A Monte Carlo Synthetic Acceleration Method for Time-Dependent Radiation Transport

Thomas Evans

Nuclear Science & Technology Division

Managed by UT-Battelle
for the Department of Energy

Oak Ridge National Laboratory

Outline

- **Monte Carlo Linear Solvers**
- **Monte Carlo Synthetic Acceleration**
- **Linearized Radiation Diffusion Model**
- **Results**
  - **Marshak wave**
  - **multi-material problem**
- **Conclusions and future work**

Monte Carlo Linear Solvers

- **Monte Carlo methods for solving linear systems have been around a long time (Hammersley and Handscomb, 1964).**
  - **Slow and noisy**
- **Halton's (1962-1994) Sequential Monte Carlo is a residual technique that is far more efficient (exponential convergence).**
- **This work has not made significant impacts in the computational transport (or broader numerical) community.**

A Little Review

- **The linear matrix system,**

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

- **Define the *iteration* matrix,**

$$\mathbf{x} = (\mathbf{I} - \mathbf{A})\mathbf{x} + \mathbf{b} = \mathbf{H}\mathbf{x} + \mathbf{b}$$

- **Leads to the *Neumann* Series.**

$$\mathbf{x} = (\mathbf{I} - \mathbf{H})^{-1}\mathbf{b} = (\sum_{k=0} \mathbf{H}^k)\mathbf{b} = \mathbf{b} + \mathbf{H}\mathbf{b} + \mathbf{H}^2\mathbf{b} + \mathbf{H}^3\mathbf{b} + \ldots$$

A Little More Review

- **The Neumann Series will converge given**

$$\rho(\mathbf{H}) < 1$$

- **This implies the following iterative solution**

$$\mathbf{x}^{k+1} = \mathbf{H}\mathbf{x}^{k} + \mathbf{b}$$

which is also known as *Richardson*, or fixed-point, iteration.

Monte Carlo Solutions of Linear Systems

- **Random walks can simulate the terms in the Neumann series.**
- **There are two approaches:**
  - **Direct methods**
  - **Adjoint methods**
- **Direct methods estimate each component of $\mathbf{x}$ from a random walk.**
- **Adjoint methods estimate all components of $\mathbf{x}$ from a random walk.**

Monte Carlo Direct Method

- **The direct method estimates each component of the vector from a random walk**

$$x_i = (\mathbf{b})_i + (\mathbf{Hb})_i + (\mathbf{H}^2\mathbf{b})_i + \dots$$
$$= \sum_{k=0}^{\infty} \sum_{i_1}^{N} \sum_{i_2}^{N} \dots \sum_{i_k}^{N} h_{i,i_1} h_{i_i,i_2} \dots h_{i_{k-1},i_k} b_{i_k}$$

- **The sampled value of X from a single random walk with k events is**

$$X(i_0 = i) = \sum_{m=0}^{k} W_m b_{i_m} = \sum_{m=0}^{k} w_{i,i_1} w_{i_1,i_2} \dots w_{i_{m-1},i_m} b_{i_m}$$

Monte Carlo Direct Method

- **The expected value of $\mathbf{X}$ is**

$$E[X(i_0 = i)] = x_i = \sum_{\nu} P_\nu X_\nu$$

$$= \sum_{k=0}^{\infty} \sum_{i_1}^{N} \sum_{i_2}^{N} \cdots \sum_{i_k}^{N} p_{i,i_1} p_{i_1,i_2} \cdots p_{i_{k-1},i_k} w_{i,i_1} w_{i_1,i_2} \cdots w_{i_{k-1},i_k} b_{i_k}$$

- **Equating with the Neumann series gives the weights,**

$$w_{ij} = \frac{h_{ij}}{p_{ij}}$$

- **Where p_{ij} is the probability for transitioning from state $i \rightarrow j$.**

Implementation of the Direct Method

- **The probability matrix, P, defines the PDF for selecting a new state *j* for a current state *i***

$$p_{ij} = \frac{|h_{ij}|}{\sum_j |h_{ij}|}$$

- **Termination is done by augmenting the system with a terminating state or by weight cutoff:**
  - **generally, weight cutoff is chosen**
  - **augmentation is similar to analog absorption (probability of transitioning from state *i* to a null state)**

An Example

The linear system

$$\begin{pmatrix} 1.0 & -0.3 & -0.4 \\ -0.2 & 1.0 & -0.4 \\ -0.5 & -0.4 & 1.0 \end{pmatrix} \mathbf{x} = \begin{pmatrix} 10 \\ 1 \\ 1 \end{pmatrix}$$

The iteration matrix

$$\mathbf{H} = \begin{pmatrix} 0 & 0.3 & 0.4 \\ 0.2 & 0 & 0.4 \\ 0.5 & 0.4 & 0 \end{pmatrix}$$

build a probability matrix

$$\downarrow$$

$$\mathbf{P} = \begin{pmatrix} 0.0 & 0.4 & 0.6 \\ 0.3 & 0.0 & 0.7 \\ 0.6 & 0.4 & 0.0 \end{pmatrix}$$

The spectral radius of this system is 0.738, so it can be solved by the direct method.

