Domain Decomposition Methods for the Neutron Transport Equation

Jack Blake, Ivan Graham and Alastair Spence

Mathematical Sciences,
Bath University, Bath, BA2 7AY

June, 2015

Map of the talk

- Project aimed at developing and understanding numerical methods in radiative transport.
- Focus of this talk is the mono-energetic steady-state transport equation. (Equation on next slide)
- This talk is split into three parts:
    1. Source Iteration for the transport equation
    2. Benefits and limitations of diffusion synthetic acceleration,
    3. A domain decomposition approach to solving the transport equation.
- Motivation: domain decomposition methods have good parallelisation potential and can help improve convergence rate whilst limiting computational expense.

The problem

The (mono-energetic, steady-state) neutron transport equation in 5D with an isotropic source, $Q(\mathbf{r})$, is given by

$$\Omega \cdot \nabla \psi(\mathbf{r}, \Omega) + \sigma_T(\mathbf{r}) \psi(\mathbf{r}, \Omega) = \frac{\sigma_S(\mathbf{r})}{4\pi} \int_{\mathbb{S}^2} \psi(\mathbf{r}, \Omega') \, \mathrm{d}\Omega' + Q(\mathbf{r})$$

with $\mathbf{r} \in V \subset \mathbb{R}^3$ and $\Omega \in \mathbb{S}^2$. The argument $\psi(\mathbf{r}, \Omega)$ is called the neutron flux.

The problem

The (mono-energetic, steady-state) neutron transport equation in 5D with an isotropic source, $Q(\mathbf{r})$, is given by

$$\Omega \cdot \nabla \psi(\mathbf{r}, \Omega) + \sigma_T(\mathbf{r}) \psi(\mathbf{r}, \Omega) = \frac{\sigma_S(\mathbf{r})}{4\pi} \int_{\mathbb{S}^2} \psi(\mathbf{r}, \Omega') \, \mathrm{d}\Omega' + Q(\mathbf{r})$$

with $\mathbf{r} \in V \subset \mathbb{R}^3$ and $\Omega \in \mathbb{S}^2$. The argument $\psi(\mathbf{r}, \Omega)$ is called the neutron flux. Boundary conditions

$$\psi(\mathbf{r}, \Omega) = 0, \quad \text{when} \quad n(\mathbf{r}) \cdot \Omega < 0, \quad \mathbf{r} \in \delta V.$$

Note: σ_T, σ_S and σ_A are called cross-sections. They are all strictly positive and satisfy $\sigma_T = \sigma_S + \sigma_A$

Source Iteration (described via operator notation)

Introduce

$$\mathcal{T}(\cdot) \equiv \Omega \cdot \nabla(\cdot) + \sigma_T(\mathbf{r})(\cdot) \quad \text{and define} \quad \phi(\mathbf{r}) = \frac{1}{4\pi}\int_{\mathbb{S}^2} \psi(\mathbf{r}, \Omega)\, \mathrm{d}\Omega$$

ϕ is called the scalar flux. The transport equation is then

$$\mathcal{T}\psi = \sigma_S \phi + Q$$

Source Iteration (described via operator notation)

Introduce

$$\mathcal{T}(\cdot) \equiv \Omega \cdot \nabla(\cdot) + \sigma_T(\mathbf{r})(\cdot) \quad \text{and define} \quad \phi(\mathbf{r}) = \frac{1}{4\pi}\int_{\mathbb{S}^2} \psi(\mathbf{r}, \Omega)\, \mathrm{d}\Omega$$

ϕ is called the scalar flux. The transport equation is then

$$\mathcal{T}\psi = \sigma_S \phi + Q$$

Source Iteration (SI) is defined as follows

$$\begin{aligned} \mathcal{T}\psi^{(k+1)} &= \sigma_S \phi^{(k)} + Q \\ \phi^{(k+1)} &= \tfrac{1}{4\pi} \int_{\mathbb{S}^2} \psi^{(k+1)}\, \mathrm{d}\Omega \end{aligned}$$

This basic iterative method is known to converge since

$$\|\sigma_S/\sigma_T\|_\infty < 1$$

Diffusion Approximation

Limitation of source iteration:

