A Spectral Analysis of the Domain Decomposed Monte Carlo Method for Linear Systems

Stuart R. Slattery and Paul P.H. Wilson, University of Wisconsin - Madison

Thomas M. Evans, Oak Ridge National Laboratory

May 8, 2013

Motivation

- First proposed by J. Von Neumann and S.M. Ulam in the 1940's
- General lack of published work
- Recent work by Evans and others has yielded new potential applications
- Implications for resilient exascale solver strategies
- Domain decomposed parallelism has yet to be exploited - would like a preliminary analytic framework

Monte Carlo Linear Solver Preliminaries

- Split the linear operator

$$\mathbf{H} = \mathbf{I} - \mathbf{A}$$

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \rightarrow \quad \mathbf{x} = \mathbf{H}\mathbf{x} + \mathbf{b}$$

- Generate the *Neumann series*

$$\mathbf{A}^{-1} = (\mathbf{I} - \mathbf{H})^{-1} = \sum_{k=0}^{\infty} \mathbf{H}^k$$

- Require $\rho(\mathbf{H}) < 1$ for convergence

$$\mathbf{A}^{-1}\mathbf{b} = \sum_{k=0}^{\infty} \mathbf{H}^k \mathbf{b} = \mathbf{x}$$

Monte Carlo Linear Solver Preliminaries

- Expand the Neumann series

$$x_i = \sum_{k=0}^{\infty} \sum_{i_1}^{N} \sum_{i_2}^{N} \dots \sum_{i_k}^{N} h_{i,i_1} h_{i_1,i_2} \dots h_{i_{k-1},i_k} b_{i_k}$$

- Define a sequence of state transitions

$$\nu = i \rightarrow i_1 \rightarrow \cdots \rightarrow i_{k-1} \rightarrow i_k$$

- Use the adjoint Neumann-Ulam decomposition[1]

$$\mathbf{H}^T = \mathbf{P} \circ \mathbf{W}$$

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

[1] The Hadamard product $\mathbf{A} = \mathbf{B} \circ \mathbf{C}$ is defined element-wise as $a_{ij} = b_{ij} c_{ij}$.

Adjoint Method

- Build the estimator and expectation value

$$X_j(\nu) = \sum_{m=0}^{k} W_m \delta_{i_m,j}$$

$$E\{X_j\} = \sum_{\nu} P_\nu X_\nu = \sum_{k=0}^{\infty} \sum_{i_1}^{N} \sum_{i_2}^{N} \dots \sum_{i_k}^{N} b_{i_0} h_{i_0,i_1} h_{i_1,i_2} \dots h_{i_{k-1},i_k} \delta_{i_k,j}$$
$$= x_j$$

- Terminate a random walk below a weight cutoff, $W_m < W_c$

Adjoint Method: Evolution of a Solution

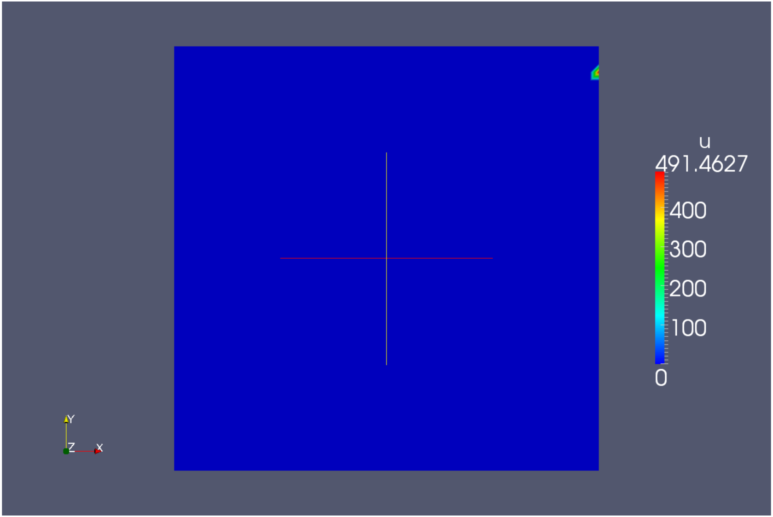


Figure: **Adjoint solution to Poisson Equation.** 1×10^0 *total histories, 0.286 seconds CPU time.*

