PHYSOR-2006, ANS Topical Meeting on Reactor Physics
Organized and hosted by the Canadian Nuclear Society. Vancouver, BC, Canada. 2006 September 10-14

Parallel Discrete Ordinates Methods in the CEPTRE Project

Shawn Pautz[*], Bill Bohnhoff, Clif Drumm, Wesley Fan,
Jennifer Liscum-Powell, and Len Lorence
Sandia National Laboratories, P.O. Box 5800, Albuquerque,
New Mexico, 87185, United States

Abstract

The CEPTRE code system consists of a parallel implementation of the second-order form of the Boltzmann transport equation. Work is underway to implement a parallel first-order solver as well. We examine the properties of both solution approaches, discussing their theoretical asymptotic runtime performance as a function of various scaling parameters. We also provide results for both the first- and second-order implementations. Understanding of the behavioral differences of these two forms can help guide the selection of an appropriate solver for a given problem.

KEYWORDS: parallel, discrete ordinates, deterministic transport, scaling

1. Introduction

CEPTRE (Coupled Electron-Photon Transport for Radiation Effects) is a deterministic code for solving the linear steady-state Boltzmann transport equation. The current implementation solves the second-order form of the transport equation by means of the multigroup energy discretization, the discrete ordinates angular discretization, and continuous finite element spatial discretization on unstructured meshes. A unique feature in the solution method of CEPTRE is that it solves the spatial and angular dependence simultaneously without the conventional inner iteration. This implementation leads to fast convergence rates for electron transport and good parallel efficiency. CEPTRE's primary application is to predict the effects of x-rays and secondary electrons on cables and other electronic components. The high resolution needed for the modeling of electron boundary layers (sub-micron) often requires the use of large meshes and massively parallel computations.

We are currently implementing a parallel solver for the first-order form of the transport equation. The iterative solution process for the first-order form is fundamentally different than that for the second-order form, resulting in a different set of strengths and weaknesses for each approach. It is these differences that we wish to exploit when choosing between the two approaches for a particular application. This can be particularly useful for problems with coupled particle species or a single species with different behaviors at different energies; one can create a hybrid solver that employs both transport forms for different particles/energies in order to minimize the overall runtime.

[*] Corresponding author, Tel. 505-284-4291, Fax. 505-844-0092, E-mail: sdpautz@sandia.gov

2. Forms of the Transport Equation

The first-order form of the monoenergetic transport equation is

$$[\Omega \cdot \nabla + \sigma_t]\psi(r,\Omega) = M\Sigma D\,\psi(r,\Omega) + Q(r,\Omega), \tag{1}$$

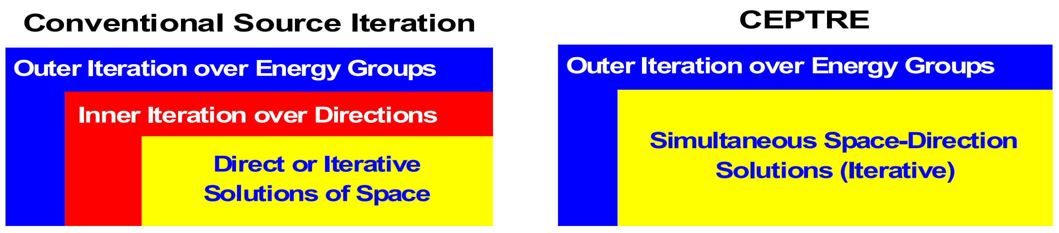
where $\psi(r,\Omega)$ is the angular flux, M and D are the moments-to-discrete and discrete-to-moments operators, respectively, and Σ is the scattering operator. [1] The solution method typically used is source iteration, depicted in Fig. 1a. Sources due to scattering from other energy groups are computed in the outer iteration. In the inner iteration the within-group scattering is fixed, thus decoupling (temporarily) the fluxes from different angles. This allows nested loops over the angular variable and the spatial variable, usually solved by a "sweeping" algorithm, after which a new within-group scattering source is computed and a new inner iteration begins.