1. build an iteration matrix
2. build a probability matrix
3. perform random walks

Random Walk Algorithm

```
for n = 1 to Np do
   {set state s}
   while walk do
      x(i) = x(i) + w * b(s)
      {sample p(i,j) to get ns}
      w = w * H(s,ns)/P(s,ns)
      if w < wc then
         walk = 0
      end if
      s = ns
   end while
end for
```

- starting state
- tally to component of $\mathbf{x}$
- sample a new state from $\mathbf{P}$ (need to build a CDF from $\mathbf{P}$)
- update weight
- check weight cutoff
- update state

Monte Carlo Adjoint Method

- **Sample all components of $\mathbf{x}$ in a single random walk.**
- **Sample the Neumann series in reverse, the weight change from state $i \rightarrow j$ is**

$$w_{ij} = \frac{h_{ji}}{p_{ij}}$$

- **And, the probability matrix is calculated from**

$$p_{ij} = \frac{|h_{ji}|}{\sum_j |h_{ji}|}$$

Adjoint Method, cont

- **The estimator for this method is,**

$$X = \sum_{m=0}^{k} W_m \delta_{i_m,i}$$
$$= \sum_{m=0} \hat{b}_{i_0} w_{i_0,i_1} w_{i_i,i_2} \ldots w_{i_{m-1},i_m} \delta_{i_m,i}$$

- $\hat{b}_{i_0}$ **is the sampled source in state** i_0**. Only tally in state where random walk resides.**
- **In this sense, the adjoint method is equivalent to forward Monte Carlo transport methods.**

Sequential Monte Carlo

- **Both the direct and adjoint methods converge slowly and are plagued by statistical noise.**
- **The Sequential Monte Carlo method of Halton allows for faster convergence because it is a residual method and therefore is not constrained by the Central Limit Theorem.**

$$\mathbf{r}^l = \mathbf{b} - \mathbf{A}\mathbf{x}^l$$ estimate residual

$$\mathbf{A}\delta\mathbf{x}^l = \mathbf{r}^l$$ adjoint Monte Carlo with residual source

$$\mathbf{x}^{l+1} = \mathbf{x}^l + \delta\mathbf{x}^l$$ update $\mathbf{x}$

Synthetic Schemes

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

Split $\mathbf{A} = \mathbf{I} - (\mathbf{I} - \mathbf{A})$.

Subtract

$$\mathbf{x} = (\mathbf{I} - \mathbf{A})\mathbf{x} + b$$

$$\mathbf{x}^{l+1} = (\mathbf{I} - \mathbf{A})\mathbf{x}^{l} + b$$

Gives

$$\delta\mathbf{x}^{l+1} = (\mathbf{I} - \mathbf{A})\delta\mathbf{x}^{l}$$

Subtract $(\mathbf{I} - \mathbf{A})\delta\mathbf{x}^{l+1}$

$$\mathbf{A}\delta\mathbf{x}^{l+1} = (\mathbf{I} - \mathbf{A})(\mathbf{x}^{l+1} - \mathbf{x}^{l})$$

$$\mathbf{x}^{l+1} = (\mathbf{I} - \mathbf{A})\mathbf{x}^{l} + \mathbf{b}$$

$$\mathbf{A}\delta\mathbf{x}^{l+1} = (\mathbf{I} - \mathbf{A})(\mathbf{x}^{l+1} - \mathbf{x}^{l})$$

$$\mathbf{x} = \mathbf{x}^{l+1} + \delta\mathbf{x}^{l+1}$$

Finished in 1 iteration

We can create a new iteration scheme by approximating $\mathbf{A}$.

An Approximate Monte Carlo Synthetic Scheme

- **We define an approximate Synthetic Scheme using Sequential Monte Carlo:**

$$\mathbf{x}^{l+1/2} = (\mathbf{I} - \mathbf{A})\mathbf{x}^{l} + \mathbf{b}$$

$$\mathbf{r}^{l+1/2} = \mathbf{b} - \mathbf{A}\mathbf{x}^{l+1/2}$$

$$\hat{\mathbf{A}}\delta\mathbf{x}^{l+1/2} = \mathbf{r}^{l+1/2} \rightarrow \text{approximates } (\mathbf{I} - \mathbf{A})(\mathbf{x}^{l+1/2} - \mathbf{x}^{l})$$

$$\mathbf{x}^{l+1} = \mathbf{x}^{l+1/2} + \delta\mathbf{x}^{l+1/2}$$

- **The adjoint Monte Carlo method is used to estimate $\delta\mathbf{x}^{l+1/2}$.**

Practicalities

- **1-D studies on time-dependent diffusion problems showed good results using Sequential Monte Carlo:**
  - **time-dependent problems provide a good estimate of the residual at each timestep**
  - **exponential convergence maintained**
- **MCSA is more robust; applicable to large 3-D problems.**
- **Eventual goal is to use MCSA in nonlinear-problems (ie. Newton-MCSA).**

Thermal Radiation Diffusion Model

- **The nonlinear equilibrium diffusion equation,**

$$(\rho C_v + 4aT^3)\frac{\partial T}{\partial t} - \nabla \cdot \left(\frac{4acT^3}{3\sigma_{\mathrm{R}}}\right)\nabla T = Q$$

- **Use the angle-energy integrated radiation intensity as the state variable,** $\phi = acT^4$

$$\left(\frac{C_v}{4acT^3} + \frac{1}{c}\right)\frac{\partial \phi}{\partial t} - \nabla \cdot D\nabla\phi = Q$$