Potentially slow convergence when $\|\sigma_S/\sigma_T\|_\infty \approx 1$

Diffusion Approximation

Limitation of source iteration:
Potentially slow convergence when $\|\sigma_S/\sigma_T\|_\infty \approx 1$

One approach:
Approximate ϕ (the scalar flux) using a diffusion equation.

$$-\nabla \cdot \left(\frac{1}{3\sigma_T(\mathbf{r})} \nabla \Theta(\mathbf{r}) \right) + \sigma_A(\mathbf{r})\Theta(\mathbf{r}) = Q(\mathbf{r}),$$

subject to

$$\Theta(\mathbf{r}) + \lambda n(\mathbf{r}) \cdot \nabla \Theta(\mathbf{r}) = 0, \quad \text{when} \quad \mathbf{r} \in \delta V,$$

with λ a known constant.
Using asymptotics: $\Theta = \phi + \mathcal{O}(\epsilon^2)$, where ϵ is an asymptotic parameter.

Diffusion Synthetic Acceleration (DSA)

Roughly speaking, as $\|\sigma_S/\sigma_T\|_\infty \to 1$:

- Source iteration converges more slowly,
- Θ better approximates ϕ.

Diffusion Synthetic Acceleration (DSA)

Roughly speaking, as $\|\sigma_S/\sigma_T\|_\infty \to 1$:

- Source iteration converges more slowly,
- Θ better approximates ϕ.

Idea: use the good diffusion approximation to "accelerate" the poor convergence of source iteration.

Diffusion Synthetic Acceleration (DSA)

Roughly speaking, as $\|\sigma_S/\sigma_T\|_\infty \to 1$:

- Source iteration converges more slowly,
- Θ better approximates ϕ.

Idea: use the good diffusion approximation to "accelerate" the poor convergence of source iteration.

Synthetic acceleration methods were first suggested by Kopp in 1963. DSA is such a method.
2-step process:

1. Do one step of source iteration,
2. use the diffusion approximation to estimate the error in step 1.

DSA: The Good and the Bad

Good:

DSA improves upon or maintains the convergence of source iteration for all values of σ_S/σ_T

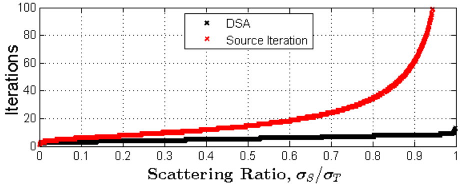


Figure: Iterations to converge to a tolerance of 10^{-6}

DSA: The Good and the Bad

Good:

DSA improves upon or maintains the convergence of source iteration for all values of σ_S/σ_T

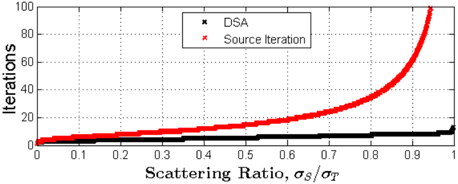


Figure: Iterations to converge to a tolerance of 10^{-6}

Bad:

- Discontinuous cross-sections can lead to degraded effectiveness of multidimensional DSA (Azmy, 1998, Warsa et. al., 2004).
- Higher computational cost per iteration.

Domain Decomposition approaches

Idea: separate the domain into *diffusive* and *non-diffusive* regions.

Diffusive $\rightarrow$ apply DSA
Non-diffusive $\rightarrow$ apply SI

Domain Decomposition approaches

Idea: separate the domain into *diffusive* and *non-diffusive* regions.

Diffusive $\rightarrow$ apply DSA
Non-diffusive $\rightarrow$ apply SI

First things first: how do we organise a *domain decomposed source iteration* algorithm?

Domain Decomposition approaches

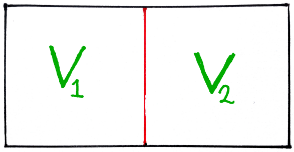
Idea: separate the domain into *diffusive* and *non-diffusive* regions.

Diffusive $\rightarrow$ apply DSA

Non-diffusive $\rightarrow$ apply SI

First things first: how do we organise a *domain decomposed source iteration* algorithm?

- Internal boundary,
- External boundary.



