High Order Discontinuous Galerkin Discrete Ordinates Solver

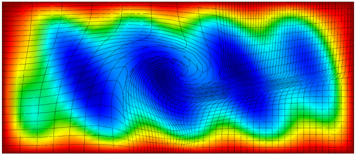
NE 255 Final Project Presentation

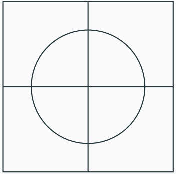
Samuel S. Olivier

November 29, 2018

Motivation

- Finite Element Method
  - Strong mathematical foundation (weak forms, approximation theory) $\Rightarrow$ good error analysis
  - Unstructured grids $\Rightarrow$ match complex geometries easily
- High Order FEM
  - More accurate per unknown (in smooth problems)
  - Use fewer more accurate but more expensive elements $\Rightarrow$ less communication
  - Curved meshes $\Rightarrow$ even better for complex geometry
- Standard continuous FEM not stable for hyperbolic PDEs $\Rightarrow$ Discontinuous Galerkin FEM
- Goal: implement DGFEM for S_N in 1D slab geometry





Linear Transport Equation

- Steady state, one-group, Linear Boltzmann Equation with isotropic scattering and source:

$$\mu \frac{\partial \psi}{\partial x} + \Sigma_t \psi = \frac{\Sigma_s}{2} \int \psi \, \mathrm{d}\Omega + \frac{Q}{2}$$

Streaming Collision Scattering Source

- Discrete Ordinates ($\mathrm{S_N}$) angular discretization

$$\mu_d \frac{\mathrm{d}\psi_d}{\mathrm{d}x} + \Sigma_t \psi_d = \frac{\Sigma_s}{2} \sum w_{d'} \psi_{d'} + \frac{Q}{2}\,, \quad d = 1, 2, \ldots, N_{\text{angles}}$$

where $\psi_d(x) = \psi(x, \mu_d)$

- N_{angles} coupled equations $\Rightarrow$ prohibitively large system of equations

Source Iteration decouples in angle

- Decouple by lagging the scattering term

$$\mu_d \frac{\mathrm{d}\psi_d^{\ell+1}}{\mathrm{d}x} + \Sigma_t \psi_d^{\ell+1} = \frac{\Sigma_s}{2} \sum w_{d'} \psi_{d'}^{\ell} + \frac{Q}{2}$$

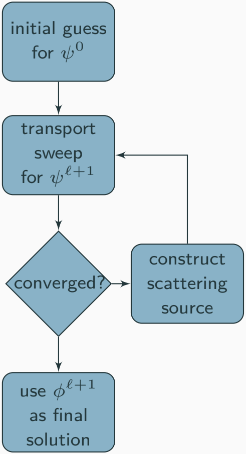
Known from previous iteration

$\rightarrow N_{\text{angles}}$ independent equations but need to solve iteratively

- At every iteration, need to solve

$$\mu_d \frac{\mathrm{d}\psi_d}{\mathrm{d}x} + \Sigma_t \psi_d = S$$

for every angle

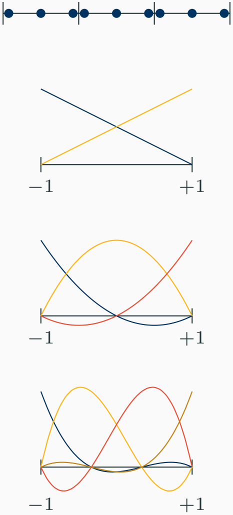


Discontinuous Galerkin Derivation

- Derive for "master" element on $\xi \in [-1, 1]$
- Approximate with linear combination of interpolating, nodal basis functions in each element

$$\psi_d(\xi) = \sum_j \psi_{d,j} b_j(\xi)\,, \quad b_j \in P_k(\xi)$$

- DGFEM procedure:
  - Plug in FEM approximation for ψ_d
  - Multiply equation by $b_i(\xi)$
  - Integrate over the element
  - Use integration by parts to get a boundary term
  - Apply upwinding to boundary term to uniquely define the element edges
  - Map to a "real" element



Discontinuous Galerkin Derivation (cont.)