Time-Discretization

- **Backward-Euler time differencing,**

$$-\nabla \cdot D^n \nabla \phi + \tilde{\sigma}^n \phi = q^n$$

- **with**

$$\tilde{\sigma}^n = \frac{\rho C_v^n}{4ac(T^n)^3 \Delta t} + \frac{1}{c\Delta t}$$

$$q^n = \tilde{\sigma}^n \phi^n + Q^n$$

$$\phi^n = ac(T^n)^4$$

- **Applying Fick's Law:**

$$\nabla \cdot \mathbf{F} + \tilde{\sigma}^n \phi = q^n$$

3-D Spatial Discretization

- **On a 3-D orthogonal mesh standard Finite-Volume differencing is applied,**

$$(F_{i+1/2} - F_{i-1/2})\Delta_j\Delta_k + (F_{j+1/2} - F_{j-1/2})\Delta_i\Delta_k + (F_{k+1/2} - F_{k-1/2})\Delta_i\Delta_j + \tilde{\sigma}^n_{ijk}\phi_{ijk}V_{ijk} = q^n_{ijk}V_{ijk}$$

- **Ensuring continuity of flux at cell faces gives face-centered diffusion coefficients,**

$$D_{l+1/2} = \frac{2\Delta_{l+1/2}}{3\sigma_l\Delta_l + 3\sigma_{l+1}\Delta_{l+1}}$$

$$D_{l-1/2} = \frac{2\Delta_{l-1/2}}{3\sigma_l\Delta_l + 3\sigma_{l-1}\Delta_{l-1}}$$

Flux Evaluation

- **Discretizing Fick's Law gives the face fluxes and**

$$\sigma_{T\,ijk}\phi_{ijk} - \sigma^-_{i+1}\frac{\Delta_{i+1}}{\Delta_i}\phi_{i+1\,jk} - \sigma^+_{i-1}\frac{\Delta_{i-1}}{\Delta_i}\phi_{i-1\,jk} - \sigma^-_{j+1}\frac{\Delta_{j+1}}{\Delta_j}\phi_{i\,j+1\,k}$$
$$- \sigma^+_{j-1}\frac{\Delta_{j-1}}{\Delta_j}\phi_{i\,j-1\,k} - \sigma^-_{k+1}\frac{\Delta_{k+1}}{\Delta_k}\phi_{ij\,k+1} - \sigma^+_{k-1}\frac{\Delta_{k-1}}{\Delta_k}\phi_{ij\,k-1} = q^n_{ijk}\,.$$

- **The $\{\sigma^+{}_{l-1}, \sigma^-_{l+1}\}$ can be thought of as leakage probabilities from adjacent cells into the current cell:**
  - **The Discrete-Diffusion Method (Densmore, Evans, Urbatsch) uses this interpretation.**
  - **The diffusion operator must be an H-matrix (symmetric in/out/leakage at faces).**
  - **Prevents pre-conditioning strategies.**

Preconditioning

- **For MCSA we do not require a SPD system; however, we do need the spectral radius less than 1,**

$$\mathbf{M}^{-1}\mathbf{D}\phi = \mathbf{M}^{-1}\mathbf{q} \qquad \mathbf{M} = \mathrm{diag}(\mathbf{D})$$

- **This choice achieves:**
  - It reduces the spectral radius such that $\rho(\mathbf{M}^{-1}\mathbf{D}) < \rho(\mathbf{D})$ and $\rho(\mathbf{M}^{-1}\mathbf{D}) < 1$. Thus, convergence is guaranteed and fewer iterations are required resulting in accelerated convergence.
  - Makes the diagonal elements of the iteration matrix zero, $\mathrm{diag}(\mathbf{I}-\mathbf{M}^{-1}\mathbf{D}) = 0$. Thus, there will be no time wasted in transitions $i \rightarrow i$ during the random walk process.