The second-order form of the monoenergetic transport equation is

$$\left[-\Omega \cdot \nabla \Re^{-1} \Omega \cdot \nabla + \Re\right]\psi(r,\Omega) = Q(r,\Omega) - \Omega \cdot \nabla\left[\Re^{-1} Q(r,\Omega)\right], \tag{2}$$

where $\Re$ is a removal term that incorporates the self-scattering cross sections. [2] The above form differs from the first-order form in that it has a second-order operator in space and all within-group scattering appears on the left side of the equation. This naturally leads to the solution methodology shown in Fig. 1b. The coupling between directions contained in the removal operator yields a simultaneous iteration over the combined angular and spatial variables. Since the operator is self-adjoint and SPD it may be solved by means of a parallel conjugate gradient solver.

Figure 1: Conventional source iteration and the iteration approach for the second-order form in CEPTRE.


Conventional Source Iteration
Outer Iteration over Energy Groups
Inner Iteration over Directions
Direct or Iterative Solutions of Space
CEPTRE
Outer Iteration over Energy Groups
Simultaneous Space-Direction Solutions (Iterative)

3. Comparison of First- and Second-Order Forms

We compare the theoretical behavior of the solvers for the first- and second-order forms, concentrating on the computational runtime performance. As a baseline case we assume that a standard "volumetric" spatial decomposition is used, that the iterative systems are not preconditioned, and that the number of directions grows at most weakly with increasing numbers of processors. In Table 1 we indicate the expected behavior of the first- and second-order solvers, both in terms of the expected runtime per iteration (conjugate gradient (CG) iteration for the second-order form and source iteration (SI) for the first-order form) and the expected number of iterations; the runtime is given by the product of these quantities. In the table N_D is the number of directions, N_E is the number of spatial elements, P is the number of processors, $\langle \sigma_a h \rangle$ is a representative absorption optical thickness of each element, c_n is the maximum of σ_{sn}/σ_t for all scattering orders, and d is the dimensionality of the problem (2 or 3). We discuss the tabulated behaviors in more detail below.

Table 1: Asymptotic behavior of first- and second-order solvers under different conditions.

Computing Conditions	**1st order**		**2nd order**	
	Time/Source Iteration	**# Iterations**	**Time/CG Iteration**	**# Iterations**
Baseline	$N_D N_E P^{-(d-1)/d}$	$[\ln(1/c_n)]^{-1}$	$N_D^2 N_E P^{-1}$	$\langle \sigma_a h \rangle^{-1}$
Large N_D	$N_D N_E P^{-1}$	$[\ln(1/c_n)]^{-1}$	$N_D^2 N_E P^{-1}$	$\langle \sigma_a h \rangle^{-1}$
KBA Decomposition	$N_D N_E P^{-1}$	$[\ln(1/c_n)]^{-1}$	$N_D^2 N_E P^{-1}$	$\langle \sigma_a h \rangle^{-1}$
Diagonal Preconditioner	N/A	N/A	$(c_1 N_D + c_2 N_D^2) N_E P^{-1}$	$\langle \sigma_a h \rangle^{-1}$
Extended Transport Correction Preconditioner	$N_D^2 N_E P^{-(d-1)/d}$	$[\ln(1/c_{n,eff})]^{-1}$	N/A	N/A

3.1 Parallel Scaling

The parallel performance of a computation depends not only on the parallel algorithm but also on the computer architecture. For the purposes of this discussion we will concentrate on behavior inherent in the parallel algorithm and neglect architecture-specific effects by assuming that communication costs are small relative to computation costs. Assuming that the spatial decomposition produces good load balancing, the parallel CG algorithm employed by the second-order solver should scale well with processor count, with runtime inversely proportional to P. The behavior of the first-order solver is more complicated. The pipeline fill and empty stages of the sweep process can leave many processors idle, yielding less than perfect scaling with processor count in many situations. This effect may be overcome by using a sufficiently large angular quadrature relative to P or by using specialized spatial decompositions such as the KBA approach. [3]

3.2 Scaling with Problem Size

Both the first- and second-order forms use the multigroup approximation and should scale similarly with the number of energy groups. Despite different spatial differencing, the CPU time per iteration for both forms increases linearly with N_E (where N_E may refer to either the number of finite elements or the number of nodes, depending on the spatial differencing). The main difference in the CPU time per iteration between the two forms is a result of the angular differencing. Although both the first- and second-order forms use discrete ordinates angular differencing, the CG solution method used in the second-order form involves a full angle-to-angle scattering matrix, which increases the runtime by N_D^2. The source iteration approach of the first-order form, however, employs angular flux moments as an intermediate part of the scattering source calculation, which for a fixed scattering kernel increases the runtime by N_D.