Adjoint Method: Evolution of a Solution

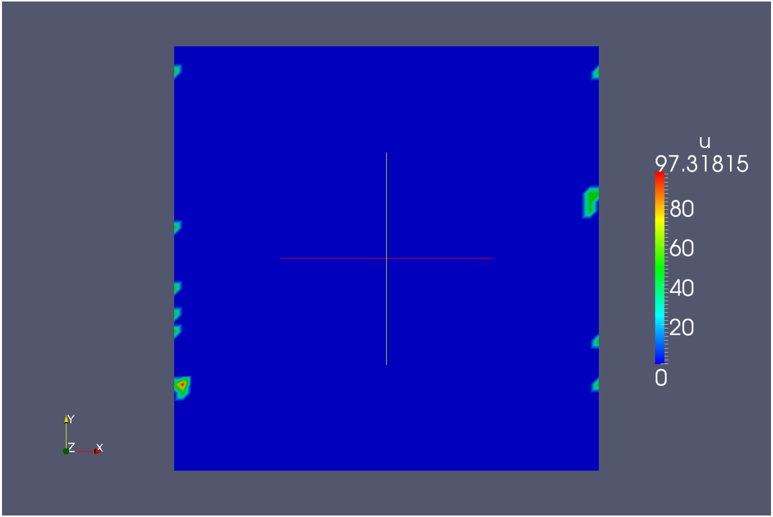


Figure: **Adjoint solution to Poisson Equation.** 1×10^1 *total histories, 0.278 seconds CPU time.*

Adjoint Method: Evolution of a Solution

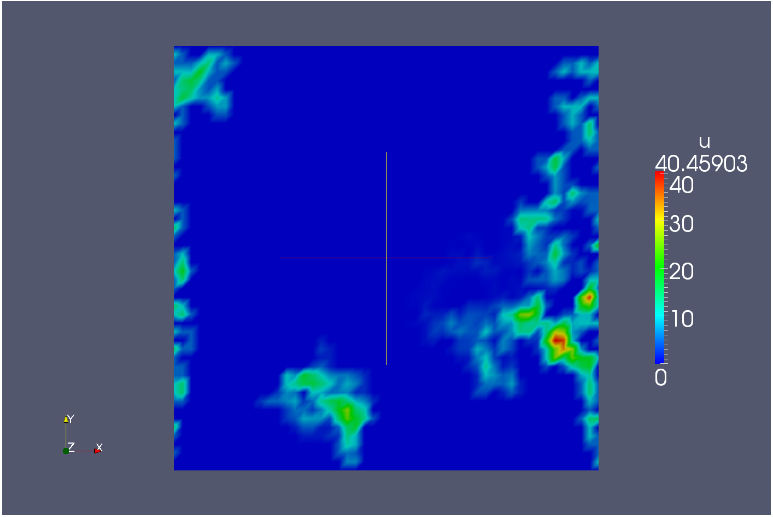


Figure: **Adjoint solution to Poisson Equation.** 1×10^2 *total histories, 0.275 seconds CPU time.*

Adjoint Method: Evolution of a Solution

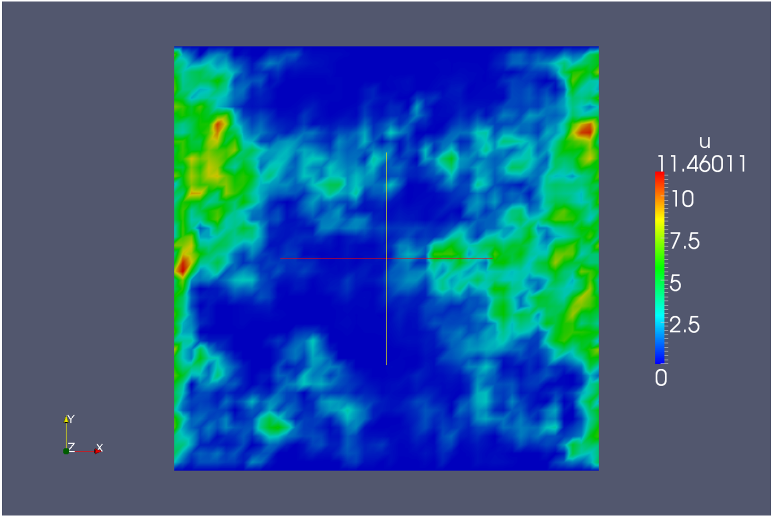


Figure: **Adjoint solution to Poisson Equation.** 1×10^3 *total histories, 0.291 seconds CPU time.*

