gradient (CG) method. This solution technique offers two advantages over the conventional source iteration. First, it eliminates the need for the source iterations, which are notoriously slow for electron transport. Second, the global matrix system is built and processed in a distributed fashion that is well suited for massively parallel computers, allowing the use of existing software to efficiently solve the Symmetric Positive Definite (SPD) system.

This approach works extremely well for electron transport. However, for photons the CG convergence is unacceptably slow, consuming as much as 90% of the runtime for typical applications. The reason for the slow convergence of the photon groups is that the transport matrix becomes ill conditioned as the mean free path (mfp) per mesh cell approaches zero. For a typical SGEMP application, electron total cross sections may be three or four orders of magnitude larger than those for photons. Accurately computing charge balance requires an extremely fine spatial mesh near material interfaces ($\sim 1\mu$m), which corresponds to a very small mfp, $\sigma\Delta \cong 10^{-4}$ for the photons.

One possible solution to this problem would be to use a coarse spatial mesh for the photons and a fine spatial mesh for the electrons. This approach has merit but would be complicated to implement efficiently in parallel for an unstructured mesh. Also, adequately meshing a complex microelectronics component may necessitate using some fairly small cells that are nearly transparent to the photons, resulting in an ill-conditioned matrix regardless.

In this work, we investigate the use of the uncollided-flux as a preconditioner. The uncollided-flux operator contains only the streaming and removal terms of the transport operator. Without the scattering terms, it is simpler to solve because the directions are all decoupled and may be solved independently. Thus, rather than performing many CG iterations on the full space-direction matrix, most of the work is shifted to solving a space-only matrix for each discrete direction. The preconditioner solve is also performed with CG, and because the preconditioner may also be ill-conditioned, convergence of the inner CG iterations may be slow. However, because the runtime of the preconditioner solution scales linearly with the number of directions, and the full space-direction solution scales with the square of the number of directions, the preconditioned solve generally wins out, especially for higher S_n orders.

We present a brief summary of the pertinent features of the CEPTRE code, followed by a description of the uncollided-flux preconditioner. We then present results showing as much as a factor of 50 speedup using the uncollided-flux preconditioner.

2. MATHEMATICAL BASIS OF THE CEPTRE CODE

The CEPTRE code is based on a second-order formulation of the transport equation, either even-odd parity or SAAF, which results in a Symmetric Positive Definite (SPD) operator. CEPTRE is based on a standard finite-elements formulation: 1) construct a weak form of the transport equation by multiplying by a weight function, integrating over the spatial domain and imposing boundary conditions, 2) discretize the domain into finite elements, 3) assemble element integrals to construct a nodal-based matrix representing the approximate solution. We outline the approach here for the even-parity equation as a preliminary to presenting our preconditioning technique.

2.1. Weak Form of the Even-Parity Transport Equation

The even-parity transport equation may be written rather compactly as

$$-\mathcal{T}\mathcal{G}_-^{-1}\mathcal{T}\Phi^+ + \mathcal{G}_+\Phi^+(\mathbf{r},\mathbf{\Omega}) = -\mathcal{T}\mathcal{G}_-^{-1}Q^-(\mathbf{r},\mathbf{\Omega}) + Q^+(\mathbf{r},\mathbf{\Omega}) \text{ for } \mathbf{r}\in V \text{ and } \mathbf{\Omega}\in S^2, \tag{1}$$

where the streaming operator is

$$\mathcal{T}\circ = \mathbf{\Omega}\cdot\nabla_{\mathbf{r}}\circ\,, \tag{2}$$

and even- and odd-parity removal operators are defined by

$$\mathcal{G}_\pm\circ = \sigma_t(\mathbf{r})\circ - \int \sigma_s^\pm(\mathbf{r},\mathbf{\Omega}'\cdot\mathbf{\Omega})\circ d\mathbf{\Omega}'. \tag{3}$$