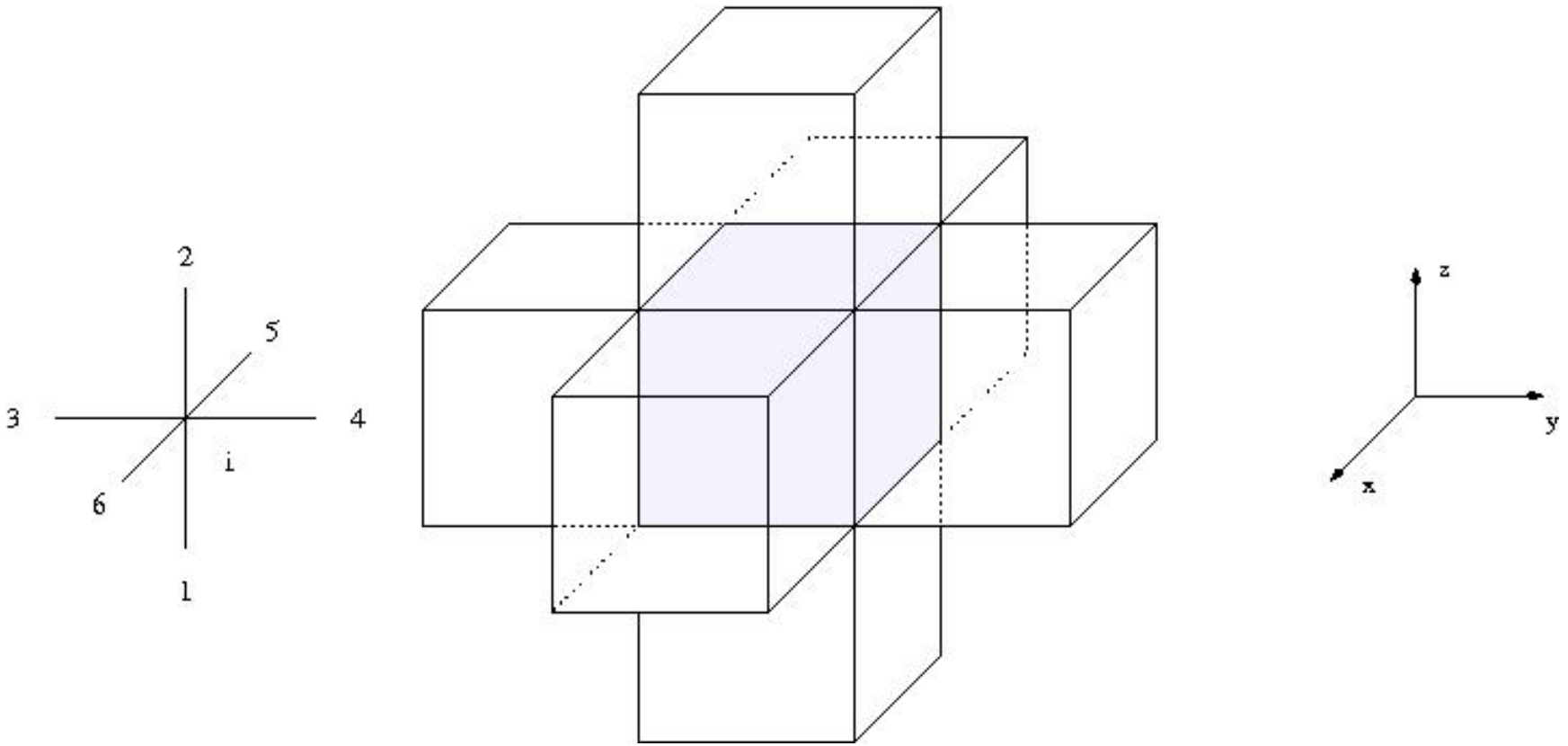
MCSA for Time-Dependent Diffusion

- **The MCSA algorithm has 2 basic steps:**
  - **a fixed-point iteration (matrix-vector multiply)**
  - **an adjoint Monte Carlo solve**

$$\begin{aligned}
\boldsymbol{\phi}^{l+1/2} &= (\mathbf{I} - \mathbf{M}^{-1}\mathbf{D})\boldsymbol{\phi}^{l} + \mathbf{M}^{-1}\mathbf{q}\,, && \text{(fixed-point iteration)}\\
\mathbf{r}^{l+1/2} &= \mathbf{M}^{-1}\mathbf{q} - \mathbf{M}^{-1}\mathbf{D}\boldsymbol{\phi}^{l+1/2}\,, && \text{(compute residual)}\\
\mathbf{M}^{-1}\mathbf{D}\delta\boldsymbol{\phi}^{l+1/2} &= \mathbf{r}^{l+1/2}\,, && \text{(estimate } \delta\phi \text{ using adjoint Monte Carlo method)}\\
\boldsymbol{\phi}^{l+1} &= \boldsymbol{\phi}^{l+1/2} + \delta\boldsymbol{\phi}^{l+1/2}\,. && \text{(update } \boldsymbol{\phi}\text{)}
\end{aligned}$$

- **Where the adjoint method is run with a small number of particles and only approximates the operator.**

Monte Carlo Interpretation



Segment of 3D orthogonal mesh:
Each cell has six adjacent cells.

Monte Carlo Interpretation

- **The adjoint Monte Carlo solve has a natural transport interpretation:**
  - **build probabilities for scattering from cell $i \rightarrow n$ using the adjacent cell's equations**

$$\mathrm{P}_{i \rightarrow n} = p_{i,n} = \frac{|h_{n,i}|}{|h_{1,i}+h_{2,i}+h_{3,i}+h_{4,i}+h_{5,i}+h_{6,i}|}\,, \quad n \in \{1,\ldots,6\}$$