Adjoint Method: Evolution of a Solution

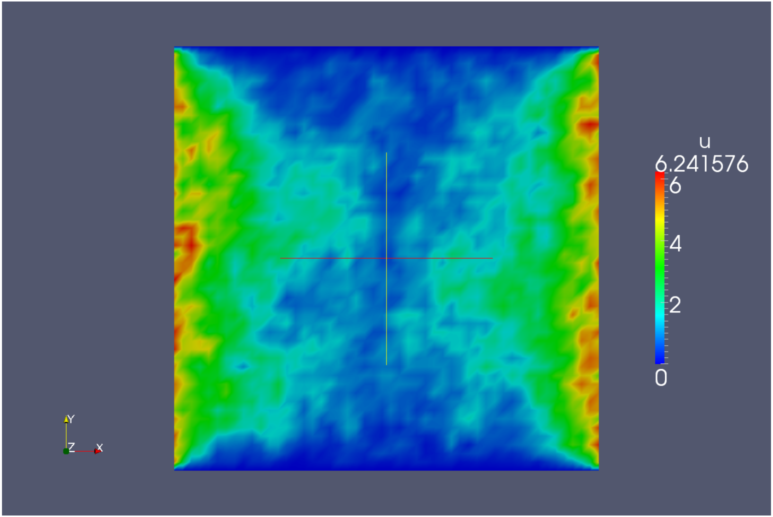


Figure: **Adjoint solution to Poisson Equation.** 1×10^4 *total histories, 0.428 seconds CPU time.*

Adjoint Method: Evolution of a Solution

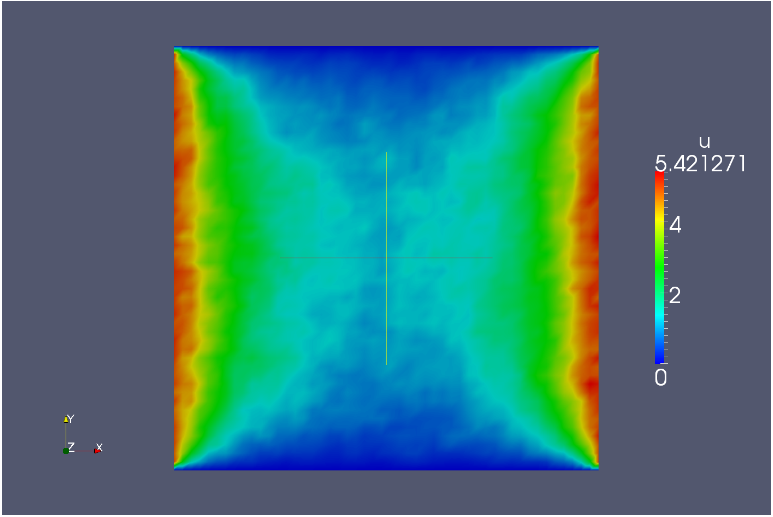


Figure: **Adjoint solution to Poisson Equation.** 1×10^5 *total histories, 1.76 seconds CPU time.*

Adjoint Method: Evolution of a Solution

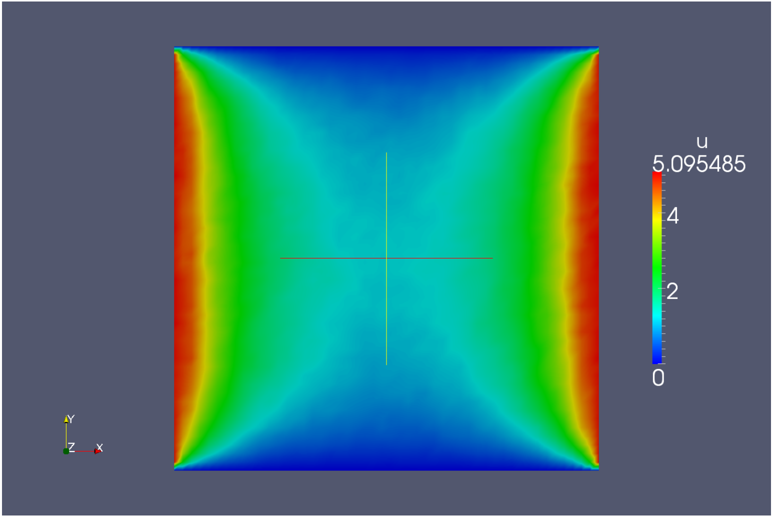


Figure: **Adjoint solution to Poisson Equation.** 1×10^6 *total histories, 15.1 seconds CPU time.*