The even- and odd-parity source terms, $Q^\pm(\mathbf{r},\mathbf{\Omega})$, contain both distributed and downscatter sources. In order to invert the removal operators, it is necessary to expand the scattering cross sections in Legendre polynomials, resulting in the Legendre expansion of the even-parity removal operator

$$\mathcal{G}_+\circ = \sigma_t(\mathbf{r})\circ - \sum_{l\ even}^{\infty}\frac{2l+1}{4\pi}\sigma_{sl}(\mathbf{r})\int P_l(\mu_0)\circ d\mathbf{\Omega}' \tag{4}$$

and the odd-parity inverse operator[6]

$$\mathcal{G}_-^{-1}\circ = \frac{1}{\sigma_t(\mathbf{r})}\left[\circ + \sum_{l\ odd}^{\infty}\frac{2l+1}{4\pi}\frac{\sigma_{sl}(\mathbf{r})}{\sigma_t(\mathbf{r})-\sigma_{sl}(\mathbf{r})}\int P_l(\mu_0)\circ d\mathbf{\Omega}'\right]. \tag{5}$$

The weak form is obtained, after some manipulation[7], by multiplying by a weight function, $u(\mathbf{r})$, integrating over space and imposing vacuum (natural) boundary conditions

$$\left\langle \mathcal{T}u(\mathbf{r})\mathcal{G}_-^{-1}\mathcal{T}\Phi^+\right\rangle + \left\langle u(\mathbf{r})\mathcal{G}_+\Phi^+(\mathbf{r},\mathbf{\Omega})\right\rangle + \oint u(\mathbf{r})\Phi^+(\mathbf{r},\mathbf{\Omega})\,|\mathbf{\Omega}\cdot\mathbf{n}|\,d\mathbf{s}$$

$$= \left\langle \mathcal{T}u(\mathbf{r})\mathcal{G}_-^{-1}Q^-(\mathbf{r},\mathbf{\Omega})\right\rangle + \left\langle u(\mathbf{r})Q^+(\mathbf{r},\mathbf{\Omega})\right\rangle + \oint u(\mathbf{r})\Phi_b(\mathbf{r},\pm\mathbf{\Omega})\,|\mathbf{\Omega}\cdot\mathbf{n}|d\mathbf{s},\ \mathbf{\Omega}\cdot\mathbf{n}\lesseqgtr 0. \tag{6}$$

Derivation of the odd-parity weak form is similar[7] and will not be elaborated here.

2.2. Development of the Linear System for Simultaneous Space-Direction Solution

The result of the finite-elements discretization described in the previous section is a large SPD linear system,

$$A\mathbf{x} = \mathbf{b}, \tag{7}$$

which can be solved with a parallel CG algorithm.

With directions grouped together, the resulting matrix has a sparse-block structure, with the number of block rows equal to the number of nodes in the finite-elements mesh. The blocks are square, with size equal to the number of directions in the discrete-ordinates quadrature. The blocks are full, because the scattering

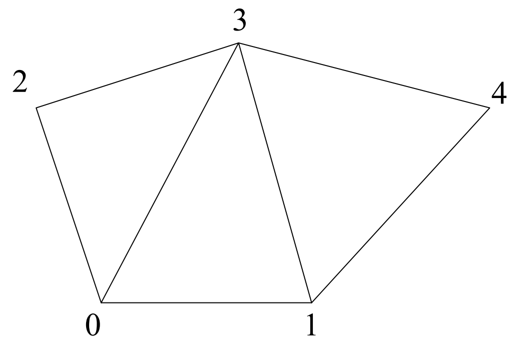

Figure 1. Simple Linear-Triangular Finite-Element Mesh

term couples all of the directions together. This sparse-block structure is tailor made for the Variable Block Row (VBR) data format in the Aztec code[8], which is a library of parallel Krylov iterative solvers.

In order to visualize the structure of the matrix, consider the simple three-element mesh shown in Fig. 1. For a quadrature with four discrete directions and the five-node spatial mesh illustrated in Fig. 1, the matrix non-zero pattern in shown in Fig. 2.

$$A = \left[\begin{array}{cccc|cccc|cccc|cccc|cccc}
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 \\
\hline
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\hline
\bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 \\
\bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 \\
\bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 \\
\bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 \\
\hline
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\hline
0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet
\end{array}\right]$$

Figure 2. Typical CEPTRE Space-Direction Matrix Non-Zero Pattern

2.3. Preconditioned Conjugate Gradients Algorithm

Instead of solving the linear system, $A\mathbf{x} = \mathbf{b}$, the Preconditioned Conjugate Gradients (PCG) algorithm solves a different linear system,

$$M^{-1}A\mathbf{x} = M^{-1}\mathbf{b}, \tag{8}$$

where M is an SPD preconditioner matrix. The convergence of PCG depends on the condition number of $M^{-1}A$, and if M is chosen properly, this linear system will be better conditioned than the original system. After initialization, the PCG algorithm[9] can be summarized as follows,

