William Gurecky

ME388F: Computational Methods for Radiation Transport

Final Project

Discrete Ordinates Neutron Transport

with

Finite Elements in 1 and 2D

Commonly when dealing with the spatial discretization of the transport equation an explicit finite differencing scheme is employed. An explicit differencing formulation allows one to "sweep" through the mesh computing by utilizing previously computed flux values (or boundary values) in the trailing nodes to project the flux to the node ahead.

In computation time this strategy scales linearly with the number of nodes. This is predominately why differencing schemes are still commonly employed to resolve the spatial dependence of the flux in the transport equation. While finite differencing is ideal for 1D geometries, in multidimensional problems differencing schemes are difficult to extend to irregular meshes. The simplest codes are limited to rectilinear meshes, representing the geometry with squares and rectangular regions only. This leads to poor capture of curved or organically shaped edges unless one very finely meshes the regions around such geometric features. It is not impossible to construct a differencing scheme for arbitrary non rectilinear meshes, but it would require pre-computing a "sweep map" which determines the node order in which the mesh is swept. In addition the differencing equations themselves become slightly more cumbersome to generate as the nodes are no longer at right angles to each other. This approach is interesting, but is not the focus of this project.

Other spatial discetization techniques are well suited for irregular meshes. Common strategies for discretizing PDE's on irregular meshes are the finite volume and the finite element techniques. We will not consider the finite volume method in this work. The finite element (FE) method takes the approach of minimizing an error function associated with using a linear combination of locally supported basis functions to approximate a true solution. The finite element method possesses the following traits relevant to this project:

1) Easily handle complicated geometries.

2) The simple implementation of the method is only 1^{st} order accurate in space.

3) It creates sparse, tri-diagonal (in 1D) and banded sparse matrices in 2 and 3D.

Method

We begin with the within group, and within angle 1D transport equation (1). Angular dependence in this case is handled by the discrete ordinance approximation. Equation (1) is written for one of these discrete ordinate directions. For each angle and for each energy pair there is a unique first order differential equation describing the spatial dependent transport. The within angle and group equations are coupled to each other by the right hand side source term.

$$\mu \frac{d\psi(\mathrm{x})}{dx} + \Sigma_t \psi(x) = S(x) \qquad \text{Equation 1}$$

The next step is to discretize (1) using the finite element method.

Equation (1) is a first order differential equation, as such the methods introduced in chapter 6 (even parity transport) of E. E. Lewis's computational methods in radiation transport largely do not apply [1]. The even parity formulation of the transport equation contains a spatial derivative of order 2 results in certain nice aspects to the FE discretized equations – primarily that even order equations yield symmetric element matrices. The even parity formulation of the transport equation is not pursued here since it has difficulty including anisotropic scatter [1]. It is possible to use the Petrov-Galerkin method on the first order equation without constructing the second order form of the transport equation.

First a test function $v(x)$ is multiplied to each side of equation 1 and the equation is integrated over the problem domain. This is known as the weak form of equation 1.

$$\int_0^L \left[\mu \frac{d\psi}{dx} v + \Sigma_t \psi v\right] dx = \int_0^L Sv\, dx \qquad \text{Equation 2}$$

Or:

$$\mu v\psi]_0^L - \mu \int_0^L \psi \frac{dv}{dx} dx + \Sigma_t \int_0^L \psi v dx = \int_0^L Sv\, dx$$

We are free to choose a functional form for $v(x)$. In the Galerkin approach, the test function is taken to be of the same form as the solution function [2]. The simple choice is to take $v(x)$ to be a linear combination of hat functions. Each hat function $h_i(x)$ is supported only at one nodal location and is zero at all other nodes. Figure (1) displays two interior neighboring hat functions $h_1\ and\ h_2$ in red and blue respectively. The hat functions are defined to have unit height over their supporting node.

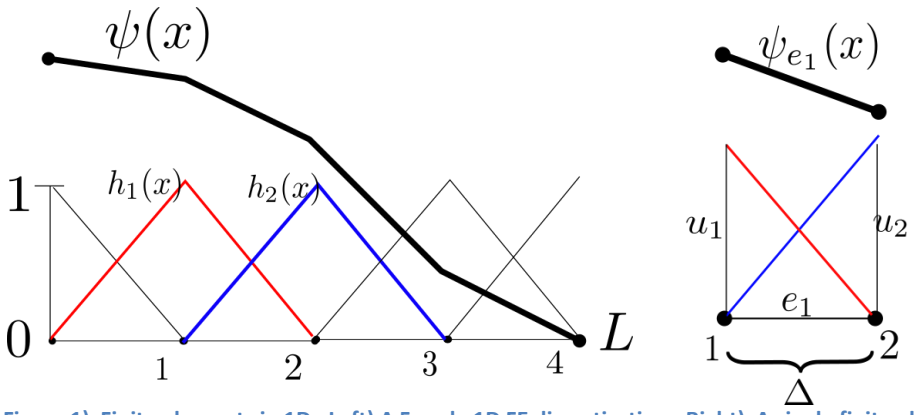


Figure 1) Finite elements in 1D. Left) A 5 node 1D FE discretization. Right) A single finite element between nodes 1 and 2.