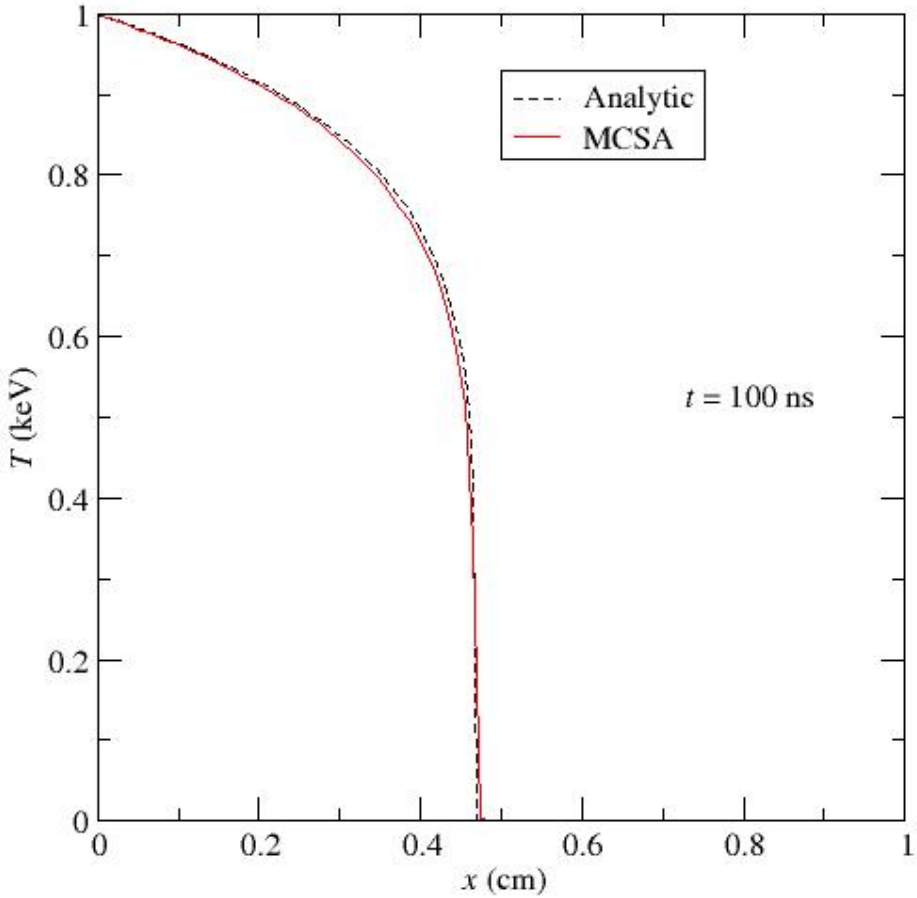
  - **do Monte Carlo transport through the mesh tallying in each cell,**

$$\mathrm{W}_m = \frac{h_{n,i}}{p_{i,n}} W_{m-1}\,, \quad n \in \{1,\ldots,6\}$$

Test Problems

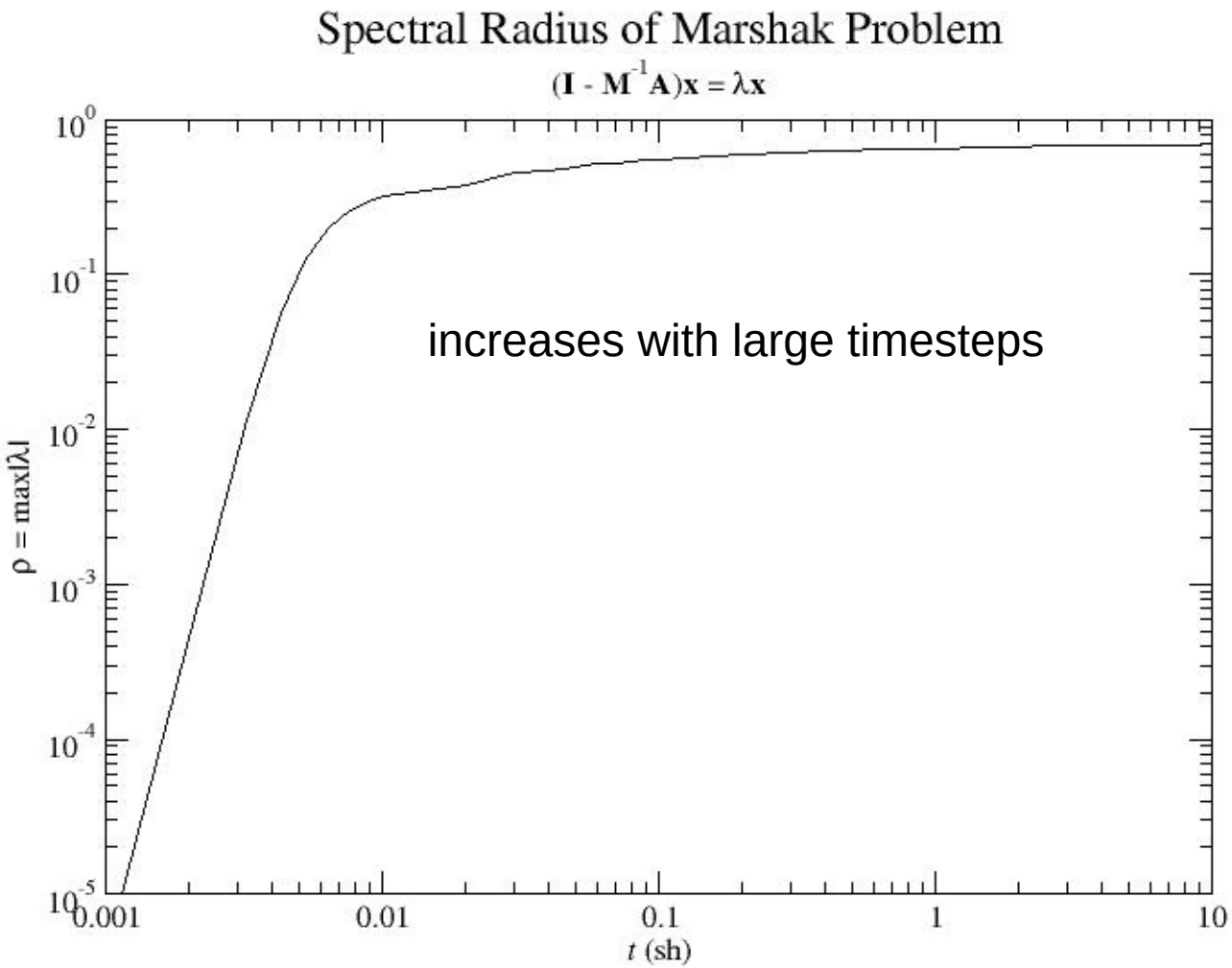
- **Compare with standard deterministic methods:**
  - **Preconditioned Conjugate Gradient (right-left Jacobi) (PCG).**
  - **Preconditioned Richardson (fixed-point) (PFIX).**
- **Two problems:**
  - **3-D Marshak Wave**
  - **multi-material duct**