3.3 Scaling with Scattering Kernel

There are two parameters in the scattering kernel that can affect the convergence rate, and hence the number of iterations, of the first- and second-order solvers: the scattering ratios c_n and the absorption optical thickness $\langle \sigma_a h \rangle$. As $c_n \to 1$ the iteration count for the first-order solver grows without bound, whereas it has no effect on the second-order solver. Conversely, as $\langle \sigma_a h \rangle \to 0$ the iteration count of the second-order form grows without bound, whereas it has no effect on the first-order solver.

3.4 Preconditioners

There are numerous preconditioners that have been developed for various first- and second-order discretizations. Here we discuss just a few of them that have been considered and/or implemented in CEPTRE. The second-order solver may employ a diagonal preconditioner, which takes one of two forms. The "diagonal scaling" preconditioner is a standard technique used to improve the condition number of a CG iteration. The "block diagonal" or "uncollided flux" preconditioner is a transport-specific technique that diagonalizes the scattering matrix. [4] Although both diagonal preconditioners can substantially reduce the iteration count compared to an unpreconditioned solver, neither one changes the ultimate asymptotic behavior.

The extended transport correction preconditioner adds or subtracts an arbitrary amount of delta (forward) scattering from the scattering kernel, which has no effect on the converged solution. Since this modifies c_n but not $\langle \sigma_a h \rangle$, it will affect the iteration count of the first-order solver but have no effect on the second-order convergence rate. We note, however, that in order to use this technique one must use a full angle-to-angle scattering operator, so the time per iteration for the first-order form will necessarily scale as N_D^2.

4. Results

In Fig. 2 we present the measured runtime performance of the second-order form on the Sandia Red platform in which all problem parameters are fixed except P. We find generally good agreement with the predictions of Table 1. In the table we predict that the time per CG iteration should be inversely proportional to processor count, i.e. that the parallel efficiency should be relatively constant; Fig. 2 reveals only weak dependence of the efficiency on processor count.

In Figs. 3 and 4 we show the effect of the number of angular directions N_D on the number of iterations and time per iteration for the first- and second-order computations, respectively. (These and all subsequent computational results were generated on a small Linux cluster.) The iteration count of the first-order form is completely independent of N_D, as expected. Except for some weak dependence for a small number of angles the same is true for the second-order form. The time per iteration of the first-order form shows the expected linear dependence on N_D, whereas the second-order form gives the predicted quadratic dependence.

In Figs. 5 and 6 we depict the effect of the scattering ratio c on the performance of the first- and second-order forms. The number of iterations of the first-order form increases as expected as c approaches unity, whereas the time per iteration remains essentially constant, also as expected. For the second-order form the scattering ratio was varied by changing the absorption cross section, so we expect that the iteration count will also increase as c approaches unity, as the plot shows.

In Figs. 7 and 8 we present the effect of decreasing cell size (increasing number of cells or nodes) on the runtime performance of the first- and second-order forms. As expected, the number of iterations of the first-order form remains constant whereas the time per iteration increases linearly with the number of cells. For the second-order form, the number of CG iterations increases without bound as the cell size decreases whereas the time per iteration is proportional to the number of nodes. In other words, the number of CG iterations will increase without bound as the cell become optically thin or transparent. This also explains the effectiveness of the second-order form in electron transport due to the short electron range.

In Fig. 9 we present the overall timing results of applying the block-diagonal preconditioner to the second-order form. For a modest number of directions this preconditioning technique reduces the overall iteration time such that the solution time is proportional to N_D instead of N_D^2. However, we have observed an effective exponent of ~1.2 for most practical problems.

Figure 2: Efficiency of second-order computation as a function of number of processors.