An element is defined as the space between two neighboring nodes. The element e_1 is labeled in figure (1) and is bounded by nodes 1 and 2. The hat functions' height are scaled by u_1 and u_2. Upon this element we can write the function:

Equation 3

$$\psi_e(x) = u_1 h_1(x) + u_2 h_2(x)$$

The solution over the entire domain is represented by equation (4). Note that the combination of hat functions forms a piecewise linear representation of the solution:

Equation 4

$$\psi(x) = \sum_{i=0}^{N} u_i h_i(x)$$

Enforce an important choice that was made previously: take the test function to be of the same functional form as the solution function. The Galerkin approach is to use the same functions for both.

Equation 5

$$v(x) = \psi(x)$$

Plugging in equations (3), (4) and (5) into (2), applying boundary conditions and carrying out the integrals will yield a system of linear equations of the form:

$$A\boldsymbol{u} = b$$

Where the unknowns are the hat function scaling factors $\{u_0, u_1, u_2 \ldots\}$. *Note that the scaling factors are the value of the solution at the nodes:* $\{\psi_0, \psi_1, \psi_2 \ldots\} = \{u_0, u_1, u_2 \ldots\}$. In practice, integrating the transport equation of the entire domain at once is not done. Returning to equation (2), rather than integrate over the entire problem domain we focus instead on a single element. Ignore boundary conditions for now and consider an interior element only:

$$\int_1^2 \left[\mu \frac{d\psi_e}{dx} v_e + \Sigma_t \psi_e v_e\right] dx = \int_1^2 S_e v_e \, dx$$

Take S_e to be constant over the element using the midpoint value.

Noting that $\frac{d\psi_e}{dx}$ is constant over the element since we have chosen a piecewise linear representation of the solution, integrating first term yields:

$$W_e = \int_1^2 \mu \frac{d\psi_e}{dx} v_e dx = \mu \left(\frac{1}{2}\right)(u_2^2 - u_1^2) = \boldsymbol{u}\left[\mu \left(\frac{1}{2}\right)\begin{bmatrix}-1 & 1\\ -1 & 1\end{bmatrix}\right]\boldsymbol{u}^T$$

Note that the element matrix produced by the gradient operator is not symmetric and therefore ordering of the nodes from left to right is important.

The second term integrates to:

$$M_e = \left(\frac{1}{3}\right)\Delta(u_1^2 + u_1 u_2 + u_2^2) = \boldsymbol{u}\left[\left(\frac{1}{3}\right)\Sigma_t \Delta \begin{bmatrix}1 & 0.5\\ 0.5 & 1\end{bmatrix}\right]\boldsymbol{u}^T$$

The right hand side is:

$$RHS_e = \left(\frac{u_1 + u_2}{2}\right)\Delta S_e = \frac{\Delta}{2} S_e \begin{bmatrix}u_1\\ u_2\end{bmatrix}$$

We can arrange the weak form of the transport equation integrated over a single element into the following:

$$[W_e + M_e]\boldsymbol{u_e} = RHS_e$$

Where M_e is known as a mass matrix and W_e is an advection matrix.

To construct the system matrix A, the individual element matrices are “stamped” into A. Since each node is assigned a unique ID the elements of A_e can be copied into the global A.

For the first order transport equation, all boundary conditions (vacuum, reflective, white) can be described as either fixed or free. A Fixed boundary condition specifies the value of ψ at the boundary. This arises in the vacuum case where inward facing ordinate fluxes are set equal to zero at the boundary. This also arises in the reflective and white cases where the banked outward fluxes from the previous scattering source iteration are assigned as fixed boundary values for the inward facing ordinate fluxes. A free boundary arises in cases where the flux is allowed to escape from the domain.

To implement a fixed boundary condition, the row in the global system matrix, A, corresponding to the boundary node is set equal to zero at all elements except at the diagonal where the diagonal entry is set equal to 1. On the right hand side the specified value for the flux at that node is set.

Free boundary conditions require no action.

Order of Accuracy

The *displacement error*, or the maximum error of between the piecewise linear approximation to the solution and the exact solution, can be shown to be of order Δx^2. However, the *error in the slope* is only of order Δx. The net result is that the finite element method is only guaranteed to be **1st order accurate** for the transport problem, unfortunately ☹ . See reference [4] and [5] for details on order of accuracy estimation.

2D Finite Elements

The weak form of the 2D angle and energy independent transport equation integrated over the problem domain, *D*, is given in equation (6).

Equation 6

$$\iint_D \left[\mu \frac{\partial \psi}{\partial x} v + \eta \frac{\partial \psi}{\partial y} v + \Sigma_t \psi v\right] dxdy = \iint_D S v dxdy$$

In 2D, the basis hat functions become pyramids, again supported only at one node. An example 2D pyramid basis function is given in figure (2).

