BART
A new framework for developing and evaluating acceleration schemes for the neutron transport equation

J. S. Rehak

Qualification Exam
September 4th, 2019

2019-08-29

Qualification Exam

BART
A new framework for developing and evaluating acceleration schemes for the neutron transport equation
J. S. Rehak
Qualification Exam
September 4th, 2019

Outline

1. Transport Equation
2. Acceleration Methods
3. Analyzing Acceleration Schemes
4. BART
5. Plan and Future Work

Outline
1. Transport Equation
2. Acceleration Methods
3. Analyzing Acceleration Schemes
4. BART
5. Plan and Future Work

Transport Equation

Transport Equation

Steady-state Boltzman Transport Equation

Our problem of interest is the time-independent transport equation on a domain of interest $\mathbf{r} \in V$ [3],

$$\begin{aligned}\left[\hat{\Omega} \cdot \nabla + \Sigma_t(\mathbf{r}, E)\right] \psi(\mathbf{r}, E, \hat{\Omega}) \\ &= \int_0^\infty dE' \int_{4\pi} d\hat{\Omega}' \Sigma_s(\mathbf{r}, E' \to E, \hat{\Omega}' \to \hat{\Omega}) \psi(\mathbf{r}, E', \hat{\Omega}') \\ &+ Q(\psi, \mathbf{r}, E, \hat{\Omega}) \ ,\end{aligned}$$

with a given boundary condition,

$$\psi(\mathbf{r}, E, \hat{\Omega}) = \Gamma(\mathbf{r}, E, \hat{\Omega}), \quad \mathbf{r} \in \partial V, \quad \hat{\Omega} \cdot \hat{n} < 0$$

Steady-state Boltzman Transport Equation

Our problem of interest is the time-independent transport equation on a domain of interest $\mathbf{r} \in V$ [3],

$$\begin{aligned}\left[\hat{\Omega} \cdot \nabla + \Sigma_t(\mathbf{r}, E)\right] \psi(\mathbf{r}, E, \hat{\Omega}) \\ &= \int_0^\infty dE' \int_{4\pi} d\hat{\Omega}' \Sigma_s(\mathbf{r}, E' \to E, \hat{\Omega}' \to \hat{\Omega}) \psi(\mathbf{r}, E', \hat{\Omega}') \\ &+ Q(\psi, \mathbf{r}, E, \hat{\Omega}) \ ,\end{aligned}$$

with a given boundary condition,

$$\psi(\mathbf{r}, E, \hat{\Omega}) = \Gamma(\mathbf{r}, E, \hat{\Omega}), \quad \mathbf{r} \in \partial V, \quad \hat{\Omega} \cdot \hat{n} < 0$$

Iterative Solving Method

Assuming the source Q is not a function of ψ, we define the source-iteration iterative scheme for iteration i,

$$\left[\hat{\Omega}\cdot\nabla + \Sigma_t(\mathbf{r},E)\right]\psi^{i+1}(\mathbf{r},E,\hat{\Omega}) = \int_0^\infty dE' \int_{4\pi} d\hat{\Omega}'\Sigma_s(\mathbf{r},E'\to E,\hat{\Omega}'\to\hat{\Omega})\psi^i(\mathbf{r},E',\hat{\Omega}') + Q(\mathbf{r},E,\hat{\Omega})\ ,$$

with the same boundary condition and initial condition $\psi^0(\mathbf{r},E,\hat{\Omega})$.

Iterative Solving Method

Assuming the source Q is not a function of ψ, we define the source-iteration iterative scheme for iteration i,

$$\left[\hat{\Omega}\cdot\nabla + \Sigma_t(\mathbf{r},E)\right]\psi^{i+1}(\mathbf{r},E,\hat{\Omega}) = \int_0^\infty dE' \int_{4\pi} d\hat{\Omega}'\Sigma_s(\mathbf{r},E'\to E,\hat{\Omega}'\to\hat{\Omega})\psi^i(\mathbf{r},E',\hat{\Omega}') + Q(\mathbf{r},E,\hat{\Omega})\ ,$$

with the same boundary condition and initial condition $\psi^0(\mathbf{r},E,\hat{\Omega})$.

Fourier analysis

To see how the error in our iterative schemes evolves, we can use Fourier analysis [2]. To do so, we use a one-group, one dimension, infinite homogeneous medium with isotropic scattering.

$$\mu\frac{\partial}{\partial x}\psi(x,\mu) + \Sigma_t\psi(x,\mu) = \frac{\Sigma_s}{2}\int_{-1}^{1}\psi(x,\mu')d\mu' + \frac{Q}{2}\ .$$

We define the source iteration scheme as discussed above,

$$\mu\frac{\partial}{\partial x}\psi^{k+1}(x,\mu) + \Sigma_t\psi^{k+1}(x,\mu) = \frac{\Sigma_s}{2}\int_{-1}^{1}\psi^{k}(x,\mu')d\mu' + \frac{Q}{2}\ .$$

and subtract the two to get an equation for the error in iteration k, giving us a similar equation for the error in iteration $k+1$ as it relates to the error in iteration k,

$$\mu\frac{\partial}{\partial x}e^{k+1}(x,\mu) + \Sigma_t e^{k+1}(x,\mu) = \frac{\Sigma_s}{2}\int_{-1}^{1}e^{k}(x,\mu')d\mu'\ .$$

Fourier analysis
To see how the error in our iterative schemes evolves, we can use Fourier analysis [2]. To do so, we use a one-group, one dimension, infinite homogeneous medium with isotropic scattering.
$$\mu\frac{\partial}{\partial x}\psi(x,\mu) + \Sigma_t\psi(x,\mu) = \frac{\Sigma_s}{2}\int_{-1}^{1}\psi(x,\mu')d\mu' + \frac{Q}{2}\ .$$
We define the source iteration scheme as discussed above,
$$\mu\frac{\partial}{\partial x}\psi^{k+1}(x,\mu) + \Sigma_t\psi^{k+1}(x,\mu) = \frac{\Sigma_s}{2}\int_{-1}^{1}\psi^{k}(x,\mu')d\mu' + \frac{Q}{2}\ .$$
and subtract the two to get an equation for the error in iteration k, giving us a similar equation for the error in iteration $k+1$ as it relates to the error in iteration k,
$$\mu\frac{\partial}{\partial x}e^{k+1}(x,\mu) + \Sigma_t e^{k+1}(x,\mu) = \frac{\Sigma_s}{2}\int_{-1}^{1}e^{k}(x,\mu')d\mu'\ .$$

- How can we be sure that source iteration will converge? What controls the convergence rate? To determine this we can use a Fourier analysis.
- We need to start with a lot of assumptions to get a very simplified version of our transport equation.
- We define what we mean by error, and get an equation that relates the error in each step to the previous step. Unsurprisingly it looks like our original equation, because the evolution of the solution and the evolution of the error are related.

Fourier Analysis

To perform an inverse Fourier transform, we need to choose a measure of spatial variation, an error “wavelength.”

Fourier Analysis
To perform an inverse Fourier transform, we need to choose a measure of spatial variation, an error “wavelength.”

- We can examine the modes of the spatial error by using an inverse Fourier transform. This will give us an idea of how the spatial frequencies of the error. We need to decide on an error wavelength, which gives us a linear error frequency. Higher n means higher error frequency, with $n = 0$ being infinite wavelength, completely non-coupled error.
- If we plug this back into our previous equation and do a large amount of manipulation, we get a fairly simple relationship between the integrated error in one step to the integrated error in the previous step.
- This lambda function is maximized when $n = 0$. The lowest frequency error converges the slowest, and at a rate proportional to Σ_s/Σ_t.

Fourier Analysis

To perform an inverse Fourier transform, we need to choose a measure of spatial variation, an error "wavelength."

$$\lambda = \frac{\ell}{n}, \quad \forall n \in \mathbb{R} \implies \tilde{\nu} = \frac{1}{\lambda} = \frac{n}{\ell} = n \cdot \Sigma_t$$

Fourier Analysis
To perform an inverse Fourier transform, we need to choose a measure of spatial variation, an error "wavelength."
$$\lambda = \frac{\ell}{n}, \quad \forall n \in \mathbb{R} \implies \tilde{\nu} = \frac{1}{\lambda} = \frac{n}{\ell} = n \cdot \Sigma_t$$

- We can examine the modes of the spatial error by using an inverse Fourier transform. This will give us an idea of how the spatial frequencies of the error. We need to decide on an error wavelength, which gives us a linear error frequency. Higher n means higher error frequency, with $n = 0$ being infinite wavelength, completely non-coupled error.
- If we plug this back into our previous equation and do a large amount of manipulation, we get a fairly simple relationship between the integrated error in one step to the integrated error in the previous step.
- This lambda function is maximized when $n = 0$. The lowest frequency error converges the slowest, and at a rate proportional to Σ_s/Σ_t.