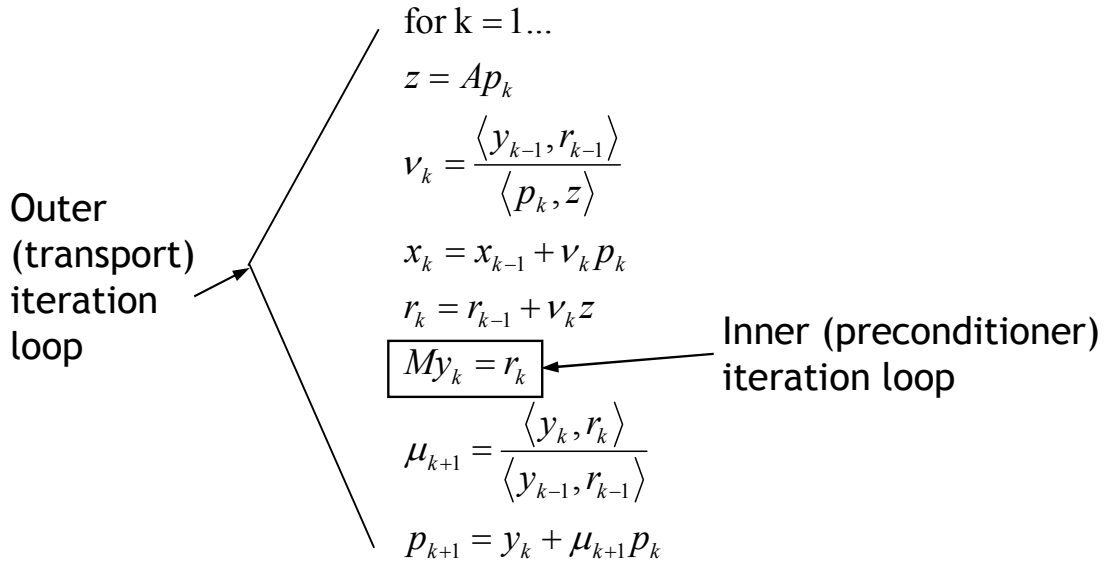

Figure 3. Preconditioned Conjugate Gradients Algorithm

The PCG algorithm contains an outer-iteration loop, for solving the transport equation, and an inner-iteration loop, for solving the preconditioner equation. The terms "outer" and "inner" as used here should not be confused with outer and inner iterations for the conventional source-iteration algorithm. As used here, "inner" refers to the CG iterations to solve the uncollided-flux preconditioner system, not to the conventional source-iteration sweeps.

Another important requirement for an effective preconditioner is that the preconditioner system

$$M\mathbf{y}_k = \mathbf{r}_k, \tag{9}$$

should be much easier to solve than the original system[10]. The uncollided-flux preconditioner, described in the next section, meets these requirements: it is both easier to solve than the original system and greatly reduces the condition number. We use CG to solve the inner (preconditioner) system, however, this is not required. It would be possible, for example, to use a single sweep of a conventional source iteration algorithm to solve the preconditioner system, but we do not investigate that further here.

3. UNCOLLIDED-FLUX PRECONDITIONING

An effective preconditioner should preserve enough of the original operator to significantly speed up the CG convergence, while being much easier to solve than the original system. We have found that the uncollided-flux transport equation, simply formed by eliminating the scattering cross section, provides an effective preconditioner for the transport equation. The uncollided-flux weak form operator, corresponding to Eq. 6 is

$$\mathcal{M} = \left\langle \mathcal{T} u(\mathbf{r}) \frac{1}{\sigma_t} \mathcal{T} \Phi^+ \right\rangle + \left\langle u(\mathbf{r}) \sigma_t \Phi^+(\mathbf{r}, \mathbf{\Omega}) \right\rangle + \oint u(\mathbf{r}) \Phi^+(\mathbf{r}, \mathbf{\Omega}) \left| \mathbf{\Omega} \cdot \mathbf{n} \right| d\mathbf{s} \tag{10}$$

The discrete form of uncollided-flux preconditioner for the simple mesh in the previous section is shown in Fig. 4.

$$M = \left[\begin{array}{cccc|cccc|cccc|cccc|cccc}
\bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 \\
\hline
\bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 \\
0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 \\
0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 \\
0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet \\
\hline
\bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 \\
\hline
\bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 \\
0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 \\
0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 \\
0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet \\
\hline
0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bullet & 0 & 0 & 0 & \bullet
\end{array}\right]$$