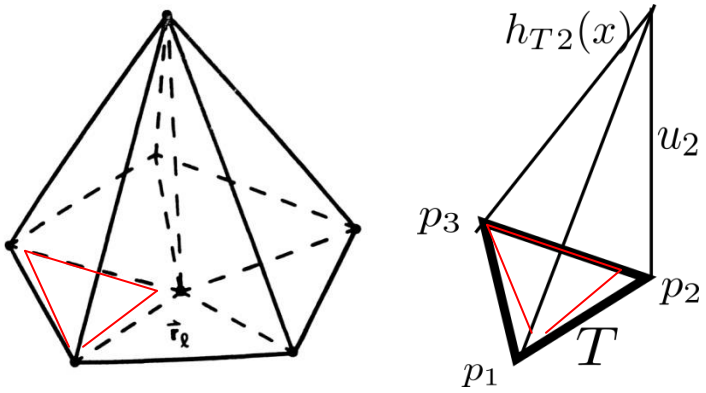


Figure 2) A pyramid type basis function on a 2D triangular grid. Right) a single triangular element with one basis function superimposed overhead.

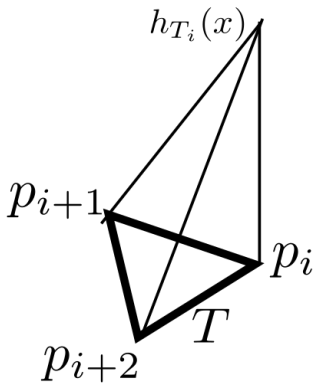


Figure 3) Counter clockwise node labeling scheme. Node indices are linked to basis function indices: eg: $h_{T_i}(p_i) = 1$.

Consider only one triangular element, T. The element has nodes: $p_1 = (x_1, y_1)$, $p_2 = (x_2, y_2)$, $p_3 = (x_3, y_3)$. It is paramount, at least in the implementation of the method, to number the nodes in each triangle in a consistent manner: a counterclockwise numbering scheme is adopted here. i is the node index. Element T is shown with a single basis function $h_{T2}(x)$ drawn above it. $h_2(x)$ is supported at node 2 and it is scaled by the quantity u_2. A linear combination of three such basis functions (each supported at one of the triangle's nodes) describes a linear surface which represents $\psi_T(x, y)$ over the element. The functional form of this surface is:

$$v_T(x, y) = \psi_T(x, y) = a + bx + cy = h_{T1}(x, y)u_1 + h_{T2}(x, y)u_2 + h_{T3}(x, y)u_3$$

The nodal positions of the element can be arranged into a Vandermonde matrix to ease to computation of the coefficients describing the linear surface equation and to compute the area of the triangular element:

$$\boldsymbol{V_T} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix}$$

The area is given by

$$A_T = \frac{1}{2}|\det(\boldsymbol{V_T})|$$

Where each of the three basis function is described by :

Equation 7

$$h_{T_i}(x, y) = \frac{1}{\det(\boldsymbol{V_T})} \det \begin{bmatrix} 1 & x & y \\ 1 & x_{i+1} & y_{i+1} \\ 1 & x_{i+2} & y_{i+2} \end{bmatrix}$$

$$= \left(\frac{1}{2A_T}\right)[x_{i+1}\, y_{i+2} - x_{i+2}y_{i+1} + xy_{i+1} - xy_{i+2} - x_{i+1}y + x_{i+2}y]$$

To describe the linear surface equation we can solve the following systems of equations for a, b, c:

$$\boldsymbol{V_T}[a, b, c]^T = [u_1, u_2, u_3]^T$$

The following is used to interpolate the linear surface at point *w* given the coefficients, a, b, c are known:

Equation 8

$$[1, x_w, y_w][a, b, c]^T = \psi_{T_w}$$

Turning our attention to the transport equation; let $\phi = \{h_{T_i}(x, y)|_{i=1,2,3}\}$ so that

$$\psi(x, y) = u^T \phi$$

Pull all scaling factors, u, from the weak form of the transport equation, integrate over an element, and recall $\psi = v$:

Equation 9

$$u\left[\iint_T \left[\mu\frac{\partial\phi}{\partial x}\phi + \eta\frac{\partial\phi}{\partial y}\phi + \Sigma_t\phi\phi\right]dxdy\right]u^T = \left[S_{T_o}\iint_T \phi dxdy\right]u^T$$

A simple quadrature rule can be used to integrate each term in equation (9) over the triangular element exactly if the integrand function, ϕ (which in this case is a polynomial function) is of order 1 or lower [3]. This works well for the first term and RHS of equation (9).