Fourier Analysis

To perform an inverse Fourier transform, we need to choose a measure of spatial variation, an error "wavelength."

$$\lambda = \frac{\ell}{n}, \quad \forall n \in \mathbb{R} \implies \tilde{\nu} = \frac{1}{\lambda} = \frac{n}{\ell} = n \cdot \Sigma_t$$

Perform an inverse Fourier transform,

$$e^k(x,\mu) = \int_{-\infty}^{\infty} \hat{e}^k(n,\mu) e^{i\Sigma_t n x} dn \,.$$

Fourier Analysis
To perform an inverse Fourier transform, we need to choose a measure of spatial variation, an error "wavelength."
$$\lambda = \frac{\ell}{n}, \quad \forall n \in \mathbb{R} \implies \tilde{\nu} = \frac{1}{\lambda} = \frac{n}{\ell} = n \cdot \Sigma_t$$
Perform an inverse Fourier transform,
$$e^k(x,\mu) = \int_{-\infty}^{\infty} \hat{e}^k(n,\mu) e^{i\Sigma_t n x} dn \,.$$

- We can examine the modes of the spatial error by using an inverse Fourier transform. This will give us an idea of how the spatial frequencies of the error. We need to decide on an error wavelength, which gives us a linear error frequency. Higher n means higher error frequency, with $n = 0$ being infinite wavelength, completely non-coupled error.
- If we plug this back into our previous equation and do a large amount of manipulation, we get a fairly simple relationship between the integrated error in one step to the integrated error in the previous step.
- This lambda function is maximized when $n = 0$. The lowest frequency error converges the slowest, and at a rate proportional to Σ_s/Σ_t.

Fourier Analysis

To perform an inverse Fourier transform, we need to choose a measure of spatial variation, an error "wavelength."

$$\lambda = \frac{\ell}{n}, \quad \forall n \in \mathbb{R} \implies \tilde{\nu} = \frac{1}{\lambda} = \frac{n}{\ell} = n \cdot \Sigma_t$$

Perform an inverse Fourier transform,

$$e^k(x,\mu) = \int_{-\infty}^{\infty} \hat{e}^k(n,\mu) e^{i\Sigma_t n x} dn \,.$$

After plugging into our equation for error and some rearranging,

$$\int_{-1}^{1} \hat{e}^{k+1}(n,\mu) d\mu = \Lambda(n) \int_{-1}^{1} \hat{e}^k(n,\mu') d\mu' \,,$$

Where

$$\Lambda(n) = \frac{\Sigma_s}{\Sigma_t} \cdot \frac{\tan^{-1}(n)}{n} \,.$$

- We can examine the modes of the spatial error by using an inverse Fourier transform. This will give us an idea of how the spatial frequencies of the error. We need to decide on an error wavelength, which gives us a linear error frequency. Higher n means higher error frequency, with $n = 0$ being infinite wavelength, completely non-coupled error.
- If we plug this back into our previous equation and do a large amount of manipulation, we get a fairly simple relationship between the integrated error in one step to the integrated error in the previous step.
- This lambda function is maximized when $n = 0$. The lowest frequency error converges the slowest, and at a rate proportional to Σ_s/Σ_t.

Problem

Problem Statement

In the presence of up-scattering and a substantial scattering cross-section, Gauss-Seidel source-iteration can converge arbitrarily slow because the error in diffuse, persistent modes after each iteration reduces by a factor of Σ_s/Σ_t.

Problem
Problem Statement
In the presence of up-scattering and a substantial scattering cross-section, Gauss-Seidel source-iteration can converge arbitrarily slow because the error in diffuse, persistent modes after each iteration reduces by a factor of Σ_s/Σ_t.

This motivates the development of acceleration schemes to speed up this convergence. This is especially applicable to shielding problems where scattering is dominant

The multigroup S_N equations

Apply the following discretizations:

- Apply a Petrov-Galerkin scheme in energy (multigroup method), splitting into G coupled equations.

The multigroup S_N equations
Apply the following discretizations:
- Apply a Petrov-Galerkin scheme in energy (multigroup method), splitting into G coupled equations.

- Multigroup method splits the equations into G coupled equations
- Collocation scheme in angle uses points for a quadrature rule for integrating angular flux to get flux moments
- Expand in Legendre polynomials, use polynomial addition theorem,

The multigroup S_N equations

Apply the following discretizations:

- Apply a Petrov-Galerkin scheme in energy (multigroup method), splitting into G coupled equations.
- Apply a collocation scheme in angle, solving at angles $\hat{\Omega}_a$.

The multigroup S_N equations
Apply the following discretizations:
- Apply a Petrov-Galerkin scheme in energy (multigroup method), splitting into G coupled equations.
- Apply a collocation scheme in angle, solving at angles $\hat{\Omega}_a$.

- Multigroup method splits the equations into G coupled equations
- Collocation scheme in angle uses points for a quadrature rule for integrating angular flux to get flux moments
- Expand in Legendre polynomials, use polynomial addition theorem,

The multigroup S_N equations

Apply the following discretizations:

- Apply a Petrov-Galerkin scheme in energy (multigroup method), splitting into G coupled equations.
- Apply a collocation scheme in angle, solving at angles $\hat{\Omega}_a$.
- Expanding scattering cross-section in Legendre Polynomials with a maximum degree N.

$$\Sigma_{s,g'g,\ell} = \int_{-1}^{1} \Sigma_{s,g'g}(\mathbf{r},\mu)P_\ell(\mu)d\mu, \quad \mu = \hat{\Omega}' \cdot \hat{\Omega}$$

$$\phi_{g,\ell,m} = \int_{4\pi} \phi_g(\mathbf{r},\hat{\Omega}')Y_{\ell,m}(\hat{\Omega}')d\hat{\Omega}'$$

The multigroup S_N equations
Apply the following discretizations:
- Apply a Petrov-Galerkin scheme in energy (multigroup method), splitting into G coupled equations.
- Apply a collocation scheme in angle, solving at angles $\hat{\Omega}_a$.
- Expanding scattering cross-section in Legendre Polynomials with a maximum degree N.
$$\Sigma_{s,g'g,\ell} = \int_{-1}^{1} \Sigma_{s,g'g}(\mathbf{r},\mu)P_\ell(\mu)d\mu, \quad \mu = \hat{\Omega}' \cdot \hat{\Omega}$$
$$\phi_{g,\ell,m} = \int_{4\pi} \phi_g(\mathbf{r},\hat{\Omega}')Y_{\ell,m}(\hat{\Omega}')d\hat{\Omega}'$$

- Multigroup method splits the equations into G coupled equations
- Collocation scheme in angle uses points for a quadrature rule for integrating angular flux to get flux moments
- Expand in Legendre polynomials, use polynomial addition theorem,

The multigroup S_N equations

Apply the following discretizations:

- Apply a Petrov-Galerkin scheme in energy (multigroup method), splitting into G coupled equations.
- Apply a collocation scheme in angle, solving at angles $\hat{\Omega}_a$.
- Expanding scattering cross-section in Legendre Polynomials with a maximum degree N.

Multigroup S_N equations

$$\left[\hat{\Omega}_a \cdot \nabla + \Sigma_{t,g}(\mathbf{r})\right] \psi_g(\mathbf{r}, \hat{\Omega}_a) = \sum_{g'=0}^{G} \sum_{\ell=0}^{N} \sum_{m=-\ell}^{\ell} \Sigma_{s,g'g,\ell} Y_{\ell,m}(\hat{\Omega}_a) \phi_{g',\ell,m}(\mathbf{r}) + Q_g(\mathbf{r}, \hat{\Omega}_a)$$

The multigroup S_N equations
Apply the following discretizations:
Apply a Petrov-Galerkin scheme in energy (multigroup method), splitting into G coupled equations.
Apply a collocation scheme in angle, solving at angles $\hat{\Omega}_a$.
Expanding scattering cross-section in Legendre Polynomials with a maximum degree N.
Multigroup S_N equations
$\left[\hat{\Omega}_a \cdot \nabla + \Sigma_{t,g}(\mathbf{r})\right] \psi_g(\mathbf{r}, \hat{\Omega}_a) = \sum_{g'=0}^{G} \sum_{\ell=0}^{N} \sum_{m=-\ell}^{\ell} \Sigma_{s,g'g,\ell} Y_{\ell,m}(\hat{\Omega}_a) \phi_{g',\ell,m}(\mathbf{r}) + Q_g(\mathbf{r}, \hat{\Omega}_a)$

- Multigroup method splits the equations into G coupled equations
- Collocation scheme in angle uses points for a quadrature rule for integrating angular flux to get flux moments
- Expand in Legendre polynomials, use polynomial addition theorem,

Iterative Solving Methods

Expressed in operator form, this is

$$\mathbf{L}_g\mathbf{\Psi}_g = \mathbf{M}\sum_{g'=0}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{Q}_g, \quad \mathbf{\Phi}_g = \mathbf{D}\mathbf{\Psi}_g \ .$$

Iterative Solving Methods
Expressed in operator form, this is
$$\mathbf{L}_g\mathbf{\Psi}_g = \mathbf{M}\sum_{g'=0}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{Q}_g, \quad \mathbf{\Phi}_g = \mathbf{D}\mathbf{\Psi}_g \ .$$

- M is the moment-to-discrete, D is the reverse
- Important to note that the G-th energy group is the lowest.