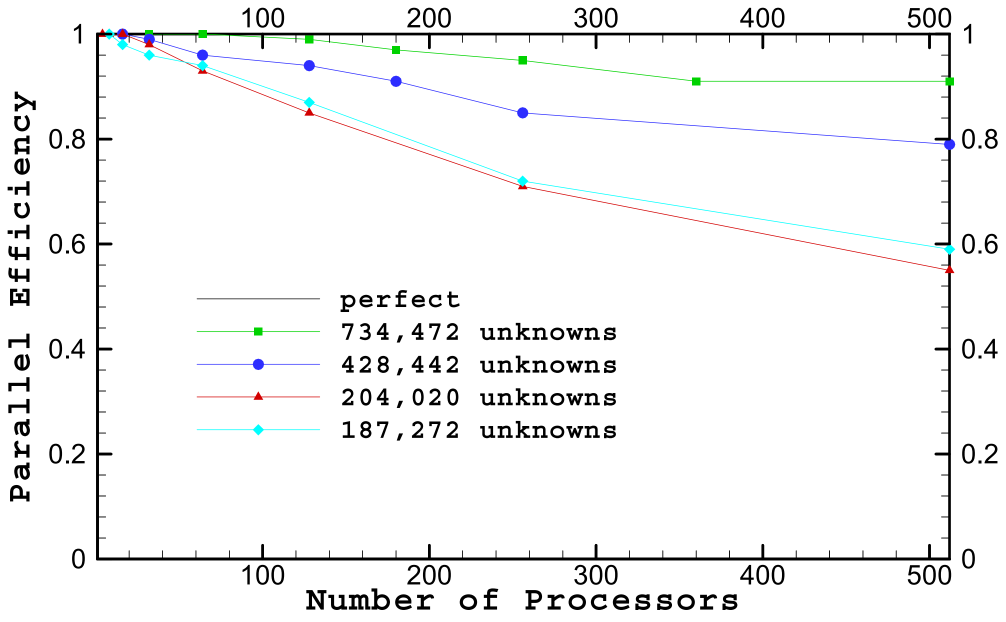


Figure 3: Number of iterations and time/iteration of first-order computation as functions of number of angular directions.

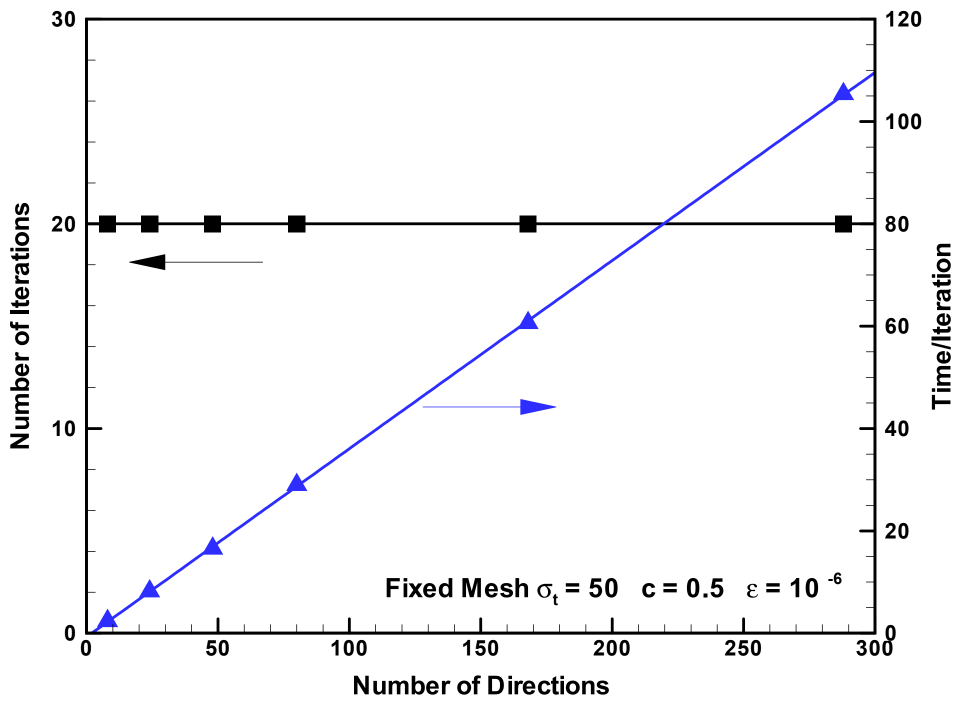


Figure 4: Number of iterations and time/iteration of second-order computation as functions of number of angular directions.