Figure 4. Unscattered-Flux Preconditioner Non-Zero Pattern

The uncollided-flux transport equation is much easier to solve, because without scattering, all of the directions are decoupled and may be solved independently. For a given direction, m, the preconditioner is an N_{nodes}xN_{nodes} matrix,

$$M^m = \left[\begin{array}{ccccc}
\bullet & \bullet & \bullet & \bullet & 0 \\
\bullet & \bullet & 0 & \bullet & \bullet \\
\bullet & 0 & \bullet & \bullet & 0 \\
\bullet & \bullet & \bullet & \bullet & \bullet \\
0 & \bullet & 0 & \bullet & \bullet
\end{array}\right].$$

If the space-direction transport matrix is ill-conditioned, the preconditioner matrix, M^m, will also be ill-conditioned, so the convergence of the inner iterations will be slow. However, because the cost of the inner iterations scales linearly with $N_{directions}$, while the cost of the outer iterations scales as $N^2_{directions}$, preconditioning is expected to reduce the overall runtime. Furthermore, it may not be necessary to completely converge the inner iterations to the same tolerance as the outer iterations, as an approximate preconditioned solution may be sufficient to improve the convergence of the outer iterations and be less costly. The unpreconditioned algorithm scales as

$$\sim N^{1.5}_{nodes} N^2_{directions}, \tag{11}$$

while the preconditioned algorithm scales as

$$\sim N^{1.5}_{nodes} N_{directions}, \tag{12}$$

so that the uncollided-flux preconditioner is expected to be more effective for higher S_n orders, as will be demonstrated numerically in the next section.

4. RESULTS

We first investigate a serial implementation of the uncollided-flux preconditioner, where we obtain nearly the theoretical speedups. We then look at a parallel implementation, where we see significant speedups, though less than obtained with the serial implementation for a number of reasons, which we explain.

4.1. Serial Implementation

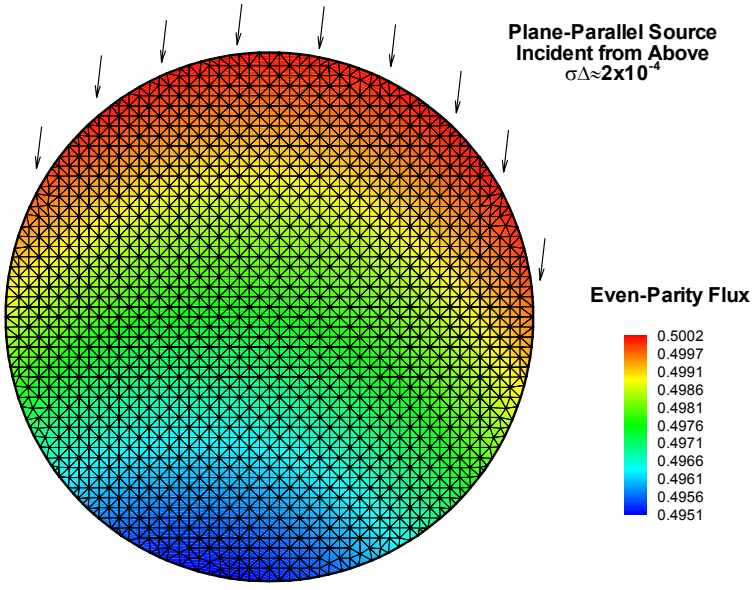

Figure 5. Even-Parity Flux for a Planar Source Incident on a Unit Circle

We have evaluated the serial performance of the uncollided-flux preconditioner using a simple test problem as shown in Fig. 5. The relevant problem parameters are as follows: a nearly transparent unit circle, linear-triangular finite-elements mesh, pure isotropic scattering, total cross section, $\sigma_t = 0.01$ cm^{-1}, triangular mesh size, $\Delta \cong 0.02$ cm, $\sigma\Delta \cong 2x10^{-4}$, number of elements, $N_{elements}= 4,335$, and number of nodes, $N_{nodes}= 2,252$. The convergence of the scaled residual norm is set to 10^{-8} for both the outer and inner iterations. Timing results, from a single 1.6GHz Athlon processor, are presented in Table I.