Iterative Solving Methods

Expressed in operator form, this is

$$\mathbf{L}_g\mathbf{\Psi}_g = \mathbf{M}\sum_{g'=0}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{Q}_g, \quad \mathbf{\Phi}_g = \mathbf{D}\mathbf{\Psi}_g \; .$$

Splitting the scattering source into down-scattering and up-scattering terms,

$$\mathbf{L}_g\mathbf{\Psi}_g = \mathbf{M}\sum_{g'=0}^{g}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{Q}_g \; ,$$

Iterative Solving Methods
Expressed in operator form, this is
$$\mathbf{L}_g\mathbf{\Psi}_g = \mathbf{M}\sum_{g'=0}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{Q}_g, \quad \mathbf{\Phi}_g = \mathbf{D}\mathbf{\Psi}_g \; .$$
Splitting the scattering source into down-scattering and up-scattering terms,
$$\mathbf{L}_g\mathbf{\Psi}_g = \mathbf{M}\sum_{g'=0}^{g}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{Q}_g \; ,$$

- M is the moment-to-discrete, D is the reverse
- Important to note that the G-th energy group is the lowest.

Iterative Solving Methods

Expressed in operator form, this is

$$\mathbf{L}_g\mathbf{\Psi}_g = \mathbf{M}\sum_{g'=0}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{Q}_g, \quad \mathbf{\Phi}_g = \mathbf{D}\mathbf{\Psi}_g \;.$$

Splitting the scattering source into down-scattering and up-scattering terms,

$$\mathbf{L}_g\mathbf{\Psi}_g = \mathbf{M}\sum_{g'=0}^{g}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{Q}_g \;,$$

And holding the source $\mathbf{Q}$ fixed leads to a Gauss-Seidel (scattering) source iteration,

$$\mathbf{L}_g\mathbf{\Psi}_g^{k+1} = \mathbf{M}\sum_{g'=0}^{g}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'}^{k+1} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'}^{k} + \mathbf{Q}_g \;.$$

Iterative Solving Methods
Expressed in operator form, this is
Splitting the scattering source into down-scattering and up-scattering terms,
And holding the source Q fixed leads to a Gauss-Seidel (scattering) source iteration,

- M is the moment-to-discrete, D is the reverse
- Important to note that the G-th energy group is the lowest.

Iterative Solving Methods

For a multiplying-medium problem, the fixed source $\mathbf{Q}$ is replaced with the fission source,

$$\mathbf{L}_g\boldsymbol{\Psi}_g = \mathbf{M}\sum_{g'=0}^{G}\left[\mathbf{S}_{g'g}\boldsymbol{\Phi}_{g'} + \frac{1}{k}\mathbf{F}_{g'}\boldsymbol{\Phi}_{g'}\right] .$$

Holding the scattering source fixed leads to power iteration (fission source iteration),

$$\mathbf{L}_g\boldsymbol{\Psi}_g^{k+1} = \mathbf{M}\sum_{g'=0}^{G}\left[\mathbf{S}_{g'g}\boldsymbol{\Phi}_{g'}^{0} + \frac{1}{k}\mathbf{F}_{g'}\boldsymbol{\Phi}_{g'}^{k}\right] .$$

In general, to converge both the fission and scattering sources, power iteration is paired with source iteration in an inner-outer convergence scheme.

Iterative Solving Methods

For a multiplying-medium problem, the fixed source $\mathbf{Q}$ is replaced with the fission source,

$$\mathbf{L}_g\boldsymbol{\Psi}_g = \mathbf{M}\sum_{g'=0}^{G}\left[\mathbf{S}_{g'g}\boldsymbol{\Phi}_{g'} + \frac{1}{k}\mathbf{F}_{g'}\boldsymbol{\Phi}_{g'}\right] .$$

Holding the scattering source fixed leads to power iteration (fission source iteration),

$$\mathbf{L}_g\boldsymbol{\Psi}_g^{k+1} = \mathbf{M}\sum_{g'=0}^{G}\left[\mathbf{S}_{g'g}\boldsymbol{\Phi}_{g'}^{0} + \frac{1}{k}\mathbf{F}_{g'}\boldsymbol{\Phi}_{g'}^{k}\right] .$$

In general, to converge both the fission and scattering sources, power iteration is paired with source iteration in an inner-outer convergence scheme.

Iterative Solve Error

Much of our analysis will require an examination of the error in each step of an iterative method. This is found by subtracting our method from the original equation.

Iterative Solve Error

Much of our analysis will require an examination of the error in each step of an iterative method. This is found by subtracting our method from the original equation.

Iterative Solve Error

Much of our analysis will require an examination of the error in each step of an iterative method. This is found by subtracting our method from the original equation.

$$\mathbf{L}_g\mathbf{\Psi}_g = \mathbf{M}\sum_{g'=0}^{g}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{Q}_g$$

$$\mathbf{L}_g\mathbf{\Psi}_g^{i+1} = \mathbf{M}\sum_{g'=0}^{g}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'}^{i+1} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'}^{i} + \mathbf{Q}_g$$

2019-08-29

Qualification Exam
└Transport Equation
└Iterative Solve Error

Iterative Solve Error
Much of our analysis will require an examination of the error in each step of an iterative method. This is found by subtracting our method from the original equation.

$$\mathbf{L}_g\mathbf{\Psi}_g = \mathbf{M}\sum_{g'=0}^{g}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'} + \mathbf{Q}_g$$

$$\mathbf{L}_g\mathbf{\Psi}_g^{i+1} = \mathbf{M}\sum_{g'=0}^{g}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'}^{i+1} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{S}_{g'g}\mathbf{\Phi}_{g'}^{i} + \mathbf{Q}_g$$

Iterative Solve Error

Much of our analysis will require an examination of the error in each step of an iterative method. This is found by subtracting our method from the original equation.

$$\mathbf{L}_g\epsilon_g^{i+1} = \mathbf{M}\sum_{g'=0}^{g}\mathbf{S}_{g'g}\varepsilon_{g'}^{i+1} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{S}_{g'g}\varepsilon_{g'}^{i}$$

$$\begin{aligned}\epsilon_g^{i+1} &= \mathbf{\Psi}_g - \mathbf{\Psi}_g^{i+1} \\ \varepsilon_g^{i+1} &= \mathbf{D}\epsilon_g^{i+1}\end{aligned}$$

Iterative Solve Error

Much of our analysis will require an examination of the error in each step of an iterative method. This is found by subtracting our method from the original equation.

$$\mathbf{L}_g\epsilon_g^{i+1} = \mathbf{M}\sum_{g'=0}^{g}\mathbf{S}_{g'g}\varepsilon_{g'}^{i+1} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{S}_{g'g}\varepsilon_{g'}^{i}$$

$$\begin{aligned}\epsilon_g^{i+1} &= \mathbf{\Psi}_g - \mathbf{\Psi}_g^{i+1} \\ \varepsilon_g^{i+1} &= \mathbf{D}\epsilon_g^{i+1}\end{aligned}$$

Acceleration Methods

Acceleration Methods

Nonlinear Diffusion Acceleration (NDA)

Start, with the single-group first-order transport equation [7, 5], and integrate over angle:

$$\nabla \cdot J_g + \left(\Sigma_{t,g} - \Sigma_s^{g\to g}\right)\phi_g = \sum_{g'\neq g} \Sigma_s^{g'\to g}\phi_{g'} + q_g, \quad J_g \equiv \int d\hat{\Omega}\hat{\Omega}\psi_g(\hat{\Omega}) \;.$$

As a closure to this problem, it is common to define current using *Fick's law*,

$$J_g = -D\nabla\phi_g \;.$$

Construct an additive correction to the current using information from an angular solve:

$$\begin{aligned} J_g &= -D\nabla\phi_g + J_g^{\text{ang}} - J_g^{\text{ang}} \\ &= -D\nabla\phi_g + \int_{4\pi} d\hat{\Omega}\hat{\Omega}\psi_g + D\nabla\phi_g \end{aligned}$$

Nonlinear Diffusion Acceleration (NDA)
Start, with the single-group first-order transport equation [7, 5], and integrate over angle:
$$\nabla \cdot J_g + \left(\Sigma_{t,g} - \Sigma_s^{g\to g}\right)\phi_g = \sum_{g'\neq g} \Sigma_s^{g'\to g}\phi_{g'} + q_g, \quad J_g \equiv \int d\hat{\Omega}\hat{\Omega}\psi_g(\hat{\Omega}) \;.$$
As a closure to this problem, it is common to define current using *Fick's law*,
$$J_g = -D\nabla\phi_g \;.$$
Construct an additive correction to the current using information from an angular solve:
$$\begin{aligned} J_g &= -D\nabla\phi_g + J_g^{\text{ang}} - J_g^{\text{ang}} \\ &= -D\nabla\phi_g + \int_{4\pi} d\hat{\Omega}\hat{\Omega}\psi_g + D\nabla\phi_g \end{aligned}$$

- Uses a lower order diffusion solve to accelerate a higher order solve.
- Start with the same single-group first-order transport equation, multiply by and integrate over angle, giving the "neutron continuity equation."
- We need closure for this problem, so often we use Fick's law, we will introduce a correction onto Fick's Law based on a higher order solve.
- We will introduce an additive correction based on our two definitions of the current.