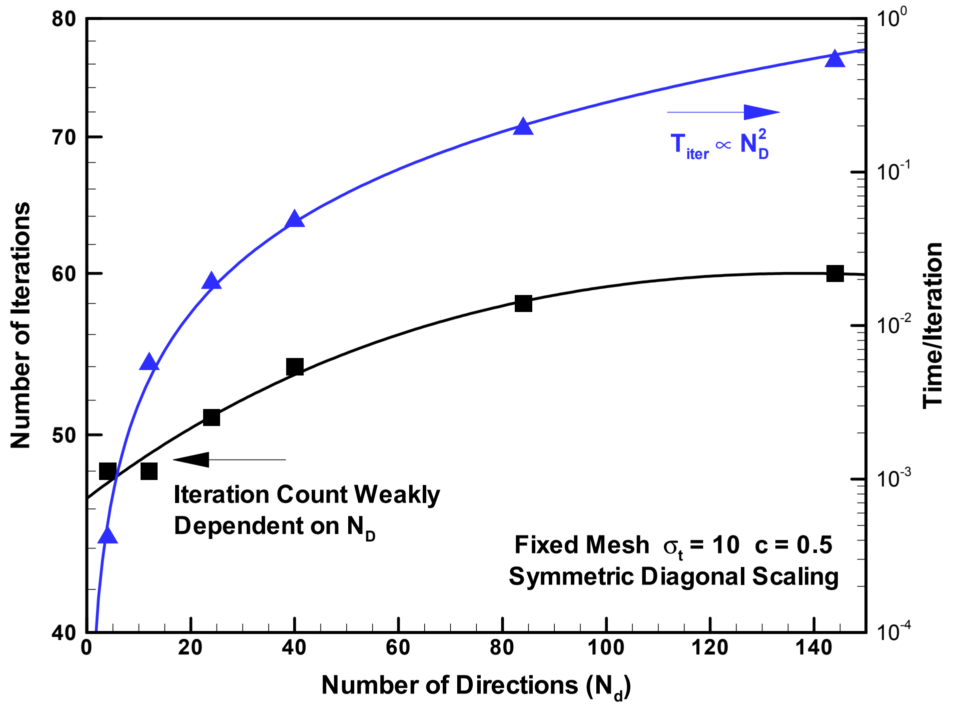


Figure 5: Number of iterations and time/iteration of first-order computation as functions of scattering ratio.