The uncollided-flux preconditioner dramatically reduced the iteration count, down from several thousand to three for all S_n approximations. The speedup ratio increased almost linearly with the number of directions, as expected theoretically. It is noted that the number of the inner iterations per outer iteration is still quite large (1,000-1,800) so that other solution techniques, such as a single sweep of the conventional source iteration or a MultiLevel (ML) method, may be employed to further reduce runtime.

Table I. Serial Uncollided-Flux Preconditioner Results

S_n order	$N_{directions}$	Outer Iterations		Runtime (s)		speedup
		Ax=b	**$M^{-1}Ax= M^{-1}b$**	**Ax=b**	**$M^{-1}Ax= M^{-1}b$**	
S_4	6	4,711	3	21	6	3.5
S_6	12	6,758	3	123	12	10
S_8	20	7,185	3	323	19	17
S_{12}	42	7,551	3	1,312	41	32
S_{16}	72	7,980	3	3,904	72	54

4.2. Parallel Implementation

The parallel implementation of the uncollided-flux preconditioner is quite complicated and requires careful balance between computation and communication to make it effective. The CEPTRE solution strategy is to partition the spatial mesh across processors, illustrated in Fig. 6, and solve the space-direction dependence of the angular flux, without partitioning the directional dependence across processors. This has proven to be an effective solution strategy, because the directions are all coupled together by the scattering term.

We have considered two parallel strategies to implement the uncollided-flux preconditioner. The first approach is to partition the linear system both in direction and in space, as is shown in Fig. 7. This space-direction partition will improve parallel efficiency for the inner iterations, because the directional

dependence may be solved independently and the increased spatial-domain size requires less communication than the partition shown in Fig. 6. However, significant interprocessor communication may be required to transfer data between the mesh partitions for the outer iteration and the one for the inner iteration.

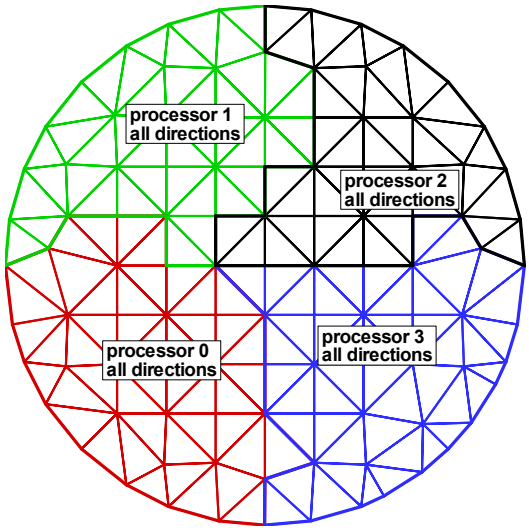

Figure 6. Typical Mesh Partition for Unpreconditioned Solve

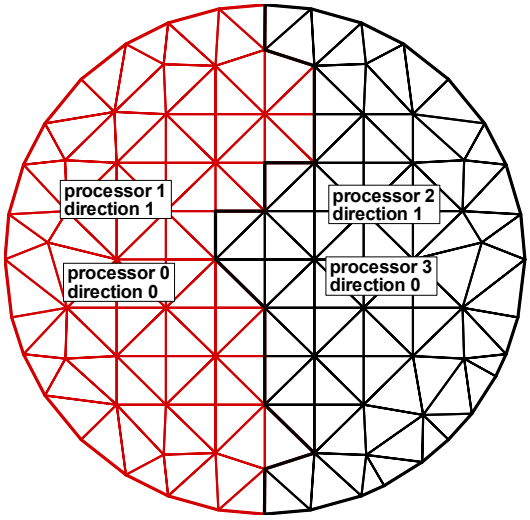

Figure 7. Mesh Partition for Preconditioned Solve with Distributed Directions

An alternative approach is to use the same parallel strategy for the inner and outer (preconditioner) iterations, using a partition strategy illustrated in Fig. 6 for both. As shown in Figs. 2 and 4, the sparsity structure is the same for both the outer and inner iterations except that the full blocks are replaced by the diagonal blocks. Specialized routines have to be implemented in the parallel iterative package (Aztec) to