Equation 10

$$\int_T \phi dxdy = \frac{A_T}{3}(\phi(p_1) + \phi(p_2) + \phi(p_3))$$

The first term is by far the most difficult to integrate, therefore the integration is shown step by step below. Utilizing equations (7), (9), and (10) we obtain:

$$\frac{dh_{T_i}}{dx} = \frac{1}{\det(V_T)}(y_{i+1} - y_{i+2})$$

$$\frac{dh_{T_i}}{dy} = \frac{1}{\det(V_T)}(-x_{i+1} + x_{i+2})$$

$$W_{e_{ik}} = \mu\iint_T \frac{\partial\phi}{\partial x}\phi = \mu\frac{A_T}{3}\frac{dh_{T_{(k+i)\%3}}}{dx}h_{T_i}(x_{(k+i)\%3}, y_{(k+i)\%3}) +$$

$$\eta\iint_T \frac{\partial\phi}{\partial y}\phi = \eta\frac{A_T}{3}\frac{dh_{T_{(k+i)\%3}}}{dy}h_{T_i}(x_{(k+i)\%3}, y_{(k+i)\%3})$$

Where % is the modulus operator. $W_{e_{ik}}$ is a 3x3 element matrix. $\{i\} = \{k\} = \{1,2,3\}$. Note that this element matrix will **not** be symmetric and will contain negative values in some off diagonal positions! This is due to the gradient operator producing a directionally dependent result. If the even parity (or diffusion like) equation were discretized by the FE method we would only have symmetric element matrices since ∇^2produces a scalar quantity.

The computation of the RHS is comparatively simple and is given by:

$$RHS_{e_i} = \frac{A_T}{3}(S_{T_o})$$

RHS_{e_i} is a length 3 vector.

Where S_{T_o} is the source computed at the geometric centroid of the triangular element. In order to do this computation, equation (8) is used to evaluate the linear flux function at the centroid. The centroid flux is then passed to the scattering kernel computation routines.

The second term on the left hand side is of order 2, and therefore requires a more accurate integration scheme. Fortunately a similar rule using the midpoint values produces the exact result for the surface integral [3]:

$$\int_T \phi dxdy = \frac{A_T}{3}(\phi(m_1) + \phi(m_2) + \phi(m_3))$$

Where $\{m\}$ are the midpoint locations on the triangle edges.

The resultant element matrix for the second term on the left is:

$$M_e = \frac{1}{24} 2A_T \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

The resulting element matrix equation is:

$$[W_e + M_e]\boldsymbol{u_e} = RHS_e$$

And when each element matrix is positioned correctly in the global system matrix we are back to the familiar:

$$A\boldsymbol{u} = b$$

Boundary condition assignment is exactly the same as described in the 1D methodology.

A is a sparse, banded matrix in 2D. In the current implementation, the sparse system $A\boldsymbol{u} = b$ is solved by a preconditioned GMRES iterative linear system solver provided by the scipy.sparse package [3]. The preconditioned iterative problem is:

$$Pu_{j+1} = (P - A)u_j + b$$

Where P was chosen to be the diagonal of A in the current implementation. This is the Jaccobi iteration choice. If P is chosen to be exactly A, we get back the Au=b system. The preconditioned problem is significantly faster to solve by iterative methods than the original equation [2]. It is worth noting that solving Au=b will not scale linearly as A becomes large (where A is a N_{nodes} x N_{nodes} sparse matrix). For large problems explicit finite differencing is a superior strategy if low computation time is the primary concern.

Results

Two cases are presented. First a benchmark result shows that the finite element method produces a nearly identical result compared to the diamond differencing scheme for the 1D case. Secondly, a 2D

pin cell k-eigenvalue computation is performed to show the geometric flexibility the finite element technique. Unstructured mesh generation is demonstrated by this case. The 2D keff results are compared to literature values for the eigenvalue.

1D case study

The 1D problem geometry is provided in figure (4) and the composition of each zone is given in table (1).