Marshak Problem



B.C.	$f_b^+(x=0, T=1.0) = \frac{ac}{4}T^4$
	$\mathbf{F} = 0 \quad \partial\Gamma \in \{x^+, y^-, y^+, z^-, z^+\}$
material properties	$\rho = 3.0\,\mathrm{g}\cdot\mathrm{cm}^{-3}$
	$C_v = 0.1\,\mathrm{Jerks}\cdot\mathrm{g}^{-1}\cdot\mathrm{keV}^{-1}$
	$T(t=0) = 1\times10^{-4}\,\mathrm{keV}$
opacity	$\sigma = 100T^{-3}\,\mathrm{cm}^2\cdot\mathrm{g}^{-1}$
mesh	$\Delta_i = \Delta_j = \Delta_k = 0.01\,\mathrm{cm}$
	$N_i = 100 \quad N_j = N_k = 4$

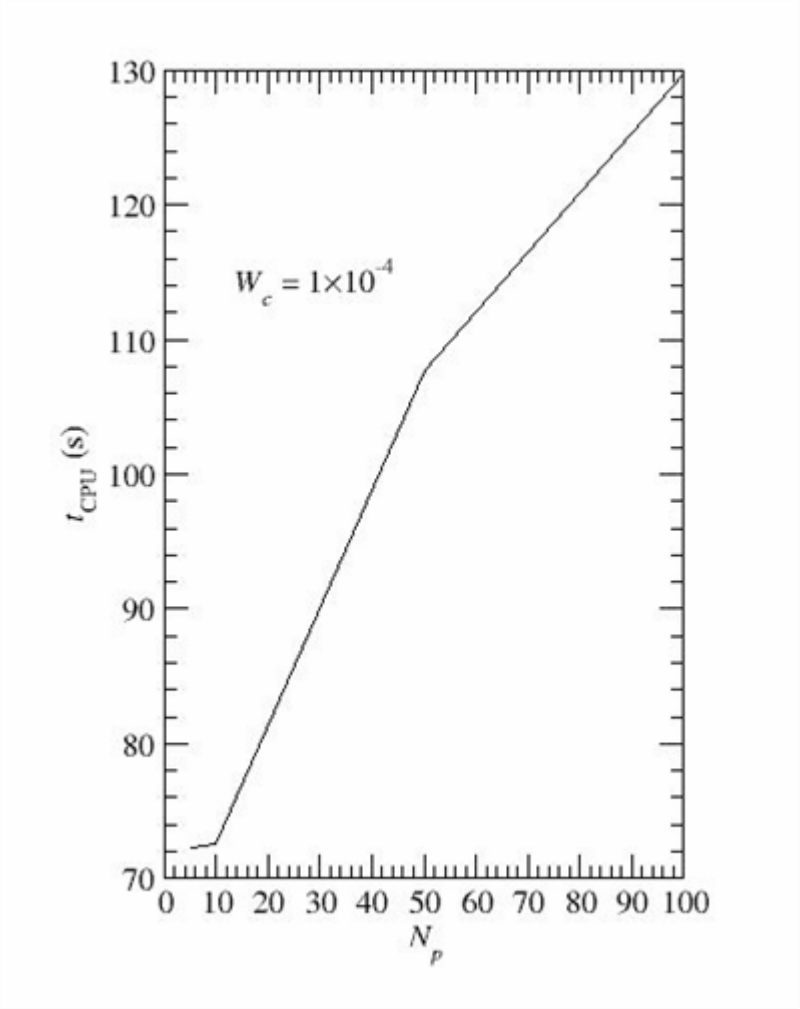
Spectral Radius of Marshak Problem

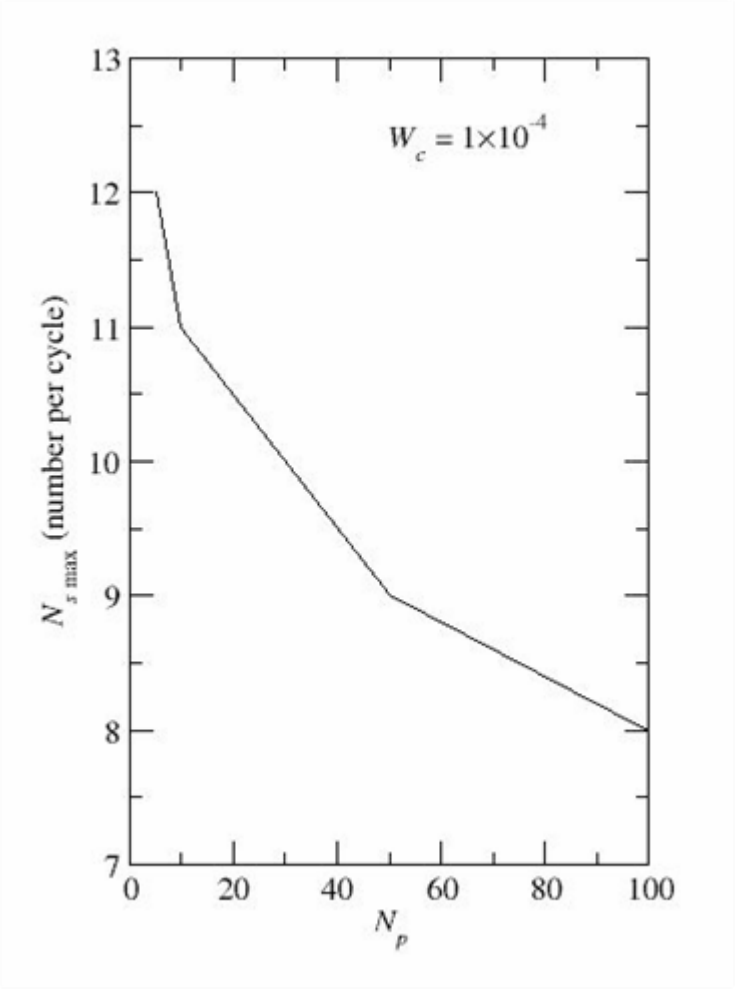


Degrees of Freedom

- **There are two degrees-of-freedom when applying MCSA:**
  - **The number of particles per stage (iteration).**
  - **The weight cutoff.**