Nonlinear Diffusion Acceleration (NDA)

Fold the additive correction into a *drift-diffusion vector*:

$$\begin{aligned}
J_g &= -D\nabla\phi_g + \int_{4\pi} d\hat{\Omega}\hat{\Omega}\psi_g + D\nabla\phi_g \\
&= -D\nabla\phi_g + \left[\frac{\int_{4\pi} d\hat{\Omega}\hat{\Omega}\psi_g + D\nabla\phi_g}{\phi_g}\right]\phi_g \\
&= -D\nabla\phi_g + \hat{D}_g\phi_g \;.
\end{aligned}$$

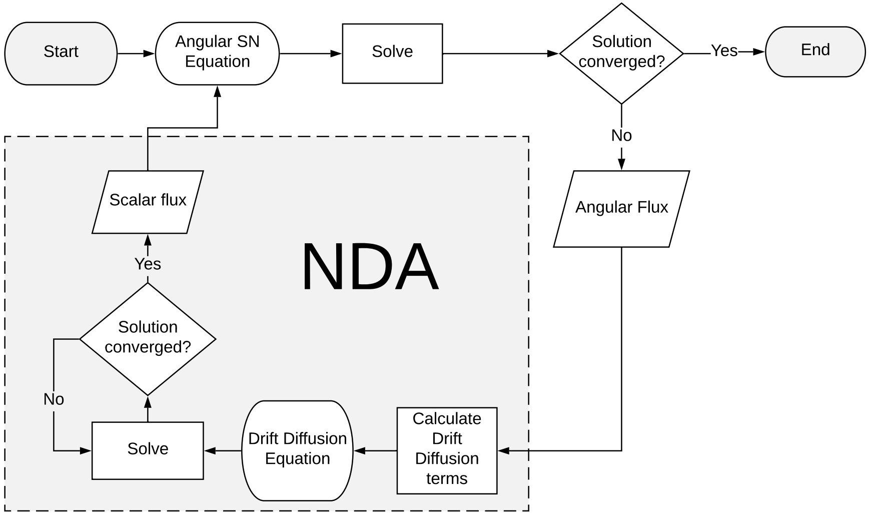
Plugging this into our integrated transport equation gives the low-order non-linear diffusion acceleration equation (LONDA),

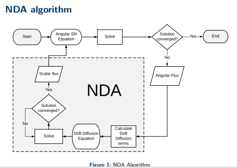
$$\nabla \cdot \left[-D\nabla + \hat{D}_g\right]\phi_g + \left(\Sigma_{t,g} - \Sigma_s^{g\to g}\right)\phi_g = \sum_{g' \neq g} \Sigma_s^{g'\to g}\phi_{g'} + q_g$$

Nonlinear Diffusion Acceleration (NDA)
Fold the additive correction into a drift-diffusion vector:
Plugging this into our integrated transport equation gives the low-order non-linear diffusion acceleration equation (LONDA).

- We combine these corrections into a drift diffusion vector.
- This gives us the LONDA equation, which is just the same integrated transport equation with a corrected current term.
- Presumably, the “higher order” angular solve will have better current information, so we can use it to calculate the drift diffusion vector.

NDA algorithm





- NDA algorithm showing inner low order loop, and outer high order loop.
- In general, outer loop updates both scattering and fission source, checking for k convergence. Inner loop updates fission source, also checking k convergence.

Two-grid acceleration

To mitigate this issue Adams and Morel [1] developed the two-grid method which rests on two assumptions:

- The persistent error modes can be accurately determined by a coarse-grid approximation.

Two-grid acceleration

To mitigate this issue Adams and Morel [1] developed the two-grid method which rests on two assumptions:

- The persistent error modes can be accurately determined by a coarse-grid approximation.

- This should speed up the solve by giving an addition reduction in those diffuse persistent error modes.

Two-grid acceleration

To mitigate this issue Adams and Morel [1] developed the two-grid method which rests on two assumptions:

- The persistent error modes can be accurately determined by a coarse-grid approximation.
- Solving this coarse-grid approximation is more economical than solving the actual equation.

Two-grid acceleration

To mitigate this issue Adams and Morel [1] developed the two-grid method which rests on two assumptions:

- The persistent error modes can be accurately determined by a coarse-grid approximation.
- Solving this coarse-grid approximation is more economical than solving the actual equation.

- This should speed up the solve by giving an addition reduction in those diffuse persistent error modes.

Two-grid acceleration

To mitigate this issue Adams and Morel [1] developed the two-grid method which rests on two assumptions:

- The persistent error modes can be accurately determined by a coarse-grid approximation.
- Solving this coarse-grid approximation is more economical than solving the actual equation.

Two-grid Acceleration

Solve for the error using a coarse-grid approximation, and use it as a correction to our solution in each step.

Two-grid acceleration

To mitigate this issue Adams and Morel [1] developed the two-grid method which rests on two assumptions:

- The persistent error modes can be accurately determined by a coarse-grid approximation.
- Solving this coarse-grid approximation is more economical than solving the actual equation.

Two-grid Acceleration

Solve for the error using a coarse-grid approximation, and use it as a correction to our solution in each step.

- This should speed up the solve by giving an addition reduction in those diffuse persistent error modes.

Two-grid acceleration

Step 1: Solve the angular S_N source-iteration equation,

$$\mathbf{L}_g \mathbf{\Psi}_g^{i+\frac{1}{2}} = \mathbf{M} \sum_{g'=0}^{g} \mathbf{S}_{g'g} \mathbf{\Phi}_{g'}^{i+\frac{1}{2}} + \mathbf{M} \sum_{g'=g+1}^{G} \mathbf{S}_{g'g} \mathbf{\Phi}_{g'}^{k} + \mathbf{Q}_g \; .$$

Step 2: Calculate the isotropic component of the residual,

$$\mathbf{R}_{g,0}^{i+\frac{1}{2}} = \sum_{g'=g+1}^{G} \mathbf{S}_{g'g} \left(\mathbf{\Phi}_{g'}^{i+\frac{1}{2}} - \mathbf{\Phi}_{g'}^{i} \right)$$

Step 3: Calculate the error.

$$\mathbf{L}_g \epsilon_g^{i+\frac{1}{2}} = \mathbf{M} \sum_{g'=0}^{G} \mathbf{S}_{g'g} \varepsilon_{g'}^{i+\frac{1}{2}} + \mathbf{R}_g^{i+\frac{1}{2}}$$

2019-08-29

Qualification Exam
└─Acceleration Methods
└─Two-grid acceleration

Two-grid acceleration
Step 1: Solve the angular S_N source-iteration equation,
Step 2: Calculate the isotropic component of the residual,
Step 3: Calculate the error.

Two-grid acceleration

Step 3a: Calculate error using integrated diffusion approximation.

$$\left(-\nabla \cdot \langle D_g \rangle \nabla + \Sigma_g\right) \tilde{\varepsilon}_g^{i+\frac{1}{2}} = \sum_{g'=0}^{G} \Sigma_{s,g'g,0} \tilde{\varepsilon}_{g'}^{i+\frac{1}{2}} + \mathbf{R}_{g,0}^{i+\frac{1}{2}}$$

Step 4: Correct the flux

$$\mathbf{\Psi}_g^{i+1} = \mathbf{\Psi}_g^{i+\frac{1}{2}} + \mathbf{M}\tilde{\varepsilon}_g^{i+\frac{1}{2}}$$

This will accelerate our solution only if it removes more error with less work than our original method.

Two-grid acceleration

Step 3a: Calculate error using integrated diffusion approximation.

$$\left(-\nabla \cdot \langle D_g \rangle \nabla + \Sigma_g\right) \tilde{\varepsilon}_g^{i+\frac{1}{2}} = \sum_{g'=0}^{G} \Sigma_{s,g'g,0} \tilde{\varepsilon}_{g'}^{i+\frac{1}{2}} + \mathbf{R}_{g,0}^{i+\frac{1}{2}}$$

Step 4: Correct the flux

$$\mathbf{\Psi}_g^{i+1} = \mathbf{\Psi}_g^{i+\frac{1}{2}} + \mathbf{M}\tilde{\varepsilon}_g^{i+\frac{1}{2}}$$

This will accelerate our solution only if it removes more error with less work than our original method.

Two-grid Acceleration

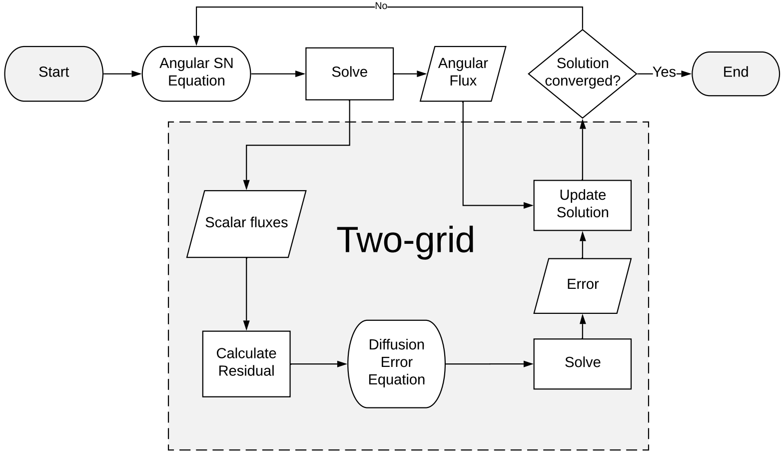


Figure 2: Two-grid flowchart.

2019-08-29

Qualification Exam
└Acceleration Methods
└Two-grid Acceleration

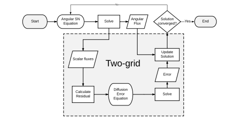


Figure 2: Two-grid flowchart.

Acceleration Methods

Two acceleration methods:

- **Nonlinear Diffusion Acceleration**: improves the convergence of the multi-group Gauss-Seidel iteration, but suffers from convergence issues with a large amount of upscattering.
- **Two-grid Acceleration**: improves the convergence of multi-group problems with a large amount of upscattering.