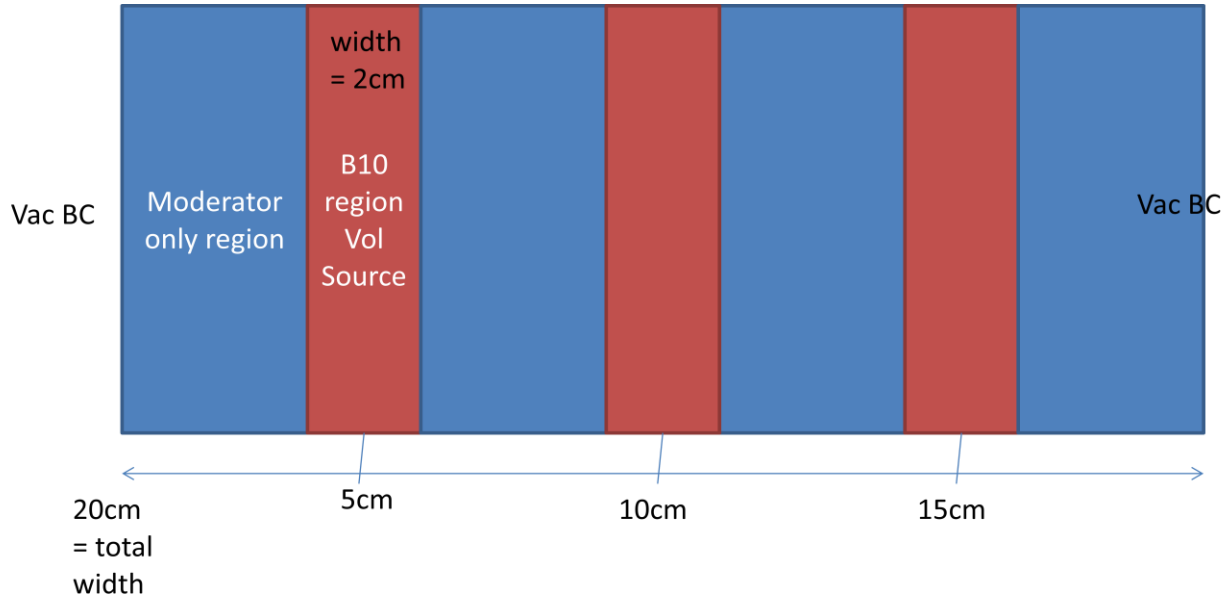


Figure 4) 1D slab problem setup.

The problem specifies a fixed isotropic volumetric source of strength 1e10 [n/cm^3/s] in the borated regions. Each borated region has a width of 2cm. The borated regions are centered at 5, 10, and 15cm. The total width is 20cm. The same node spacing of 0.1cm in the moderator regions and 0.02cm in the borated regions was used in for the finite element calculation and the finite difference calculation. The total number of nodes in both cases was 434. The problem was run using the S_8 Gauss-Legendre quadratue set for both cases. 180 scattering iterations were run in each case.

Table 1) Material composition for 1D benchmark case.

Species	Number Density [#/cm^3]	Location
^{1}H	3.35×10^{22}	Everywhere within 20 cm slab
^{16}O	1.67×10^{22}	Everywhere within 20 cm slab
^{10}B	2.00×10^{21}	Within absorber strips only

The spatial flux distributions as predicted by the diamond difference method are plotted in figure (5) on the right. The spatial flux distributions as predicted by the 1D finite element method are shown on the left.

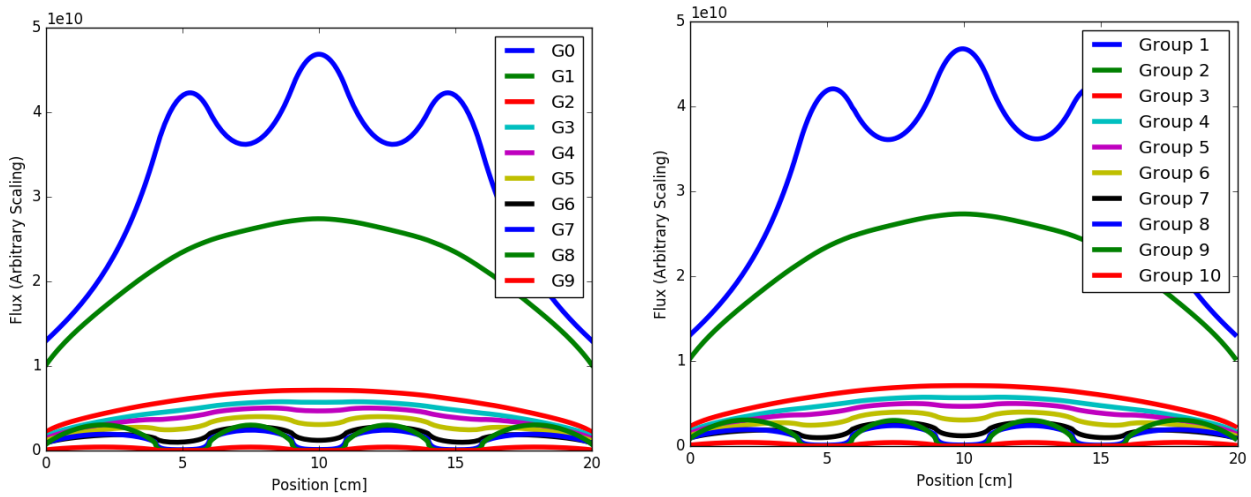


Figure 5) Left) Finite Element solution. Right) Diamond difference solution.

The differences in group fluxes between the two methods is too small to visually see. A truncated set of differences are summarized in table (2). The relative % difference was computed according to: $100(\psi_{g_{FE}} - \psi_{g_{FD}})/\psi_{gFD}$.

Table 2) 1D finite element and finite difference solution differences.

Group	Max Diff (rel%diff)
1	0.62
2	0.31
3	0.29
4	0.33
5	0.43
6	0.73
7	1.66
8	4.76
9	7.53

From this comparison we can conclude that the finite element method produces nearly the same result as the diamond difference method given that the mesh spacing is sufficiently small such that a grid independent solution is reached. Since the finite element method is only first order accurate where the diamond differencing scheme is second order accurate, a finer mesh size would be required in the finite element case to achieve *exactly* the same solution.