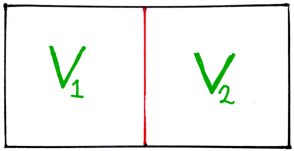
Domain Decomposition approaches

Idea: separate the domain into *diffusive* and *non-diffusive* regions.

Diffusive $\rightarrow$ apply DSA
Non-diffusive $\rightarrow$ apply SI

First things first: how do we organise a *domain decomposed source iteration* algorithm?

– Internal boundary,
– External boundary.



Boundary conditions:

- Zero incoming flux on the external boundary,
- **Internal boundary?**

In general: use the flux on V_1 to impose internal conditions for V_2,
use the flux on V_2 to impose internal conditions for V_1.

Two Domain Decomposition approaches

DDSI $\equiv$ Domain Decomposed Source Iteration

Jacobi DDSI:
Internal boundary conditions for each subdomain are imposed using the **previous** iteration on neighbouring subdomains.

Gauss-Seidel DDSI:
Internal boundary conditions for each subdomain are imposed using the **current** iteration on neighbouring subdomains.
Proved: Gauss-Seidel DDSI $\equiv$ Source Iteration

Two Domain Decomposition approaches

DDSI $\equiv$ Domain Decomposed Source Iteration

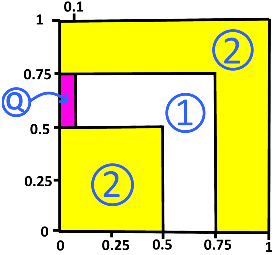
Jacobi DDSI:
Internal boundary conditions for each subdomain are imposed using the **previous** iteration on neighbouring subdomains.

Gauss-Seidel DDSI:
Internal boundary conditions for each subdomain are imposed using the **current** iteration on neighbouring subdomains.
Proved: Gauss-Seidel DDSI $\equiv$ Source Iteration

Algorithm:	**Parallelisation Potential**	**Rate of convergence**	**Arbitrary subdomain shapes**
Jacobi DDSI	Angle & space	Slower than SI	✓
Gauss-Seidel DDSI	Angle only	Same as SI	✗

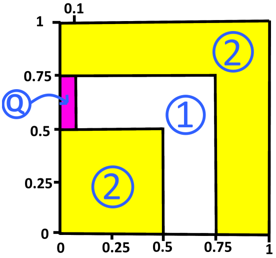
Numerical Results

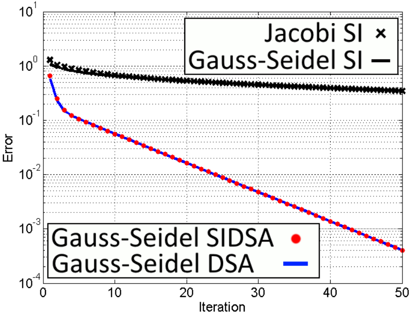
$$\textcolor{blue}{(1)}\text{: Diffusive}$$
$$\textcolor{blue}{(2)}, \textcolor{blue}{(Q)}\text{: Non-diffusive}$$



Numerical Results

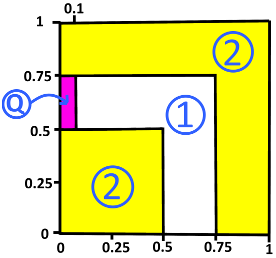
(1): Diffusive
(2), (Q): Non-diffusive

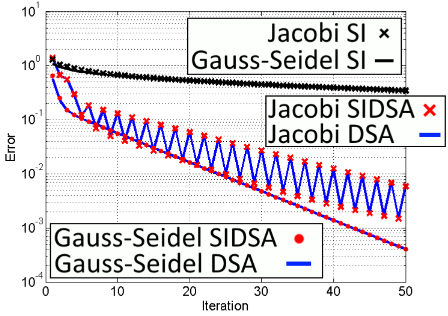




Numerical Results

(1): Diffusive
(2), (Q): Non-diffusive





Method	Time
Jacobi SI	758
Gauss-Seidel SI	748
Jacobi DSA	2155
Gauss-Seidel DSA	2028
Jacobi SIDSA	1182
Gauss-Seidel SIDSA	1123