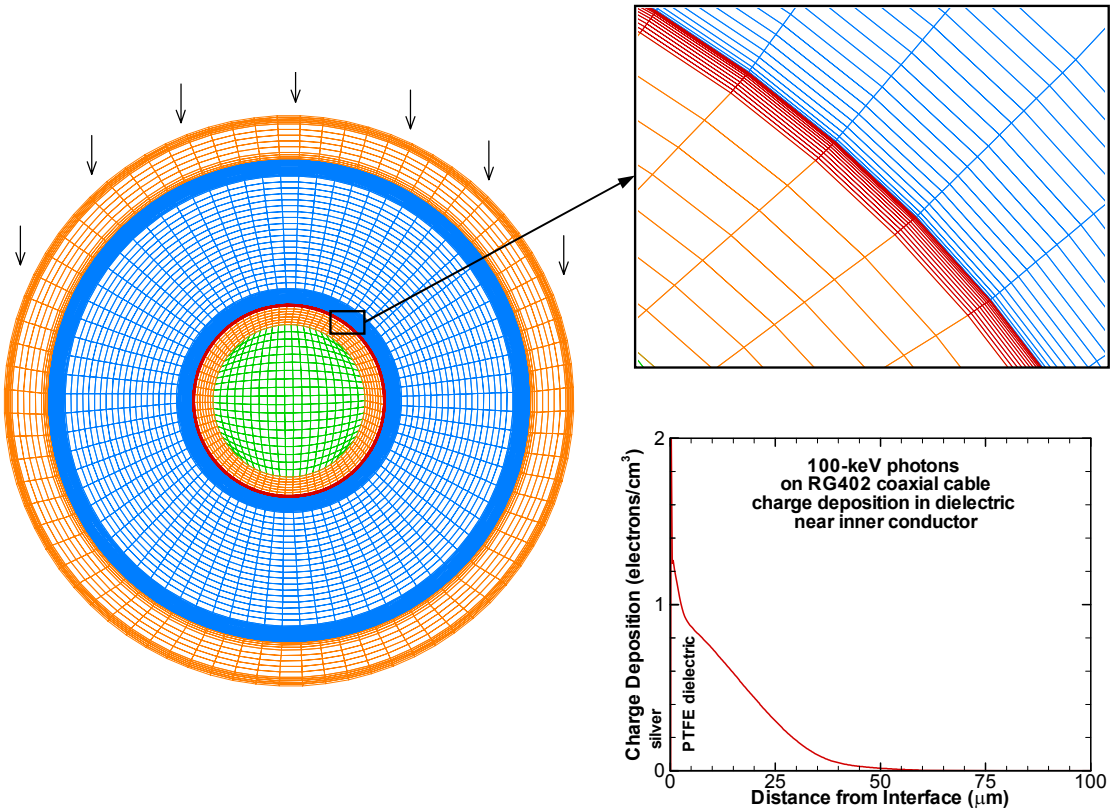

Figure 8. Linear-Quadrilateral Mesh of a Coaxial Cable and Solution Near Interface

handle the diagonal block matrix data format. This approach is less efficient for the inner iterations than that described previously but is more straightforward to implement and requires less data communication and manipulation between the outer and inner iterations. We have found that the performance of these two approaches depends upon specific computer architecture, that is, processor speed vs. communication bandwidth. Comparison of these two different parallel strategies warrants further studies.

In order to evaluate the parallel performance of the uncollided-flux preconditioner, we investigate the transport of 100-keV photon source normally incident on a coaxial cable as shown in Fig. 8. The 7,240-node linear-quadrilateral mesh is refined near the material interfaces to resolve the details of electron transport near the conductor-dielectric interfaces. Minimum mesh size, $\sigma\Delta$, is about 10^{-4} for the first-group photons. A P_3S_8 approximation is applied to represent the scattering cross section and the angular flux. The convergence rates for the first-group even-parity flux are plotted in Figure 9, showing significant reduction in iteration count with the uncollided-flux preconditioner. The measured runtime for the first-group photons, using 80 processors on the ASCI-Red computer, are given in Table II. The speedup ratio increases with the number of directions but is not as dramatic as that for the serial case, which can be attributed to the inefficiency of the diagonal-block implementation.