2D Case Study

The geometry and material composition of the pin cell is specified in the OECD Burnup credit benchmark report [6].

Table 3) Pin cell geometry.

Geometric Parameter	Value [cm]
Pin pitch	1.33
Fuel Radius	0.412
Cladding Radius	0.475

Table 4) Pin cell composition.

Material	Composition	Density [g/cc]
UOx	3.6% enriched	10.35
Cladding	100% w/o Z90	5.87
Moderator	1/3 a/o O16, 2/3 a/o H1	1.00

The geometry was constructed in an open source CAD/Meshing tool called GMSH [7]. The unstructured triangular mesh used for this problem contained 355 nodes and is shown in figure (6).

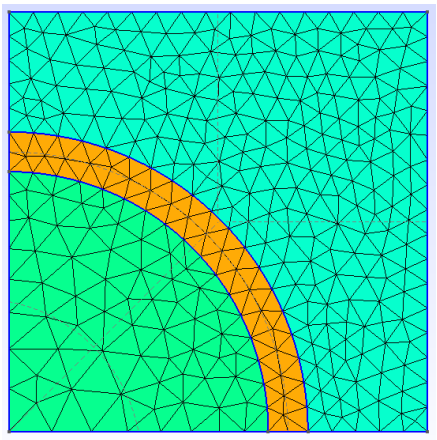
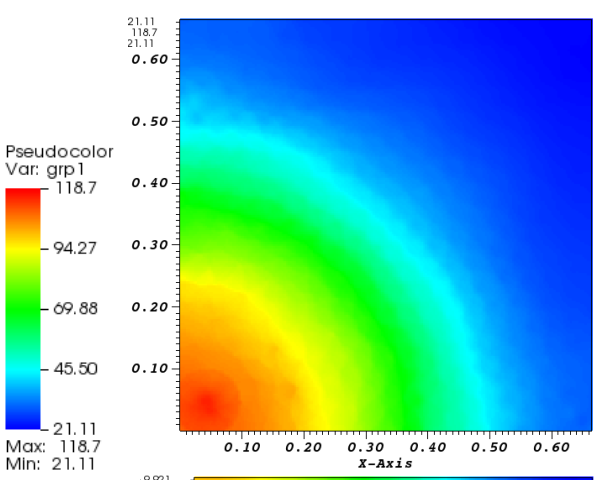
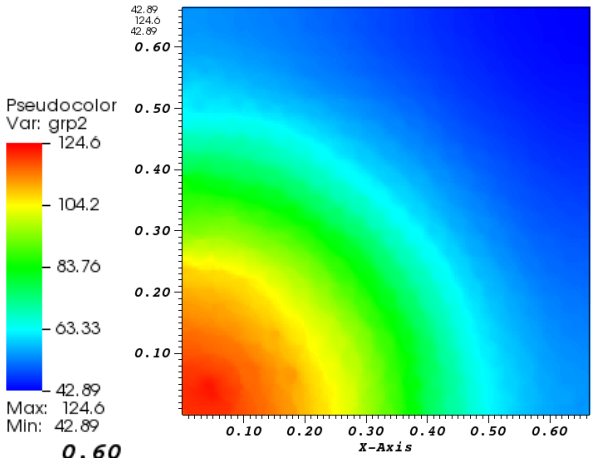


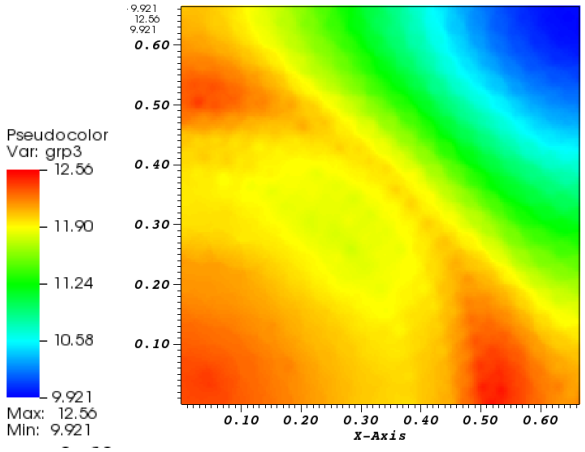
Figure 6) Finite element mesh produced by GMSH [7].

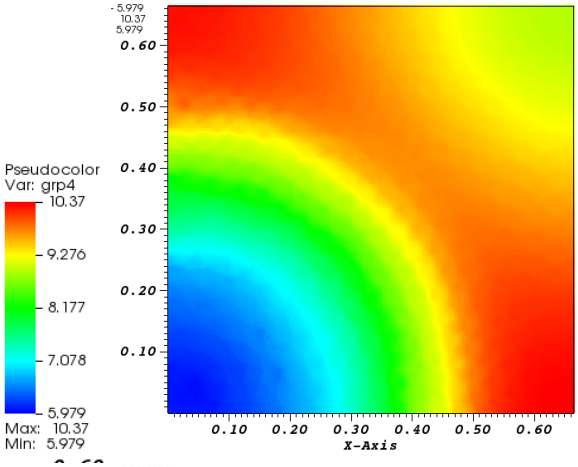
The problem was modeled in quarter symmetry with reflective boundary conditions on all outer faces. The calculation was performed using the S_4 level symmetric ordinate set and in 10 energy groups.