Novel Combination

Use two-grid acceleration to improve the convergence rate of the low-order portion of the NDA solve.

Acceleration Methods

Two acceleration methods:

- **Nonlinear Diffusion Acceleration**: improves the convergence of the multi-group Gauss-Seidel iteration, but suffers from convergence issues with a large amount of upscattering.
- **Two-grid Acceleration**: improves the convergence of multi-group problems with a large amount of upscattering.

Novel Combination

Use two-grid acceleration to improve the convergence rate of the low-order portion of the NDA solve.

Combining Acceleration Methods

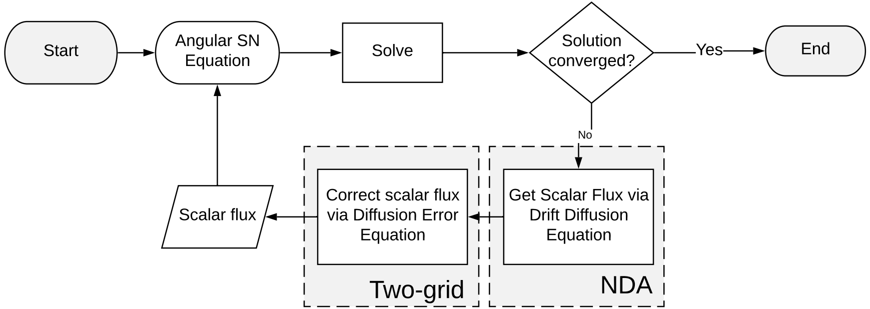
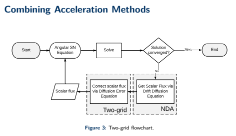


Figure 3: Two-grid flowchart.

2019-08-29

Qualification Exam
└Acceleration Methods
└Combining Acceleration Methods



Analyzing Acceleration Schemes

Analyzing Acceleration Schemes

State of the art

For an acceleration scheme to be worthwhile, it must remove more error from the solution for less work. Defining *work* is challenging. In general, we use inversions of the transport matrix (or *sweeps*) as a unit of "work."

State of the art

For an acceleration scheme to be worthwhile, it must remove more error from the solution for less work. Defining *work* is challenging. In general, we use inversions of the transport matrix (or *sweeps*) as a unit of "work."

The problem with this is that it's unclear how much actual work is being done in each step. You could form an acceleration scheme that solves in a single outer iteration, but is doing so *actually* accelerating removing more error in less work, or just moving work around?

We can use Fourier analysis like before, but things get complicated when we move into multidimensional problems, and start combining accelerating schemes. We need more insight into the acceleration process.

State of the art

For an acceleration scheme to be worthwhile, it must remove more error from the solution for less work. Defining *work* is challenging. In general, we use inversions of the transport matrix (or *sweeps*) as a unit of "work."

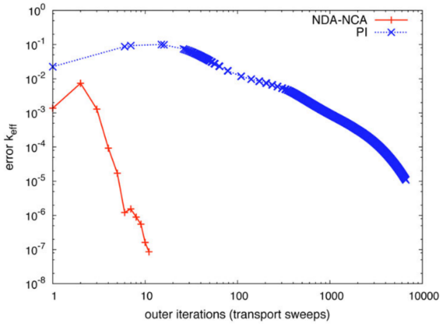
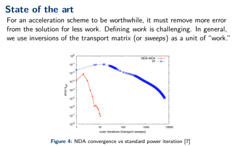


Figure 4: NDA convergence vs standard power iteration [7]



The problem with this is that it's unclear how much actual work is being done in each step. You could form an acceleration scheme that solves in a single outer iteration, but is doing so *actually* accelerating removing more error in less work, or just moving work around?

We can use Fourier analysis like before, but things get complicated when we move into multidimensional problems, and start combining accelerating schemes. We need more insight into the acceleration process.

Analysis Challenges

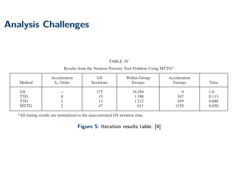
TABLE IV

Results from the Neutron Porosity Tool Problem Using MTTG*

Method	Acceleration S_N Order	GS Iterations	Within-Group Sweeps	Acceleration Sweeps	Time
GS	—	175	16 294	0	1.0
TTG	8	15	1 398	547	0.113
TTG	2	13	1 212	459	0.086
MTTG	2	47	611	1329	0.050

*All timing results are normalized to the unaccelerated GS iteration time.

Figure 5: Iteration results table. [4]



Here is an example of an iteration table from a paper analyzing the two-grid method. It shows both Gauss-Seidel iterations, within group sweeps and acceleration sweeps, but we don't have a clear idea of what parts of the problem are doing all the work. We don't know where the error is being removed, and if this method is doing it more economically or just shifting it around. The *time* is a good indication, but not ideal. Is it proper to use clock time? CPU Time? How do we know that it's not faster because of better computer science. We not only need insight into the inner workings of acceleration schemes, but we need to dis-aggregate the computer science from the mathematics.

Analysis Challenges

A few challenges when analyzing the effectiveness of acceleration schemes include:

- Our definition of work is based on assumptions about algorithm efficiency.

Analysis Challenges

A few challenges when analyzing the effectiveness of acceleration schemes include:

- Our definition of work is based on assumptions about algorithm efficiency.

Analysis Challenges

A few challenges when analyzing the effectiveness of acceleration schemes include:

- Our definition of work is based on assumptions about algorithm efficiency.
- Combining or using complex acceleration schemes may invalidate these assumptions.

Analysis Challenges

A few challenges when analyzing the effectiveness of acceleration schemes include:

- Our definition of work is based on assumptions about algorithm efficiency.
- Combining or using complex acceleration schemes may invalidate these assumptions.

Analysis Challenges

A few challenges when analyzing the effectiveness of acceleration schemes include:

- Our definition of work is based on assumptions about algorithm efficiency.
- Combining or using complex acceleration schemes may invalidate these assumptions.
- Implementing new schemes can be complicated, making it difficult to dis-aggregate implementation from theory.

Analysis Challenges

A few challenges when analyzing the effectiveness of acceleration schemes include:

- Our definition of work is based on assumptions about algorithm efficiency.
- Combining or using complex acceleration schemes may invalidate these assumptions.
- Implementing new schemes can be complicated, making it difficult to dis-aggregate implementation from theory.

Analysis Challenges

A few challenges when analyzing the effectiveness of acceleration schemes include:

- Our definition of work is based on assumptions about algorithm efficiency.
- Combining or using complex acceleration schemes may invalidate these assumptions.
- Implementing new schemes can be complicated, making it difficult to dis-aggregate implementation from theory.
- It can be difficult to reproduce results when accelerated codes are not portable.

Analysis Challenges

A few challenges when analyzing the effectiveness of acceleration schemes include:

- Our definition of work is based on assumptions about algorithm efficiency.
- Combining or using complex acceleration schemes may invalidate these assumptions.
- Implementing new schemes can be complicated, making it difficult to dis-aggregate implementation from theory.
- It can be difficult to reproduce results when accelerated codes are not portable.

BART

BART

Design Goals for BART

BART is a new code in development designed with features to overcome some of the challenges of analyzing acceleration methods. Four major design goals include:

1. Leverage polymorphism to make implementing new methods easier and limit the code needed to do so.

Design Goals for BART

BART is a new code in development designed with features to overcome some of the challenges of analyzing acceleration methods. Four major design goals include:

1. Leverage polymorphism to make implementing new methods easier and limit the code needed to do so.

Design Goals for BART

BART is a new code in development designed with features to overcome some of the challenges of analyzing acceleration methods. Four major design goals include:

1. Leverage polymorphism to make implementing new methods easier and limit the code needed to do so.
2. Include tools to analyze the effectiveness of acceleration schemes.

Design Goals for BART

BART is a new code in development designed with features to overcome some of the challenges of analyzing acceleration methods. Four major design goals include:

1. Leverage polymorphism to make implementing new methods easier and limit the code needed to do so.
2. Include tools to analyze the effectiveness of acceleration schemes.

Design Goals for BART

BART is a new code in development designed with features to overcome some of the challenges of analyzing acceleration methods. Four major design goals include:

1. Leverage polymorphism to make implementing new methods easier and limit the code needed to do so.
2. Include tools to analyze the effectiveness of acceleration schemes.
3. Provide a framework for users to experiment with novel combinations of and modifications to existing acceleration schemes.

Design Goals for BART

BART is a new code in development designed with features to overcome some of the challenges of analyzing acceleration methods. Four major design goals include:

1. Leverage polymorphism to make implementing new methods easier and limit the code needed to do so.
2. Include tools to analyze the effectiveness of acceleration schemes.
3. Provide a framework for users to experiment with novel combinations of and modifications to existing acceleration schemes.

Design Goals for BART

BART is a new code in development designed with features to overcome some of the challenges of analyzing acceleration methods. Four major design goals include:

1. Leverage polymorphism to make implementing new methods easier and limit the code needed to do so.
2. Include tools to analyze the effectiveness of acceleration schemes.
3. Provide a framework for users to experiment with novel combinations of and modifications to existing acceleration schemes.
4. Utilize modern coding and tests practices to make it easier for users to develop and have confidence in their solutions.