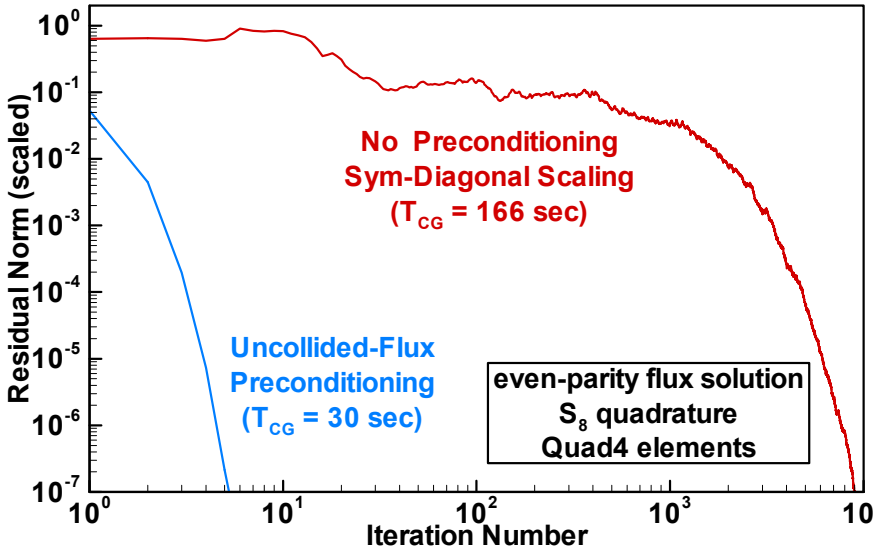

Figure 9. Convergence Rates of Unpreconditioned and Preconditioned Even-Parity Solution

Table II. Parallel Implementation of Uncollided-Flux Preconditioner

S_n order	Number of Directions	80 ASCI-Red Processor Runtime (s)		speedup
		Without Preconditioning (except diagonal scaling)	Uncollided-Flux Preconditioning	
S_2	2	18	45	0.4
S_4	6	36	47	0.8
S_6	12	140	52	2.7
S_8	20	324	75	4.3
S_{12}	42	1,001	176	5.7
S_{16}	72	3,462	430	8.0

5. CONCLUSIONS

We investigated the use of the uncollided-flux formulation as a preconditioner of the second-order transport equations. The preconditioner has similar structure to the full transport operator, but is much easier to solve because there is no directional coupling, without the scattering term. Computational results indicate that

the uncollided-flux preconditioner is quite effective in reducing both the iteration count and runtime in the CG algorithm.

Parallel implementation of the uncollided-flux preconditioner is interesting, as there are many options to partition the space-direction dependence and processor configuration. We have two parallel implementations in place in the CEPTRE code but they require more studies to optimize computation/communication performance. Furthermore, we will investigate the effectiveness of MultiLevel (ML) methods to accelerate convergence of the uncollided-flux preconditioner solutions. Incompletely converging the inner (preconditioner) iterations in some cases slightly reduced runtimes, however more study is needed to optimize balance between the inner and outer iterations.

6 ACKNOWLEDGEMENTS

The authors would like to express appreciation to Len Lorence and Gary Scrivner for support of this work.

REFERENCES

[1] C. R. Drumm and J. Lorenz, “Parallel FE Approximation of the Even/Odd-Parity Form of the Linear Boltzmann Equation,” *Mathematical and Computer Modelling*, **31**, pp. 55-71 (2000).

[2] W. C. Fan, C. R. Drumm and J. L. Powell, “Discrete Ordinates Approximations to the First- and Second-Order Radiation Transport Equations,” Sandia National Laboratories Report, SAND2002-1880 (2002).

[3] “ASCI Technology Prospectus: Simulation and Computational Science,” US Department of Energy Report, DOE/DP/ASC-ATP-001 (2001).

[4] J. E. Morel and J. M McGhee, “A Self-Adjoint Angular Flux Equation,” *Nuclear Science and Engineering*, 1332, pp. 312-325 (1999).

[5] E. A. Boucheron, et al., "ALEGRA: User Input and Physics Descriptions Version 4.0," Sandia National Laboratories Report, SAND2001-1992 (2001).

[6] J. J. Duderstadt and W. R. Martin, *Transport Theory*, Wiley-Interscience, New York (1979).

[7] C. R. Drumm, “Parallel Finite Element Electron-Photon Transport Analysis on 2-D Unstructured Mesh,” Sandia National Laboratories Report, SAND99-0098 (1999).

[8] Ray S. Tuminaro, Mike Heroux, Scott A. Hutchinson and John N Shadid, “Official Aztec User’s Guide Version 2.1,” Sandia National Laboratories Report, SAND99-8801J (1999).

[9] J. W. Demmel, Applied Numerical Linear Algebra, SIAM, Philadelphia (1997).

[10] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, Boston, MA (1996).