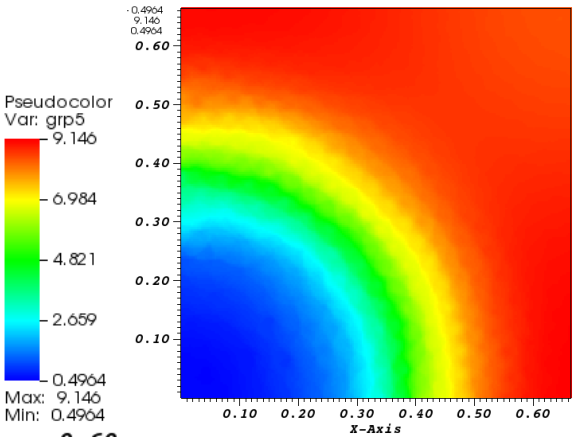
5 k-eigenvalue iterations were performed until a keff tolerance of 5e-3 was achieved. The following images show the final result for the scalar group fluxes. There are visual "bubble" artifacts in the plots that are not really present in the solution because the plotter uses nearest neighbor interpolation when constructing the heat map. For the plotter used (Mayavi2) it was not possible to use a piecewise linear interpolating function that respected the original mesh/node layout.

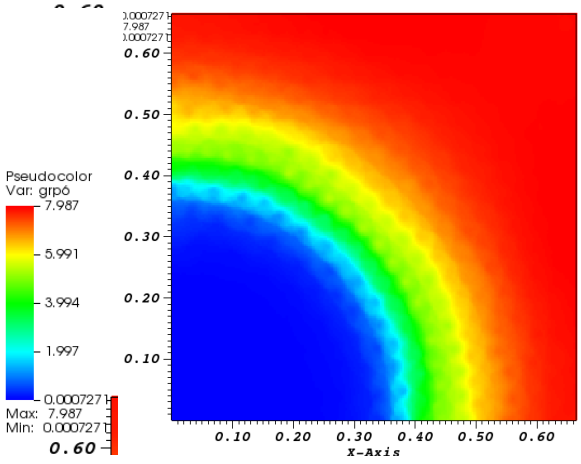

Pseudocolor
Var: grp1
118.7
94.27
69.88
45.50
21.11
Max: 118.7
Min: 21.11
X-Axis


Pseudocolor
Var: grp2
124.6
104.2
83.76
63.33
42.89
Max: 124.6
Min: 42.89
X-Axis


Pseudocolor
Var: grp3
12.56
11.90
11.24
10.58
9.921
Max: 12.56
Min: 9.921
X-Axis


Pseudocolor
Var: grp4
10.37
9.276
8.177
7.078
5.979
Max: 10.37
Min: 5.979
X-Axis


Pseudocolor
Var: grp5
9.146
6.984
4.821
2.659
0.4964
Max: 9.146
Min: 0.4964
X-Axis


