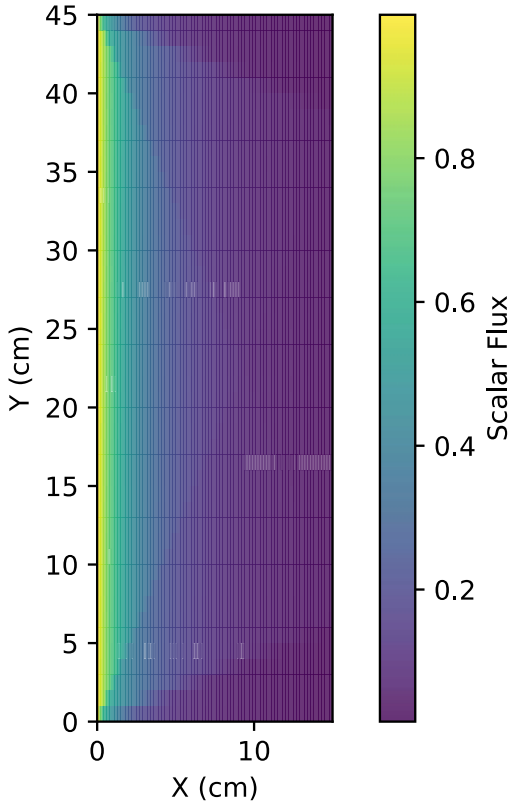
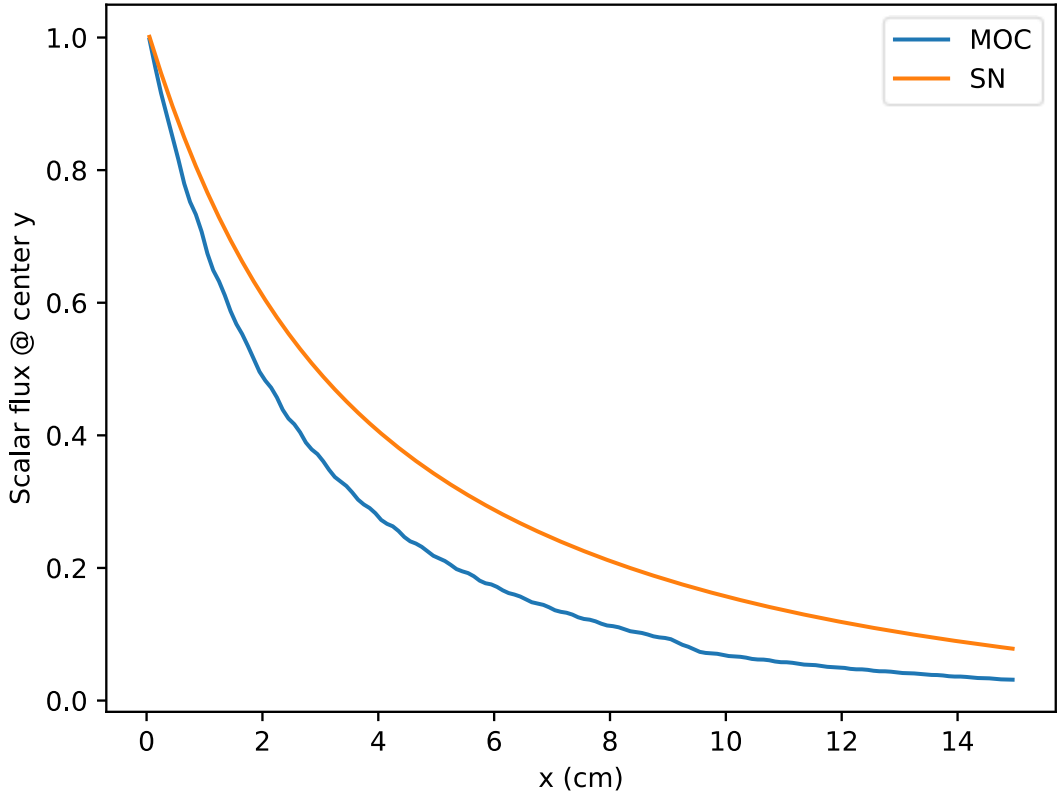
ME388F Final Presentation: MOC Order of Convergence