- Multiply by a "test" function $b_i(\xi)$ and integrate over the "master" element $\xi \in [-1, 1]$:

$$\sum_j \psi_{d,j} \int_{-1}^{1} \mu_d b_i \frac{\mathrm{d}b_j}{\mathrm{d}x} + \Sigma_t b_i b_j \,\mathrm{d}\xi = \int_{-1}^{1} b_i S \,\mathrm{d}\xi$$

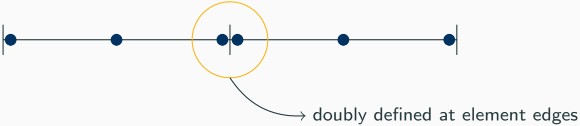
- Integrate by parts:

$$[\mu_d b_i \psi_d]_{-1}^{+1} + \sum_j \psi_{d,j} \int_{-1}^{1} -\mu_d \frac{\mathrm{d}b_i}{\mathrm{d}x} b_j + \Sigma_t b_i b_j \,\mathrm{d}\xi = \int_{-1}^{1} b_i S \,\mathrm{d}\xi$$

- Note

$$[\mu_d b_i \psi_d]_{-1}^{+1} = \mu_d b_i(1)\psi_d(1) - \mu_d b_i(-1)\psi_d(-1)$$

- How to evaluate $\psi_d(\pm 1)$?



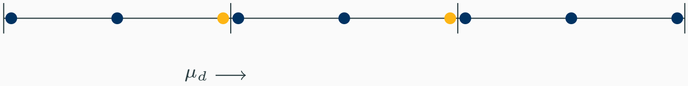
Use upwinding to uniquely define the edges

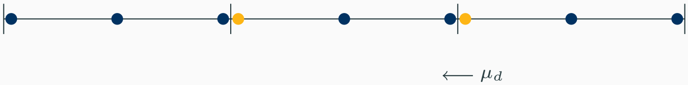
- Upwinding = search backward along particle path

$$\mu_d > 0 : \quad \psi_d(-1) = \psi_{\text{in}} \,, \quad \psi_d(+1) = \sum_j b_j(+1)\psi_{d,j}$$

$$\mu_d < 0 : \quad \psi_d(-1) = \sum_j b_j(-1)\psi_{d,j} \,, \quad \psi_d(+1) = \psi_{\text{in}}$$

where ψ_{in} is incoming flux known from the boundary condition or an upwind element





Map to physical space

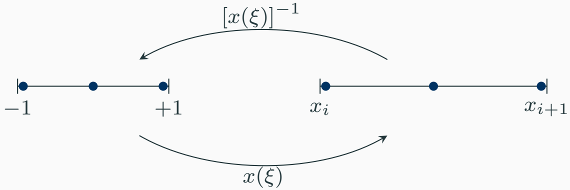
- Convert master to physical space with isoparametric transformation

$$x(\xi) = \sum_i b_i(\xi) x_i\,, \quad J(\xi) = \frac{\mathrm{d}x}{\mathrm{d}\xi} = \sum_i \frac{\mathrm{d}b_i}{\mathrm{d}\xi} x_i$$

  where x_i are the node locations in physical space
- Can integrate any function of x in "reference" space $\xi \in [-1, 1]$ with

$$\int_{-1}^{1} f(x(\xi))\, J(\xi)\, \mathrm{d}\xi$$

- Evaluate numerically with Gauss Quadrature



Matrix Form

Can rewrite in matrix form:

$$[\mu_d \mathbf{G} + \mathbf{M}]\,\vec{\psi}_d = \vec{S} + \vec{B}\,,$$

where

$$\mathbf{G}_{ij} = \begin{cases} \psi_{d,j} b_i \Big|_{\xi=+1} - \int_{-1}^{1} \frac{\mathrm{d}b_i}{\mathrm{d}\xi} b_j \,\mathrm{d}\xi\,, & \mu_d > 0 \\ -\psi_{d,j} b_i \Big|_{\xi=-1} - \int_{-1}^{1} \frac{\mathrm{d}b_i}{\mathrm{d}\xi} b_j \,\mathrm{d}\xi\,, & \mu_d < 0 \end{cases}, \quad \mathbf{M}_{ij} = \int_{-1}^{1} \Sigma_t b_i b_j J(\xi)\,\mathrm{d}\xi\,,$$

$$\vec{B}_i = \begin{cases} \mu_d \psi_{\mathsf{in}} v_i \Big|_{\xi=-1}\,, & \mu_d > 0 \\ -\mu_d \psi_{\mathsf{in}} v_i \Big|_{\xi=+1}\,, & \mu_d < 0 \end{cases}, \qquad \vec{S}_i = \int_{-1}^{1} b_i S J(\xi)\,\mathrm{d}\xi$$

Matrices are $(p+1) \times (p+1)$

DGFEM Sweeps

- Start with $\psi_{\text{in}} =$ known boundary condition
- Solve for the $\vec{\psi}_d$ in the first element with:

$$\vec{\psi}_d = [\mu_d \mathbf{G} + \mathbf{M}]^{-1} \left[\vec{S} + \vec{B}\right]$$

  by computing the inverse (the system is $p+1 \times p+1 \Rightarrow$ small)
- Compute the incoming angular flux for the next element as the outgoing from the previous element

$$\psi_{\text{in}} = \sum_j b_j(1)\psi_{d,j}$$

- Repeatedly solve using the outgoing flux from the previous element until the right boundary is reached
- Same process for negative angles but sweep from right to left

Implementation Details

- Implemented in C++ ($\sim$ 3k LOC!)
- Calls LAPACK for dense linear algebra (matrix inversions, matrix vector products, etc)
- Uses Lua scripts for parameter input i.e. number of angles, number of elements, cross sections, source
- CMake build system
- Outputs in VisIt's curve file format for visualization

Method of Manufactured Solutions

- Choose known solution, plug into transport equation, and solve for the source

$$\psi_d(x) = \frac{1}{2}\sin(\pi x) \Rightarrow \phi(x) = \sin(\pi x)$$

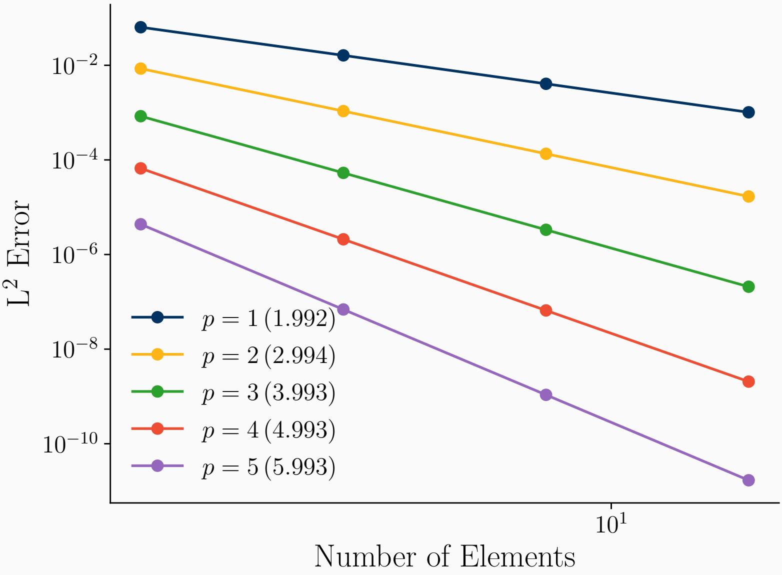
$$\Rightarrow Q_d = \frac{1}{2}[\mu_d \pi \cos(\pi x) + \Sigma_a \sin(\pi x)]$$