Pseudocolor
Var: grp6
7.987
5.991
3.994
1.997
Max: 7.987
X-Axis

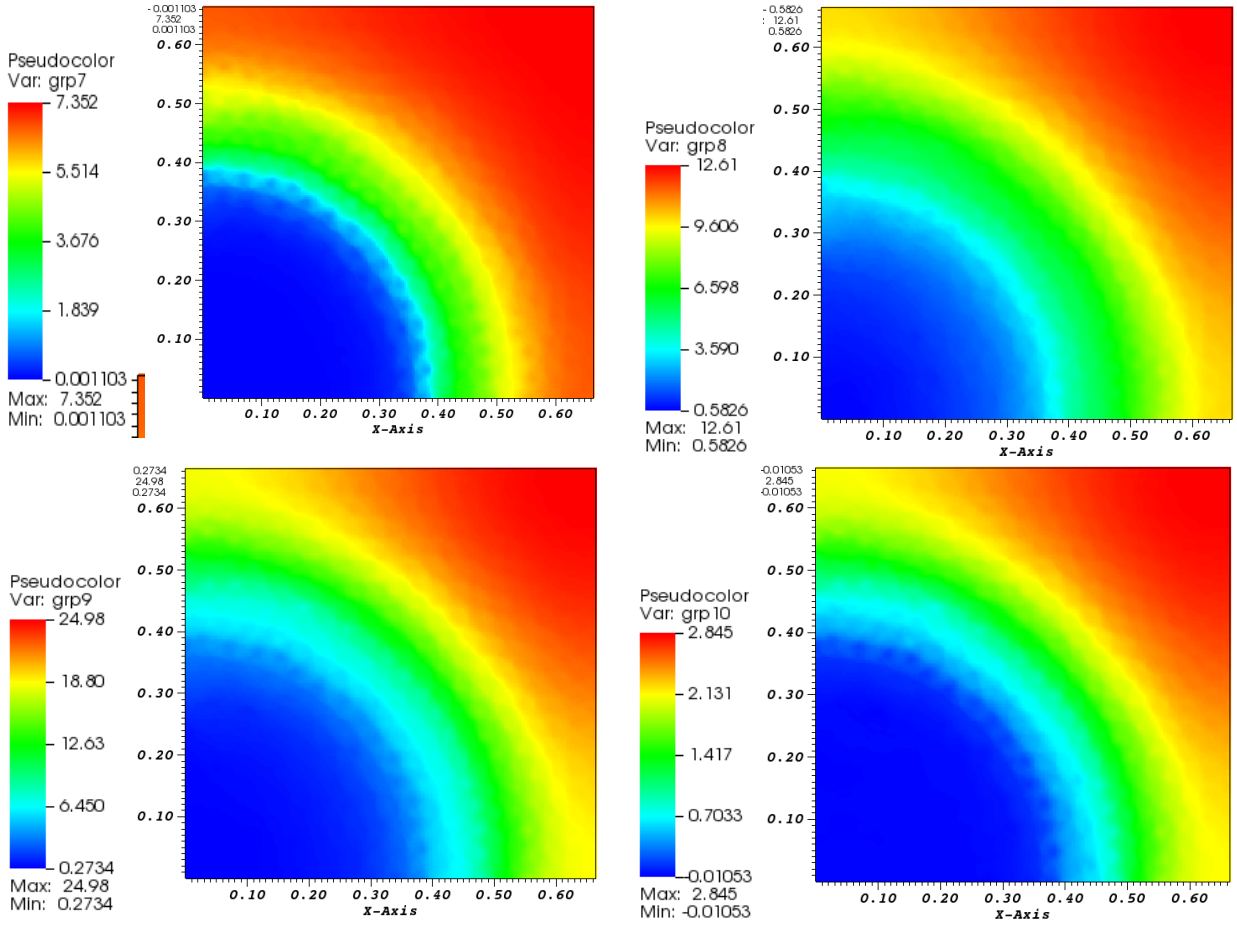


Figure 7) Scalar group fluxes.

Table (5) gives the final keff estimate with a comparison to the value found in the literature. Self shielding was not enabled in this case due to a software bug. The result should improve slightly if self shielding is included. A net increase in keff is expected since the impact of the large 238U capture resonance will be significantly reduced with self shielding on.

Table 5) 2D pin cell keff result.

S_4, 10 group estimate	OECD Benchmark Value +/- 2σ	%diff
1.32251	1.4387±0.0168	8.07%

Conclusion

The finite element spatial discritization strategy was shown to be a viable alternative to the finite differencing scheme. A similar result between the finite element approach and diamond difference was achieved for a 1D problem, though the finite element solution is only 1st order accurate in space. In 2 dimensions, the finite element method allowed for an excellent representation of the geometry and produced an acceptable result for the spatial flux distribution(s) and eigenvalue for a pin cell problem.

The finite element method did not show it's strength until the 2D case. In the 1D case, a similar result was achieved but it was recognized that the finite difference scheme was computationally more efficient. In 2D one can make the argument that achieving a "good" representation of the geometry using finite elements requires fewer nodes than would be required if finite differences were used on a rectilinear mesh.

Additional benchmarks, and mesh convergence studies could be performed as a part of future work.

Improving the order of accuracy of the finite element discretization is another potential avenue for future work. This would involve increasing the polynomial order of the basis functions from 1 to 2.

Future studies could also investigate parallelization strategies. In some respects, it easy to parallelize the spatial transport solve because each angle and energy independent Au=b system can be solve independently. In addition, the scattering source can be updated in each element independently.

Many implementation specific details were not included in this brief report. The code, which is much too lengthy to include here, is available online at https://github.com/wgurecky/spyTran/tree/finiteElm .

References

[1] EE Lewis, WF Miller. Computational Methods of Neutron Transport.

[2]J. Alberty. C. Carstensen, A. Fuken. Remarks around 50 lines of Matlab: short finite element implementation. Numerical Algorithems. v20 pg 117-137. 1999.

[3]FJ. Sayas. A gentle introduction to the finite element method. 2008.

[4] G. Strang. Lectures on the Finite Element Method. http://ocw.mit.edu/courses/mathematics/18-085-computational-science-and-engineering-i-fall-2008/video-lectures

[5] The finite Element Method for 2D Problems. http://www4.ncsu.edu/~zhilin/TEACHING/MA587/chap9.pdf

[6] M. Takano. Burnup Credit Criticality Benchmark Result of Phase 1A. OECD/NEA. 1994.

[7] C. Geuzaine and J.-F. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. International Journal for Numerical Methods in Engineering 79(11), pp. 1309-1331, 2009. http://geuz.org/gmsh/.