- **MCSA uses the same convergence criterion as PCG and PFIX, so numerical accuracy is identical between the methods (within numerical precision).**

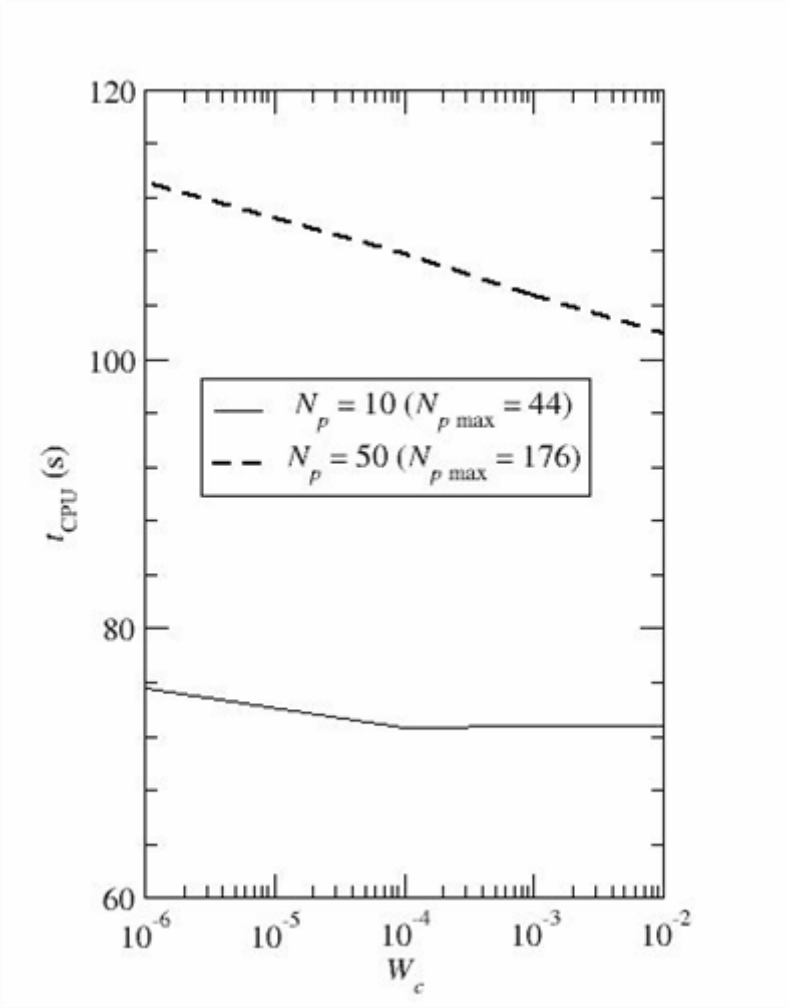
Variation with Number of Particles





Increasing particles per stage reduces iterations at significant cost.

Variation with Weight Cutoff



MCSA is relatively insensitive to weight cutoff.

The source algorithm does a fit to generate the source. This can result in more source particles than requested. The maximum number of particles in any given iteration is shown in parenthesis.

Comparisons with Deterministic Methods

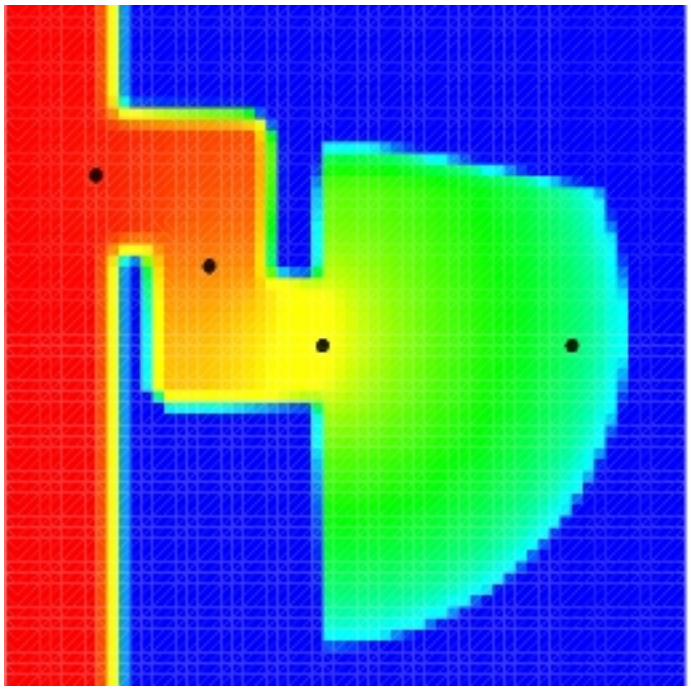
- **Setting the stopping criterion to $1x10^{-8}$ for all methods we have**