Adjoint Method: Evolution of a Solution

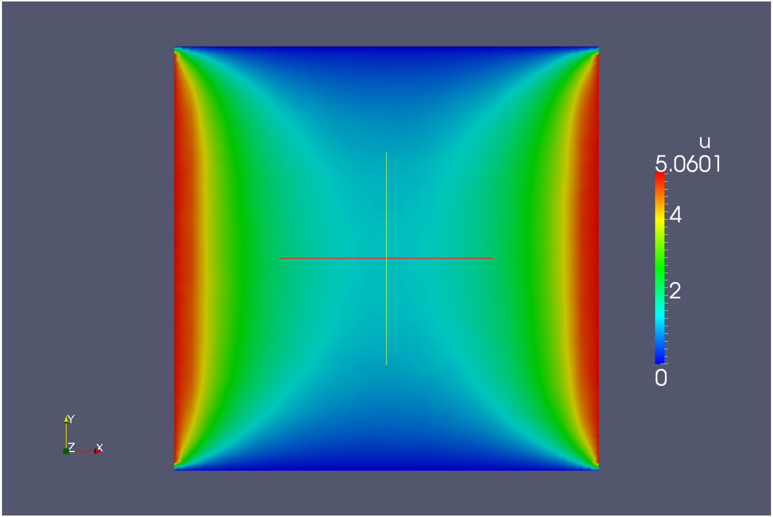


Figure: **Adjoint solution to Poisson Equation.** 1×10^7 *total histories, 149 seconds CPU time.*

Domain Decomposed Monte Carlo

- Each parallel process owns a piece of the domain (linear system)
- Random walks must be transported between adjacent domains through parallel communication
- Domain decomposition determined by the input system

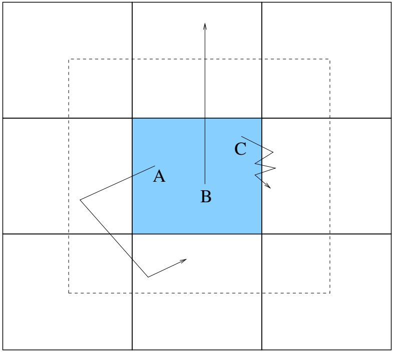


Figure: **Domain decomposition example illustrating how domain-to-domain transport creates communication costs.**

Model Problem - 2D Neutron Diffusion

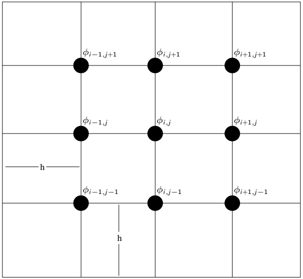


Figure: **Nine-point Laplacian stencil.**

$$-D\nabla^2\phi + \Sigma_a\phi = S$$

$$\mathbf{D}\boldsymbol{\phi} = \mathbf{s}$$

$$\begin{aligned}-\frac{1}{6h^2}[4\phi_{i-1,j} + 4\phi_{i+1,j} + 4\phi_{i,j-1} + 4\phi_{i,j+1} + \phi_{i-1,j-1} \\ + \phi_{i-1,j+1} + \phi_{i+1,j-1} + \phi_{i+1,j+1} - 20\phi_{i,j}] + \Sigma_a\phi_{i,j} = s_{i,j} \quad (1)\end{aligned}$$

Eigenvalue Computation

$$\Phi_{p,q}(x,y) = e^{2\pi\imath px} e^{2\pi\imath qy}$$

$$\begin{aligned}
\mathbf{D}\Phi_{p,q}(x,y) = \lambda_{p,q}(\mathbf{D}) = \\
-\frac{D}{6h^2}\Big[4e^{-2\pi\imath ph} + 4e^{2\pi\imath ph} + 4e^{-2\pi\imath qh} + 4e^{2\pi\imath qh} + e^{-2\pi\imath ph}e^{-2\pi\imath qh} \\
+ e^{-2\pi\imath ph}e^{2\pi\imath qh} + e^{2\pi\imath ph}e^{-2\pi\imath qh} + e^{2\pi\imath ph}e^{2\pi\imath qh} - 20\Big] + \Sigma_a \quad (2)
\end{aligned}$$

$$\lambda_{p,q}(\mathbf{D}) = -\frac{D}{6h^2}[8\cos(\pi ph) + 8\cos(\pi qh) + 4\cos(\pi ph)\cos(\pi qh) - 20] + \Sigma_a$$