Design Goals for BART

BART is a new code in development designed with features to overcome some of the challenges of analyzing acceleration methods. Four major design goals include:

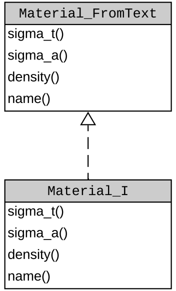
1. Leverage polymorphism to make implementing new methods easier and limit the code needed to do so.
2. Include tools to analyze the effectiveness of acceleration schemes.
3. Provide a framework for users to experiment with novel combinations of and modifications to existing acceleration schemes.
4. Utilize modern coding and tests practices to make it easier for users to develop and have confidence in their solutions.

Polymorphism

Material_I
sigma_t()
sigma_a()
density()
name()

Polymorphism
Material_I
sigma_t()
sigma_a()
density()
name()

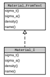
Polymorphism



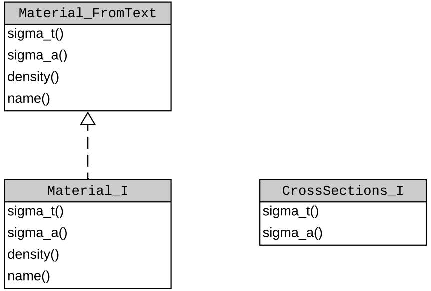
Qualification Exam
└BART
└Polymorphism

2019-08-29

Polymorphism

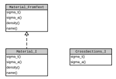


Polymorphism

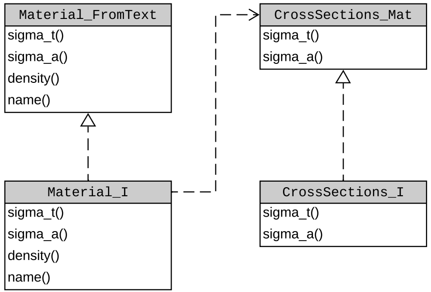


2019-08-29

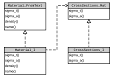
Qualification Exam
└BART
└Polymorphism


Polymorphism

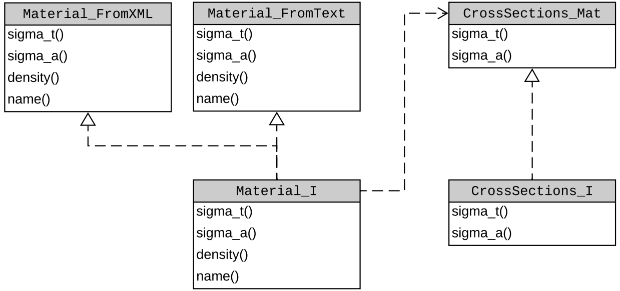
Polymorphism



Polymorphism

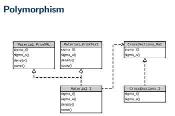


Polymorphism

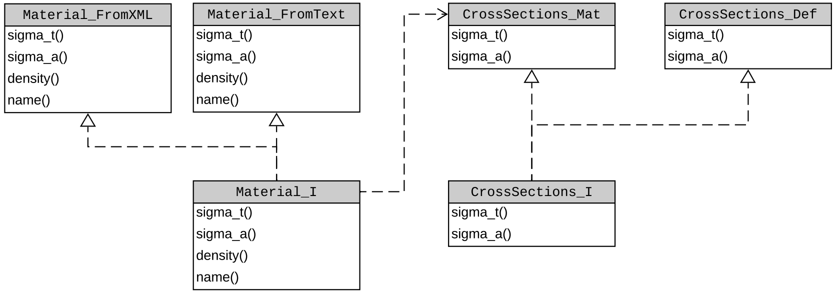


2019-08-29

Qualification Exam
└BART
└Polymorphism

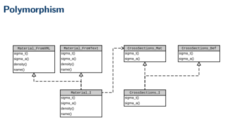


Polymorphism



2019-08-29

Qualification Exam
└BART
└Polymorphism

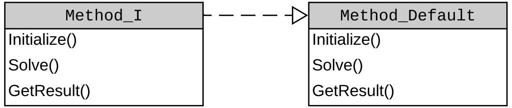


Polymorphism

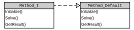
Method_I
Initialize()
Solve()
GetResult()

Polymorphism

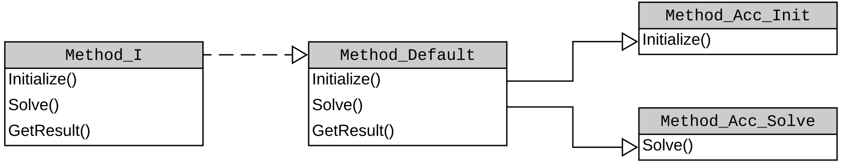
Polymorphism



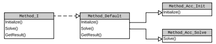
Polymorphism



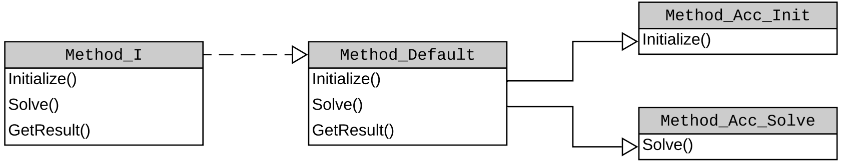
Polymorphism



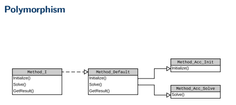
Polymorphism



Polymorphism



2019-08-29
Qualification Exam
└BART
└Polymorphism



Polymorphism Benefits

The use of polymorphism in BART

- Minimizes code changes needed to implement new methods, making it faster and easier.

Polymorphism Benefits

The use of polymorphism in BART

- Minimizes code changes needed to implement new methods, making it faster and easier.

Polymorphism Benefits

The use of polymorphism in BART

- Minimizes code changes needed to implement new methods, making it faster and easier.
- Enables a true comparison of the accelerated solve to a control solve.

Polymorphism Benefits

The use of polymorphism in BART

- Minimizes code changes needed to implement new methods, making it faster and easier.
- Enables a true comparison of the accelerated solve to a control solve.

Polymorphism Benefits

The use of polymorphism in BART

- Minimizes code changes needed to implement new methods, making it faster and easier.
- Enables a true comparison of the accelerated solve to a control solve.
- Makes the modifications *portable*.

Polymorphism Benefits

The use of polymorphism in BART

- Minimizes code changes needed to implement new methods, making it faster and easier.
- Enables a true comparison of the accelerated solve to a control solve.
- Makes the modifications *portable*.

Polymorphism Benefits

The use of polymorphism in BART

- Minimizes code changes needed to implement new methods, making it faster and easier.
- Enables a true comparison of the accelerated solve to a control solve.
- Makes the modifications *portable*.
- Enables us to compare the implementation of the method to dis-aggregate the computer science from the method itself.

Polymorphism Benefits

The use of polymorphism in BART

- Minimizes code changes needed to implement new methods, making it faster and easier.
- Enables a true comparison of the accelerated solve to a control solve.
- Makes the modifications *portable*.
- Enables us to compare the implementation of the method to dis-aggregate the computer science from the method itself.

Instrumentation

Goal 2

Include tools to analyze the effectiveness of acceleration schemes.

Instrumentation
Goal 2
Include tools to analyze the effectiveness of acceleration schemes.

Instrumentation

Goal 2

Include tools to analyze the effectiveness of acceleration schemes.

BART will include the ability to *instrument* different parts of the solve, to gather enough information to assess the effectiveness of acceleration schemes. Ideas for instrumentation are:

Instrumentation

Goal 2
Include tools to analyze the effectiveness of acceleration schemes.

BART will include the ability to *instrument* different parts of the solve, to gather enough information to assess the effectiveness of acceleration schemes. Ideas for instrumentation are:

Instrumentation

Goal 2

Include tools to analyze the effectiveness of acceleration schemes.

BART will include the ability to *instrument* different parts of the solve, to gather enough information to assess the effectiveness of acceleration schemes. Ideas for instrumentation are:

- Storage of solve parameters (eigenvalues, fluxes) at specified steps interval.

Instrumentation

Goal 2
Include tools to analyze the effectiveness of acceleration schemes.

BART will include the ability to *instrument* different parts of the solve, to gather enough information to assess the effectiveness of acceleration schemes. Ideas for instrumentation are:

- Storage of solve parameters (eigenvalues, fluxes) at specified steps interval.

Instrumentation

Goal 2

Include tools to analyze the effectiveness of acceleration schemes.

BART will include the ability to *instrument* different parts of the solve, to gather enough information to assess the effectiveness of acceleration schemes. Ideas for instrumentation are:

- Storage of solve parameters (eigenvalues, fluxes) at specified steps interval.
- Storage of hierarchy of iterations.

Instrumentation

Goal 2
Include tools to analyze the effectiveness of acceleration schemes.

BART will include the ability to *instrument* different parts of the solve, to gather enough information to assess the effectiveness of acceleration schemes. Ideas for instrumentation are:

- Storage of solve parameters (eigenvalues, fluxes) at specified steps interval.
- Storage of hierarchy of iterations.

Instrumentation

Goal 2

Include tools to analyze the effectiveness of acceleration schemes.

BART will include the ability to *instrument* different parts of the solve, to gather enough information to assess the effectiveness of acceleration schemes. Ideas for instrumentation are:

- Storage of solve parameters (eigenvalues, fluxes) at specified steps interval.
- Storage of hierarchy of iterations.
- Calculation and storage of error or residual at specified step interval.