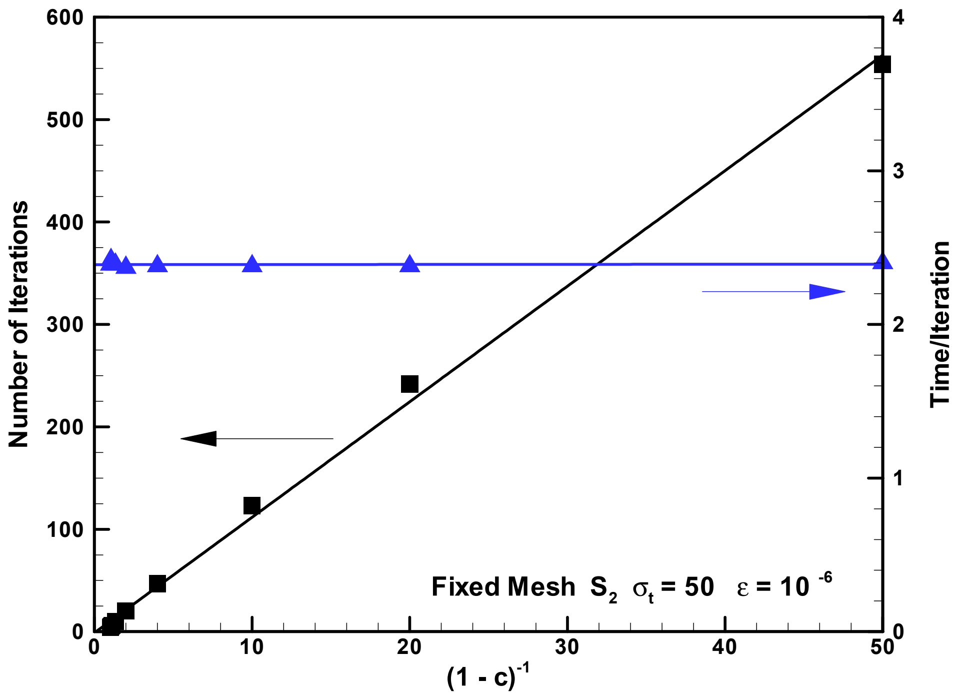


Figure 6: Number of iterations and time/iteration of second-order computation as functions of scattering ratio (variable absorption).

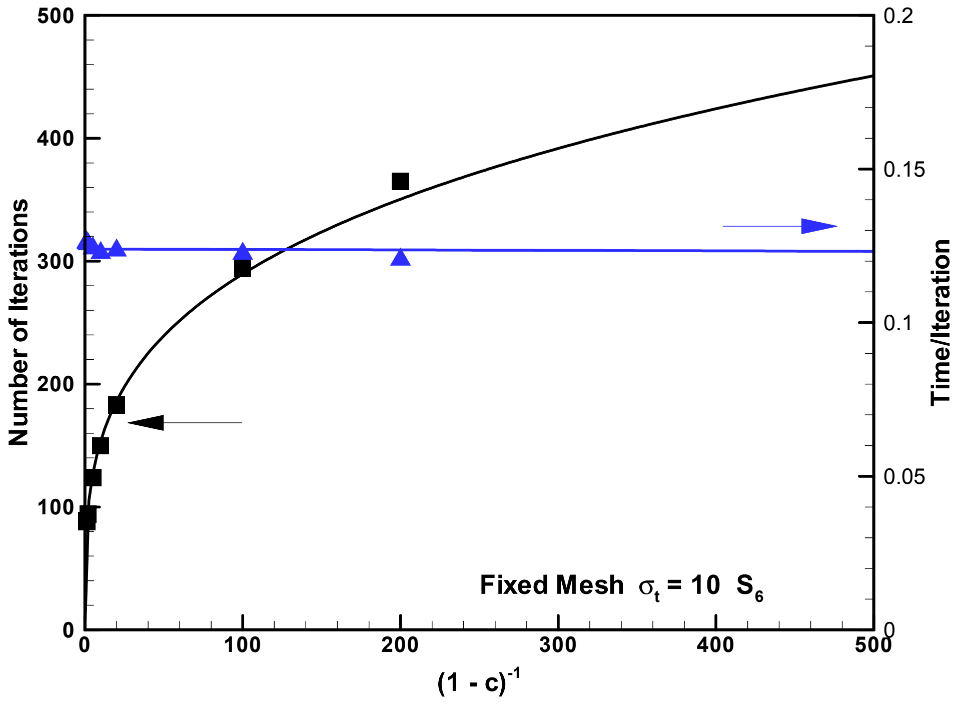


Figure 7: Number of iterations and time/iteration of first-order computation as functions of cell size.