Jacobi Preconditioned Iteration Matrix Spectrum

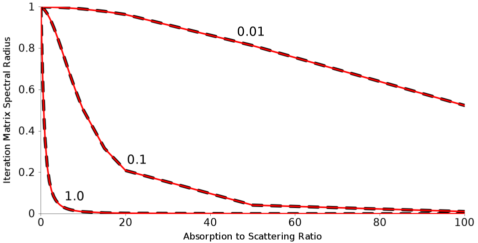


Figure: **Measured and analytic preconditioned diffusion operator spectral radius as a function of the absorption cross section to scattering cross section ratio.**

$$\mathbf{M}^{-1}\mathbf{D}\phi = \mathbf{M}^{-1}\mathbf{s}$$

$$\lambda_{p,q}(\mathbf{M}^{-1}\mathbf{D}) = \alpha\lambda_{p,q}(\mathbf{D})$$

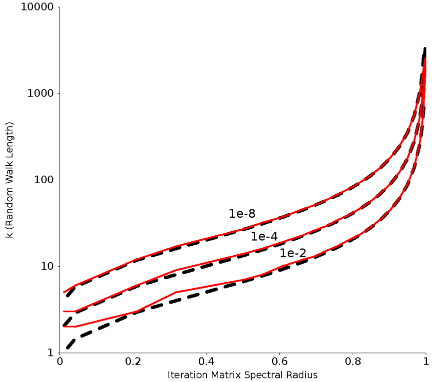
$$\alpha = \left[\frac{10D}{3h^2} + \Sigma_a\right]^{-1}$$

$$\mathbf{H} = \mathbf{I} - \mathbf{M}^{-1}\mathbf{D}$$

$$\rho(\mathbf{H}) = \frac{10\alpha D}{3h^2}$$

Random Walk Length

$$\mathbf{e}^k = \mathbf{H}^k \mathbf{e}^0 \quad \rightarrow \quad ||\mathbf{e}^k||_2 \leq \rho(\mathbf{H})^k ||\mathbf{e}^0||_2 \quad \rightarrow \quad k \approx \frac{\log(W_c)}{\log(\rho(\mathbf{H}))}$$



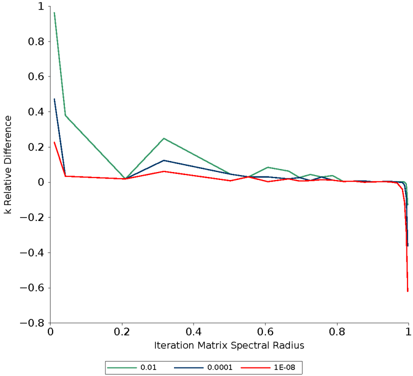


Figure: **Measured and analytic random walk length as a function of the iteration matrix spectral radius.**

Linear Operator Domain Optical Thickness

$$\langle \bar{r_1^2} \rangle = (n_s h)^2 \quad \rightarrow \quad \langle \bar{r_k^2} \rangle = k(n_s h)^2$$

- l = chord length
- n_i = # of discrete states along the chord
- n_s = # of discrete states per transition along the chord
- d = dimensionality of problem

$$\langle \bar{r_k^2} \rangle = k\left(\frac{n_s l}{n_i}\right)^2$$

$$\tau = \frac{l}{2d\sqrt{\langle \bar{r_k^2} \rangle}}$$

$$\tau = \frac{n_i}{2dn_s\sqrt{k}}$$

$$\tau = \frac{n_i}{2dn_s}\sqrt{\frac{\log(\rho(\mathbf{H}))}{\log(W_c)}}$$

Choosing d, n_i, and n_s

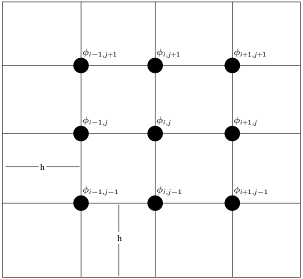


Figure: **Nine-point Laplacian stencil.**

$$d = 2 \quad (i, j)$$

$$n_i = 5$$

$$n_s = \frac{3}{5}$$

$$-\frac{1}{6h^2}[4\phi_{i-1,j} + 4\phi_{i+1,j} + 4\phi_{i,j-1} + 4\phi_{i,j+1} + \phi_{i-1,j-1} + \phi_{i-1,j+1} + \phi_{i+1,j-1} + \phi_{i+1,j+1} - 20\phi_{i,j}] + \Sigma_a \phi_{i,j} = s_{i,j} \quad (3)$$