Instrumentation

Goal 2
Include tools to analyze the effectiveness of acceleration schemes.

BART will include the ability to *instrument* different parts of the solve, to gather enough information to assess the effectiveness of acceleration schemes. Ideas for instrumentation are:

- Storage of solve parameters (eigenvalues, fluxes) at specified steps interval.
- Storage of hierarchy of iterations.
- Calculation and storage of error or residual at specified step interval.

Instrumentation

Goal 2

Include tools to analyze the effectiveness of acceleration schemes.

BART will include the ability to *instrument* different parts of the solve, to gather enough information to assess the effectiveness of acceleration schemes. Ideas for instrumentation are:

- Storage of solve parameters (eigenvalues, fluxes) at specified steps interval.
- Storage of hierarchy of iterations.
- Calculation and storage of error or residual at specified step interval.
- Analysis of error modes via fast Fourier transform.

Instrumentation

Goal 2
Include tools to analyze the effectiveness of acceleration schemes.

BART will include the ability to *instrument* different parts of the solve, to gather enough information to assess the effectiveness of acceleration schemes. Ideas for instrumentation are:

- Storage of solve parameters (eigenvalues, fluxes) at specified steps interval.
- Storage of hierarchy of iterations.
- Calculation and storage of error or residual at specified step interval.
- Analysis of error modes via fast Fourier transform.

Framework for experimentation

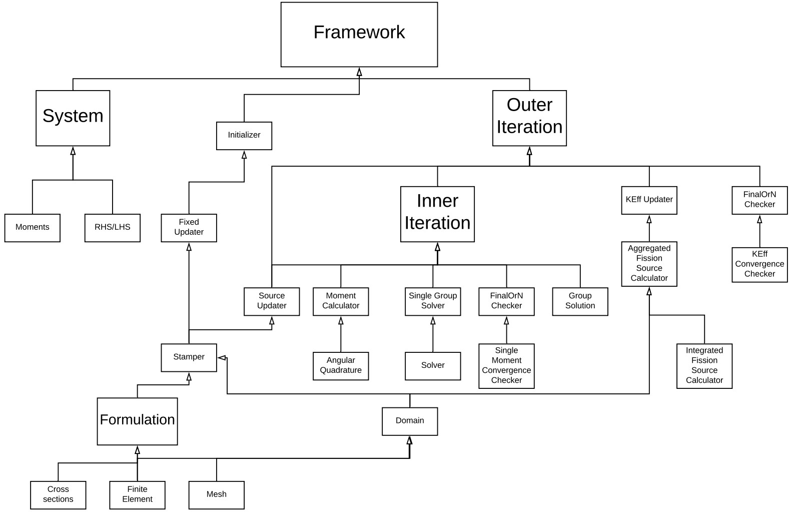
Goal 3

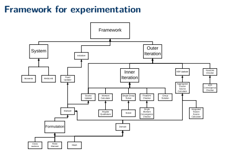
Provide a framework for users to experiment with novel combinations of and modifications to existing acceleration schemes.

Framework for experimentation

Goal 3
Provide a framework for users to experiment with novel combinations of and modifications to existing acceleration schemes.

Framework for experimentation





Modern Coding Practices

Goal 4

Utilize modern coding and tests practices to make it easier for users to develop and have confidence in their solutions.

- Build using the methods of modern `C++-14`.
- Cross-sections have the capability of being stored in Google Protocol Buffers.
- BART uses the `googletest` and `googlemock` libraries for unit testing. Unit testing coverage via `codecov`
- All dependencies for BART are built in an available Docker container.
- Continuous integration via `travis.ci`.

Modern Coding Practices

Goal 4
Utilize modern coding and tests practices to make it easier for users to develop and have confidence in their solutions.

- Build using the methods of modern C++-14.
- Cross-sections have the capability of being stored in Google Protocol Buffers.
- BART uses the googletest and googlemock libraries for unit testing. Unit testing coverage via codecov
- All dependencies for BART are built in an available Docker container.
- Continuous integration via travis.ci.

Plan and Future Work

BART Implementation Plan

Formulations:

- Interface for second-order transport equation formulations using continuous finite element methods.
- Implementation of Diffusion.
- Implementation of Self-Adjoint angular flux equation.

BART Implementation Plan
Formulations:
- Interface for second-order transport equation formulations using continuous finite element methods.
- Implementation of Diffusion.
- Implementation of Self-Adjoint angular flux equation.

BART Implementation Plan

Formulations:

- Interface for second-order transport equation formulations using continuous finite element methods.
- Implementation of Diffusion.
- Implementation of Self-Adjoint angular flux equation.

Acceleration methods:

- Nonlinear diffusion acceleration.
- Two-grid acceleration.
- Nonlinear diffusion acceleration with two-grid acceleration.

BART Implementation Plan
Formulations:
- Interface for second-order transport equation formulations using continuous finite element methods.
- Implementation of Diffusion.
- Implementation of Self-Adjoint angular flux equation.
Acceleration methods:
- Nonlinear diffusion acceleration.
- Two-grid acceleration.
- Nonlinear diffusion acceleration with two-grid acceleration.

BART Implementation Plan

Formulations:

- Interface for second-order transport equation formulations using continuous finite element methods.
- Implementation of Diffusion.
- Implementation of Self-Adjoint angular flux equation.

Acceleration methods:

- Nonlinear diffusion acceleration.
- Two-grid acceleration.
- Nonlinear diffusion acceleration with two-grid acceleration.

Instrumentation:

- In-step Fourier-transform.
- Iteration hierarchy counting.
- Automated runs based on mesh refinement.

BART Implementation Plan
Formulations:
- Interface for second-order transport equation formulations using continuous finite element methods.
- Implementation of Diffusion.
- Implementation of Self-Adjoint angular flux equation.
Acceleration methods:
- Nonlinear diffusion acceleration.
- Two-grid acceleration.
- Nonlinear diffusion acceleration with two-grid acceleration.
Instrumentation:
- In-step Fourier-transform.
- Iteration hierarchy counting.
- Automated runs based on mesh refinement.

Project Deliverable

1. A new C++ code that solves the continuous finite-element discretization of second-order transport equations.

Project Deliverable

1. A new C++ code that solves the continuous finite-element discretization of second-order transport equations.

Project Deliverable

1. A new C++ code that solves the continuous finite-element discretization of second-order transport equations.
2. Implementation of a novel acceleration method that utilizes NDA and TG, NDA-TG.

Project Deliverable
1. A new C++ code that solves the continuous finite-element discretization of second-order transport equations.
2. Implementation of a novel acceleration method that utilizes NDA and TG, NDA-TG.

Project Deliverable

1. A new C++ code that solves the continuous finite-element discretization of second-order transport equations.
2. Implementation of a novel acceleration method that utilizes NDA and TG, NDA-TG.
3. An in-depth analysis of the three acceleration methods (NDA, TG, NDA-TG) using novel instrumentation implemented in the code.

Project Deliverable

1. A new C++ code that solves the continuous finite-element discretization of second-order transport equations.
2. Implementation of a novel acceleration method that utilizes NDA and TG, NDA-TG.
3. An in-depth analysis of the three acceleration methods (NDA, TG, NDA-TG) using novel instrumentation implemented in the code.

Future Work

- Addition of Discontinuous-Galerkin Finite Element Formulations support to BART.

Future Work

- Addition of Discontinuous-Galerkin Finite Element Formulations support to BART.

Future Work

- Addition of Discontinuous-Galerkin Finite Element Formulations support to BART.
- More acceleration methods implemented to test different combinations.

2019-08-29
Qualification Exam
└Plan and Future Work
└Future Work

Future Work

- Addition of Discontinuous-Galerkin Finite Element Formulations support to BART.
- More acceleration methods implemented to test different combinations.

Future Work

- Addition of Discontinuous-Galerkin Finite Element Formulations support to BART.
- More acceleration methods implemented to test different combinations.
- Better or more complex *in situ* analysis of acceleration efficiency.

Future Work

- Addition of Discontinuous-Galerkin Finite Element Formulations support to BART.
- More acceleration methods implemented to test different combinations.
- Better or more complex *in situ* analysis of acceleration efficiency.

Future Work

- Addition of Discontinuous-Galerkin Finite Element Formulations support to BART.
- More acceleration methods implemented to test different combinations.
- Better or more complex *in situ* analysis of acceleration efficiency.
- Automated acceleration control (adaptive acceleration).

Future Work

- Addition of Discontinuous-Galerkin Finite Element Formulations support to BART.
- More acceleration methods implemented to test different combinations.
- Better or more complex *in situ* analysis of acceleration efficiency.
- Automated acceleration control (adaptive acceleration).

References

[1] B. T. Adams and Jim E. Morel.
A Two-Grid Acceleration Scheme for the Multigroup Sn Equations with Neutron Upscattering.
Nuclear Science and Engineering, 115(May):253–264, 1993.

[2] Marvin L. Adams and Edward W. Larsen.
Fast iterative methods for discrete-ordinates particle transport calculations.
Progress in Nuclear Energy, 2002.

[3] E. E. Lewis and W.F. Miller, Jr.
Computational Methods of Neutron Transport.
American Nuclear Society, 1993.