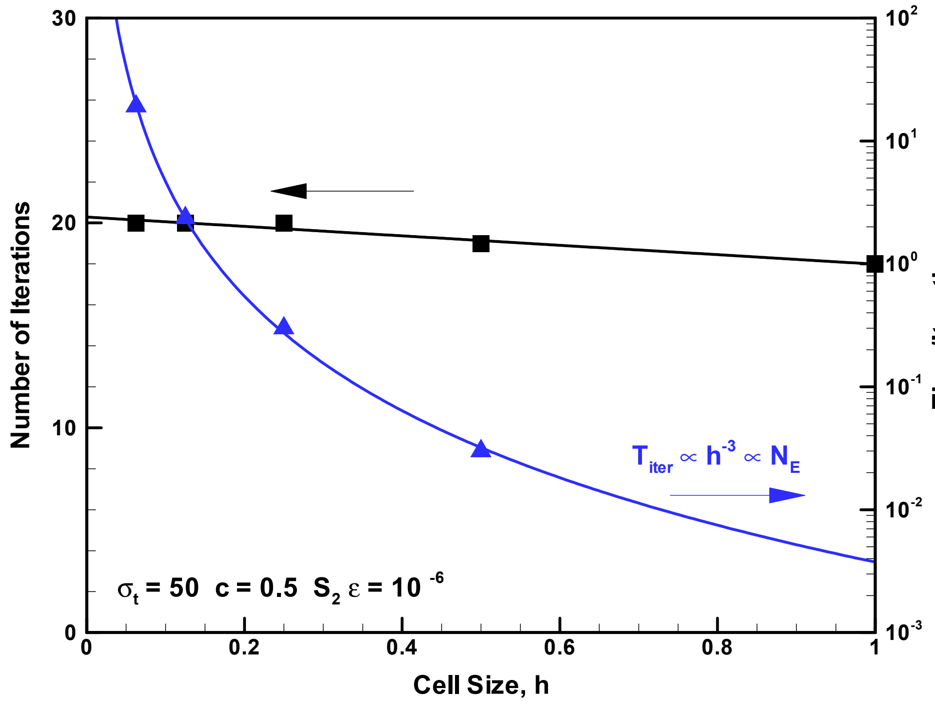


Figure 8: Number of iterations and time/iteration of second-order computation as functions of cell size.

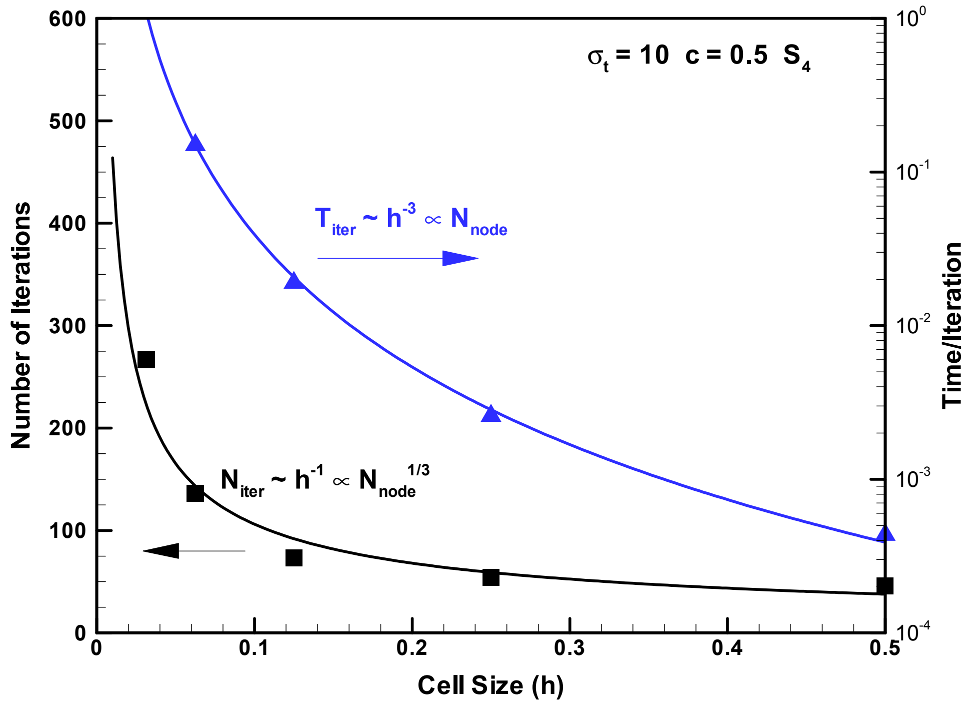
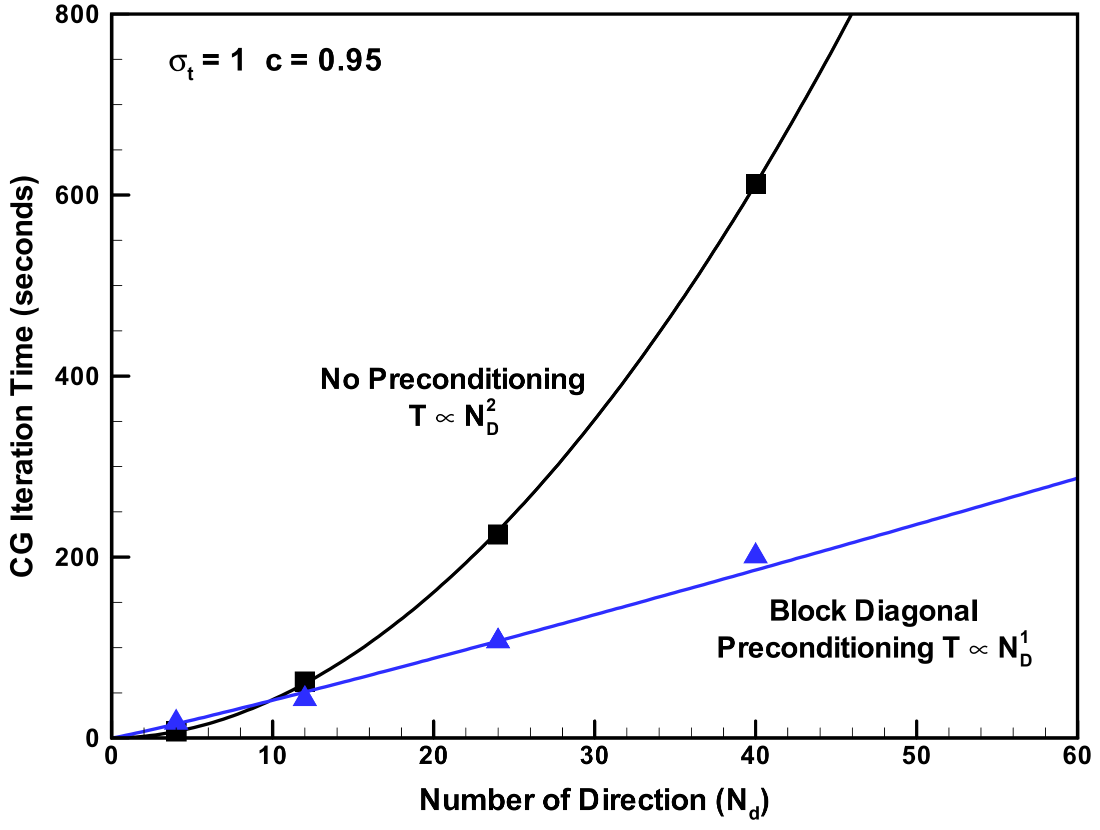