Domain Leakage Approximations

$$\Lambda = \frac{\text{average \# of histories leaving local domain}}{\text{total of \# of histories starting in local domain}}$$

Wigner Rational Approximation

$$\Lambda = \frac{1}{1+\tau}$$

Mean Chord Approximation

$$\Lambda = \frac{1-e^{-\tau}}{\tau}$$

Domain Leakage Results

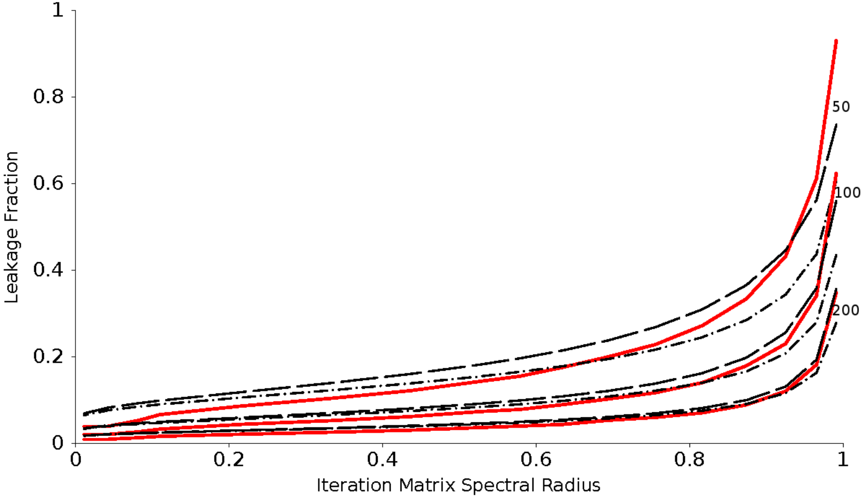


Figure: **Measured and analytic domain leakage as a function of the iteration matrix spectral radius.**

Domain Leakage Results

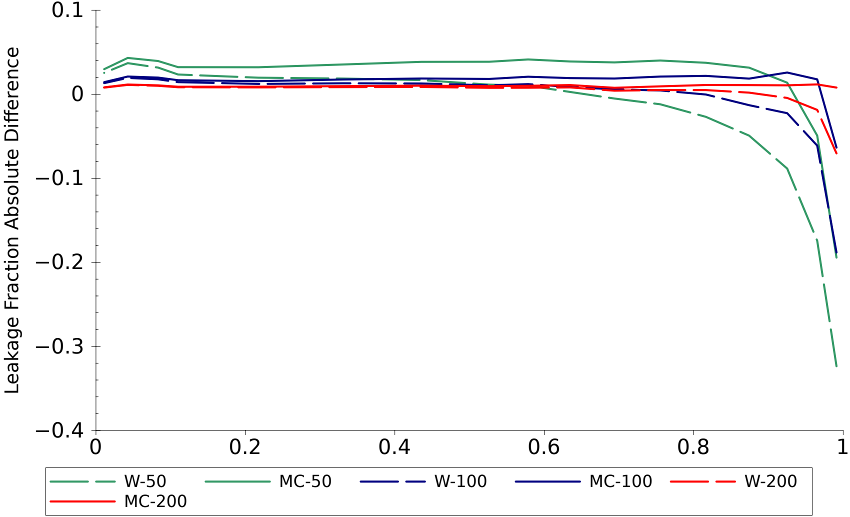


Figure: **Measured and analytic domain leakage absolute error as a function of the iteration matrix spectral radius.**

Conclusions and Future Work

- Good agreement between theory and numerical experiments
- Extension to asymmetric systems and communication cost analysis
- Coordinate measurements with massively parallel computations
- Explore in the context of synthetic acceleration methods

Acknowledgments

This work was performed under appointment to the Nuclear Regulatory Commission Fellowship program at the University of Wisconsin - Madison Engineering Physics Department.