- Provides analytic solution to compare to
- Expect

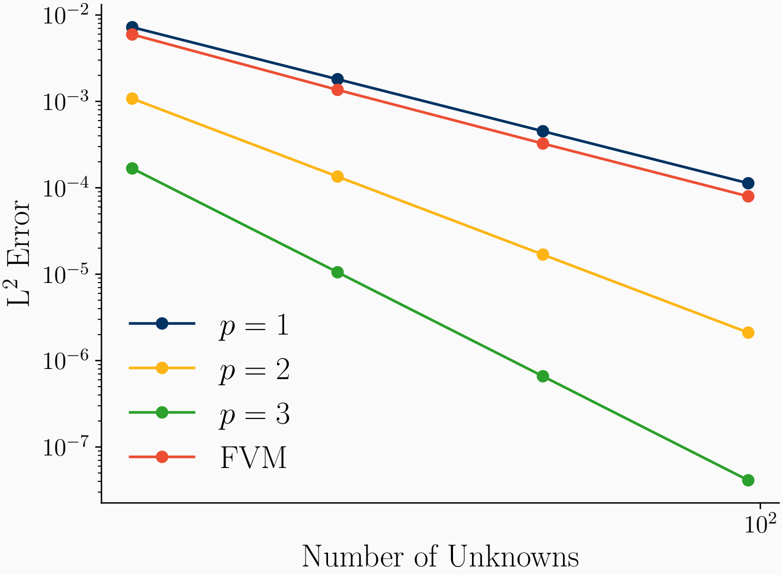
$$E = Ch^{p+1}$$

  in the L^2 norm
- All calculations were S_{32}

MMS test verifies arbitrary order of accuracy

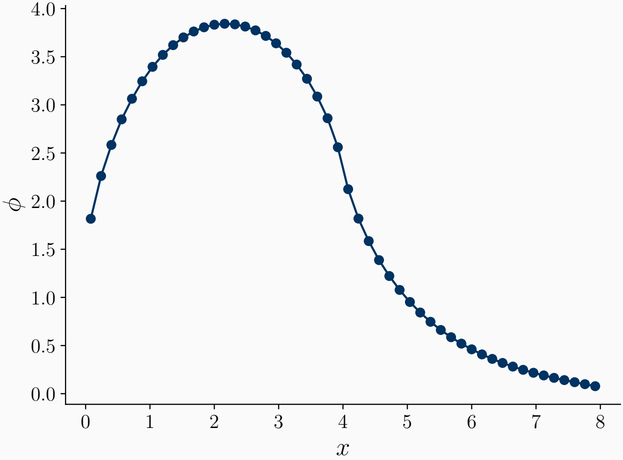


High order is more accurate per unknown



Solution for Inhomogeneous Source

$$\Sigma_t = 1\,\mathrm{cm}^{-1}\,, \quad \Sigma_s = 0.8\,\mathrm{cm}^{-1}\,, \quad Q = \begin{cases} 1\,, & x < 4 \\ 0\,, & x \geq 4 \end{cases}$$



Conclusions

- Implemented DGFEM S_N solver in C++
- Verified code with order of accuracy MMS tests
- Showed that high order is more accurate per unknown

References

[1] M. L. Adams and E. W. Larsen, *Fast Iterative Methods for Discrete–Ordinates Particle Transport Calculations*, vol. 40, Progress in Nuclear Technology, 2002.

[2] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, third ed., 1999.

[3] J. L. Guermond, G. Kanschat, and J. Ragusa, *Discontinuous galerkin methods for the radiative transport equation*.

[4] R. Ierusalimschy, L. H. d. Figueiredo, and W. Celes, *Lua 5.1 Reference Manual*, Lua.Org, 2006.

[5] P. G. Maginot, *Higher Order Discontinuous Finite Element Methods for Discrete Ordinates Thermal Radiative Transfer*, Doctoral Dissertation, Texas A&M University, 2015.

[6] C.-W. Shu, *Discontinuous galerkin methods: General approach and stability*.

[7] B. Whitlock, *Getting Data Into Visit*, Lawrence Livermore National Laboratory, 2010.

Questions?