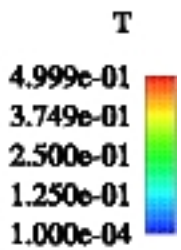
Method	Max Iterations	Relative CPU Time
PCG	11	1.11
PFIX	11	1.42
MCSA	11	1.0

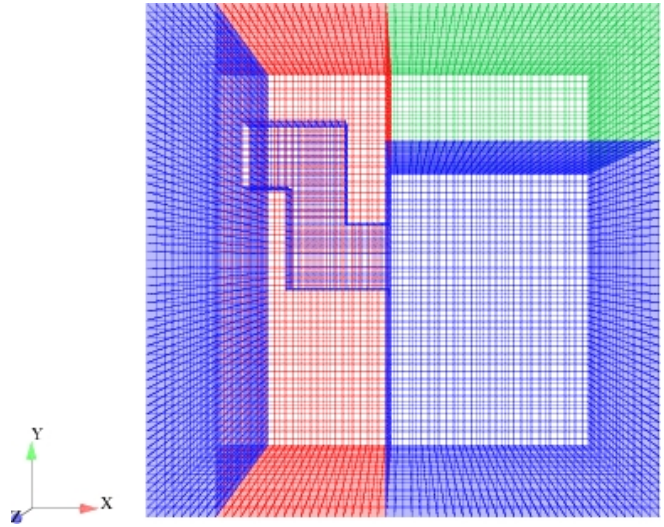
- **MCSA used 10 particles per stage and a weight cutoff of $1x10^{-4}$.**

Multi-Material Problem

B.C.	$f_b^+(x=0, T=0.5) = \frac{ac}{4}T^4$
	$\mathbf{F} = 0 \quad \partial\Gamma \in \{x^+, y^-, y^+, z^-, z^+\}$
material properties	$\rho = 1.5\,\mathrm{g}\cdot\mathrm{cm}^{-3}$ (duct)
	$\rho = 8.0\,\mathrm{g}\cdot\mathrm{cm}^{-3}$ (wall)
	$\rho = 4.0\,\mathrm{g}\cdot\mathrm{cm}^{-3}$ (foil)
	$C_v = 0.1\,\mathrm{Jerks}\cdot\mathrm{g}^{-1}\cdot\mathrm{keV}^{-1}$
	$T(t=0) = 1\times10^{-4}\,\mathrm{keV}$
opacity	$\sigma = T^{-3}\,\mathrm{cm}^2\cdot\mathrm{g}^{-1}$ (duct)
	$\sigma = 1000T^{-1}\,\mathrm{cm}^2\cdot\mathrm{g}^{-1}$ (wall)
	$\sigma = 5T^{-2}\,\mathrm{cm}^2\cdot\mathrm{g}^{-1}$ (foil)
mesh	$\Delta_i = \Delta_j = \Delta_k = 0.025\,\mathrm{cm}$
	$N_i = N_j = N_k = 60$







Output at 1000 ns.

Comparisons with Deterministic Methods

- **Setting the stopping criterion to $1x10^{-8}$ for all methods we have**

Method	Max Iterations	Relative CPU Time
PCG	18	1.03
PFIX	38	1.43
MCSA	20	1.0

- **MCSA used 1000 particles per stage and a weight cutoff of $1x10^{-3}$.**
- **Problem run to an elapsed time of 100 ns.**

Conclusions

- **MCSA is competitive with PCG on 3-D, time-dependent problems.**
- **MCSA is not efficient in non-sparse systems. (Too many terms in the Neumann series)**
- **MCSA is well-suited to time-dependent problems because the state at tn provides a reasonable estimate of the residual.**

Nonlinear-Consistent Methods

- **MCSA may have significant benefits in fully-nonlinear, time-dependent problems:**

$$\mathbf{J}\delta\mathbf{u}^k = -\mathbf{f}(\mathbf{u}^k) \qquad \mathbf{J} = \frac{\partial \mathbf{f}(\mathbf{u}^\mathbf{k})}{\partial \mathbf{u}^k} \qquad \mathbf{u}^{k+1} = \mathbf{u}^k + \delta\mathbf{u}^k$$

- **Newton-methods will convert SPD operators into non-symmetric matrices.**
- **The traditional linear solver for these systems is GMRES. MCSA may be very competitive with GMRES for non-symmetric systems.**

Adaptive Meshing

- **Adaptive meshing can provide significant accuracy gains in certain classes of problems.**
- **Generally, adaptive mesh systems are characterized by poor condition numbers due to varying volumetric effects.**
- **MCSA, or modified MCSA, may be efficient on poorly conditioned matrices.**