Figure 9: Effectiveness of block-diagonal preconditioning for the second-order computation.



5. Conclusions

Table 1 may be profitably used to guide the selection of an appropriate solver. For our particular application the dependence on the cross section is important. The elements of a mesh refined enough to resolve electron boundary layers are very optically thin for x-rays, thus driving up the corresponding number of CG iterations substantially. Since the first-order form is insensitive to the optical thickness it is attractive as an x-ray solver, particularly as spatial refinement studies are conducted. This is the immediate driver for our implementing a first-order solver in CEPTRE. But since these predictions are not specific to our particular application, they may be used in other fields such as reactor physics that may find a need for first- or second-order deterministic transport.

Acknowledgements

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DEAC04-94AL85000

References

1) E. E. Lewis and W. F. Miller, Jr., *Computational Methods of Neutron Transport*, John Wiley and Sons, New York (1984).
2) C. R. Drumm and J. Lorenz, "Parallel Finite-Element Electron-photon Transport Analysis on a 2D Unstructured Mesh," Conference Proceedings of Mathematics and Computation, Reactor Physics and Environmental Analysis in Nuclear Applications, Madrid, Spain, pp. 858-868, **1** (1999).
3) S. D. Pautz, "An Algorithm for Parallel S_n Sweeps on Unstructured Meshes," *Nucl. Sci. Eng.*, **140**, 111-136 (2002).
4) C. R. Drumm and W. C. Fan, "Uncollided-Flux Preconditioning of the Conjugate Gradients Solution of the Transport Equation," Proc. Int. Conf. Nuclear Mathematical and Computational Sciences, Gatlinburg, TN, April 6-10, 2003.