[4] Thomas M. Evans, Kevin T. Clarno, and Jim E. Morel.
A Transport Acceleration Scheme for Multigroup Discrete Ordinates with Upscattering.
Nuclear Science and Engineering, 165:292–304, 2010.

[5] Hans R Hammer, Jim E. Morel, and Yaqi Wang.
Nonlinear Diffusion Acceleration in Voids for the Weighted Least-Square Transport Equation.
In *Mathematics and Computation2*, 2017.

[6] J E Morel and J M Mcghee.
A Self-Adjoint Angular Flux Equation.
Nuclear Science and Engineering, 132:312–325, 1999.

[7] H. Park, D. A. Knoll, and C. K. Newman.
Nonlinear Acceleration of Transport Criticality Problems.
Nuclear Science and Engineering, 172(1):52–65, 2012.

Backup Slides

Energy discretization

Introduce a discretization of the energy domain $\mathbb{E}$ into G non-overlapping elements, such that

$$E_h = \{E_1, E_2, \dots, E_G\}\,, \quad \mathbb{E} = \bigcup_{g=1}^{G} E_g$$

Assume that the energy-dependent angular flux can be separated into a group angular flux and a energy function within each of these groups

$$\psi(\mathbf{r}, E, \hat{\Omega}) \approx \psi_g(\mathbf{r}, \hat{\Omega}) f_g(E), \quad E \in E_g$$

This gives us G coupled equations for each energy group, converting the integral scattering term into a summation,

$$\left[\hat{\Omega}\cdot\nabla + \Sigma_{t,g}(\mathbf{r})\right]\psi_g(\mathbf{r},\hat{\Omega}) = \sum_{g'=0}^{G} \Sigma_{s,g'\to g}(\mathbf{r},\hat{\Omega}'\to\hat{\Omega})\psi_{g'}(\mathbf{r},\hat{\Omega}') + Q_g(\mathbf{r},\hat{\Omega})\,.$$

- Say that the function f_g is zero inside element, and 0 outside, Petrov-Galerkin scheme.

Second-order forms of the Transport Equation

There are various second-order, self-adjoint forms of the transport equation.

- Even/Odd-parity equations (EP).
- Weighted least-squared formulation (WLS).
- Self-Adjoint angular flux (SAAF).

With advantages and disadvantages compared to the standard first-order forms. Advantages include:

- They can be solved on multidimensional finite element meshes using standard continuous finite element methods (CFEM).
- CFEM methods result in symmetric positive-definite (SPD) matrices.
- When using the P_N formulation, the flux moments are strongly coupled via $\hat{\Omega} \cdot \nabla$.

Second-order forms of the Transport Equation
There are various second-order, self-adjoint forms of the transport equation.
- Even/Odd-parity equations (EP).
- Weighted least-squared formulation (WLS).
- Self-Adjoint angular flux (SAAF).
With advantages and disadvantages compared to the standard first-order forms. Advantages include:
- They can be solved on multidimensional finite element meshes using standard continuous finite element methods (CFEM).
- CFEM methods result in symmetric positive-definite (SPD) matrices.
- When using the P_N formulation, the flux moments are strongly coupled via $\hat{\Omega} \cdot \nabla$.

- First-order forms of the TE form block lower-triangular that can be swept. But on many meshes, there are slightly re-entrant cells that will break this pattern.
- Solution methods for SPD matrices are better, CG vs. GMRES.

Second-order forms of the Transport Equation

Disadvantages include:

- CFEM methods result in a general sparse matrix, not a block lower-triangular.

Second-order forms of the Transport Equation
Disadvantages include:
- CFEM methods result in a general sparse matrix, not a block lower-triangular.

- Block lower-triangular would have allowed us to sweep.
- In pure scattering, there is a singularity in the scattering matrix for OP and SAAF in the spherical-harmonic basis. This is because it is diagonal and the first entry is $1/(\Sigma_t - \Sigma_{s0})$ which is $1/0$

Second-order forms of the Transport Equation

Disadvantages include:

- CFEM methods result in a general sparse matrix, not a block lower-triangular.
- In some forms (EP), reflective boundary conditions have fully implicit coupling between incoming and outgoing flux. SAAF avoids this only coupling incoming to outgoing.

Second-order forms of the Transport Equation

Disadvantages include:
- CFEM methods result in a general sparse matrix, not a block lower-triangular.
- In some forms (EP), reflective boundary conditions have fully implicit coupling between incoming and outgoing flux. SAAF avoids this only coupling incoming to outgoing.

- Block lower-triangular would have allowed us to sweep.
- In pure scattering, there is a singularity in the scattering matrix for OP and SAAF in the spherical-harmonic basis. This is because it is diagonal and the first entry is $1/(\Sigma_t - \Sigma_{s0})$ which is $1/0$

Second-order forms of the Transport Equation

Disadvantages include:

- CFEM methods result in a general sparse matrix, not a block lower-triangular.
- In some forms (EP), reflective boundary conditions have fully implicit coupling between incoming and outgoing flux. SAAF avoids this only coupling incoming to outgoing.
- Full angular flux can be hard to calculate for parity-forms.

Second-order forms of the Transport Equation

Disadvantages include:

- CFEM methods result in a general sparse matrix, not a block lower-triangular.
- In some forms (EP), reflective boundary conditions have fully implicit coupling between incoming and outgoing flux. SAAF avoids this only coupling incoming to outgoing.
- Full angular flux can be hard to calculate for parity-forms.

- Block lower-triangular would have allowed us to sweep.
- In pure scattering, there is a singularity in the scattering matrix for OP and SAAF in the spherical-harmonic basis. This is because it is diagonal and the first entry is $1/(\Sigma_t - \Sigma_{s0})$ which is $1/0$

Second-order forms of the Transport Equation

Disadvantages include:

- CFEM methods result in a general sparse matrix, not a block lower-triangular.
- In some forms (EP), reflective boundary conditions have fully implicit coupling between incoming and outgoing flux. SAAF avoids this only coupling incoming to outgoing.
- Full angular flux can be hard to calculate for parity-forms.
- Difficulties with solving in voids (can be avoided using SAAF).

Second-order forms of the Transport Equation

Disadvantages include:

- CFEM methods result in a general sparse matrix, not a block lower-triangular.
- In some forms (EP), reflective boundary conditions have fully implicit coupling between incoming and outgoing flux. SAAF avoids this only coupling incoming to outgoing.
- Full angular flux can be hard to calculate for parity-forms.
- Difficulties with solving in voids (can be avoided using SAAF).

- Block lower-triangular would have allowed us to sweep.
- In pure scattering, there is a singularity in the scattering matrix for OP and SAAF in the spherical-harmonic basis. This is because it is diagonal and the first entry is $1/(\Sigma_t - \Sigma_{s0})$ which is $1/0$

Second-order forms of the Transport Equation

Disadvantages include:

- CFEM methods result in a general sparse matrix, not a block lower-triangular.
- In some forms (EP), reflective boundary conditions have fully implicit coupling between incoming and outgoing flux. SAAF avoids this only coupling incoming to outgoing.
- Full angular flux can be hard to calculate for parity-forms.
- Difficulties with solving in voids (can be avoided using SAAF).
- Difficulties in a pure scattering media.

Second-order forms of the Transport Equation

Disadvantages include:

- CFEM methods result in a general sparse matrix, not a block lower-triangular.
- In some forms (EP), reflective boundary conditions have fully implicit coupling between incoming and outgoing flux. SAAF avoids this only coupling incoming to outgoing.
- Full angular flux can be hard to calculate for parity-forms.
- Difficulties with solving in voids (can be avoided using SAAF).
- Difficulties in a pure scattering media.

- Block lower-triangular would have allowed us to sweep.
- In pure scattering, there is a singularity in the scattering matrix for OP and SAAF in the spherical-harmonic basis. This is because it is diagonal and the first entry is $1/(\Sigma_t - \Sigma_{s0})$ which is $1/0$

Self-adjoint angular flux equation (SAAF)

Start with the single-group first-order transport equation [6]:

$$\hat{\Omega} \cdot \nabla \psi + \Sigma_t \psi = S\psi + q \, . \tag{1}$$

Solve for ψ,

$$\psi = \frac{1}{\Sigma_t}\left[S\psi + q - \hat{\Omega} \cdot \nabla \psi\right] \, ,$$

and plug back into the gradient term in Eq.1.

$$-\hat{\Omega} \cdot \nabla \frac{1}{\Sigma_t} \hat{\Omega} \cdot \nabla \psi + \Sigma_t \psi = S\psi + q - \hat{\Omega} \cdot \nabla \frac{S\psi + q}{4\pi}$$

With boundary conditions, for all $\mathbf{r} \in \partial D$:

$$\psi = f, \quad \hat{\Omega} \cdot \hat{n} < 0$$

$$\hat{\Omega} \cdot \nabla \psi + \Sigma_t \psi = S\psi + q, \quad \hat{\Omega} \cdot \hat{n} > 0$$

Back to implementation

The Self-adjoint angular flux equation (SAAF) is a second-order from of the transport equation introduced by Morel and McGhee in 1999. To derive, consider scattering term part of tte source. Properties of SAAF

- +Can solve using standard CFEM methods, which give SPD matrices (can use CG instead of GMRES)
- +Full angular flux is obtained by solve (unlike Even/Odd parity)
- +BCs only coupled in one direction when reflective
- -General sparse matrix, not block lower-triangular (no sweeping)
- -Pure scattering causes issues like odd-parity