Braden Pecora

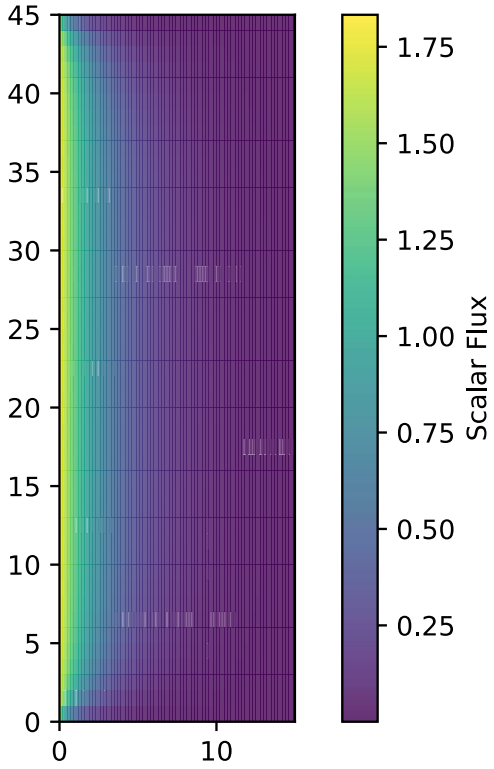
Code to Code Solution Verification

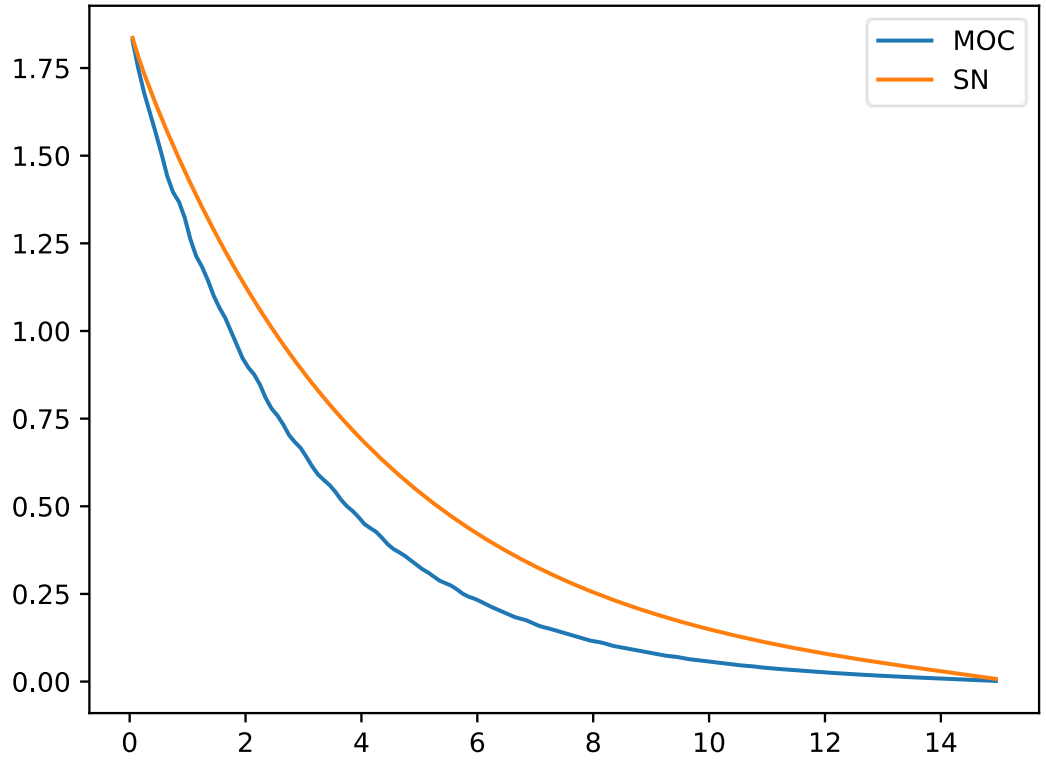
1D: Air


Y (cm)
X (cm)
Scalar Flux
0
5
10
15
20
25
30
35
40
45
0
10
0.2
0.4
0.6
0.8

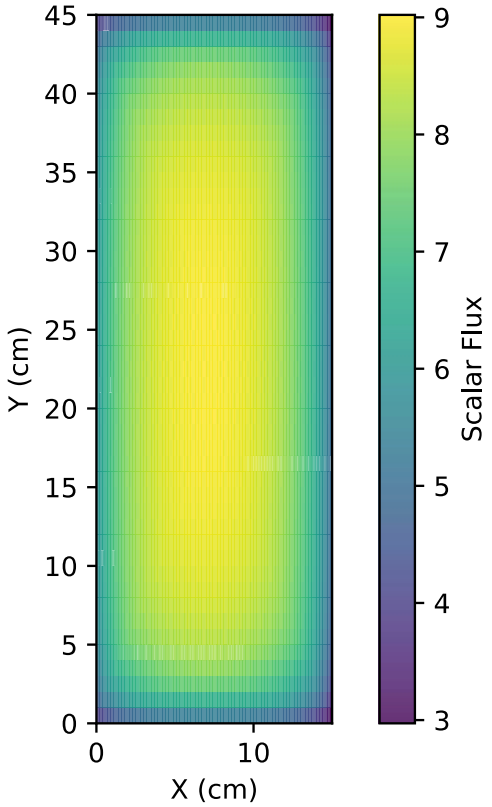

MOC
SN
Scalar flux @ center y
x (cm)
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8
10
12
14

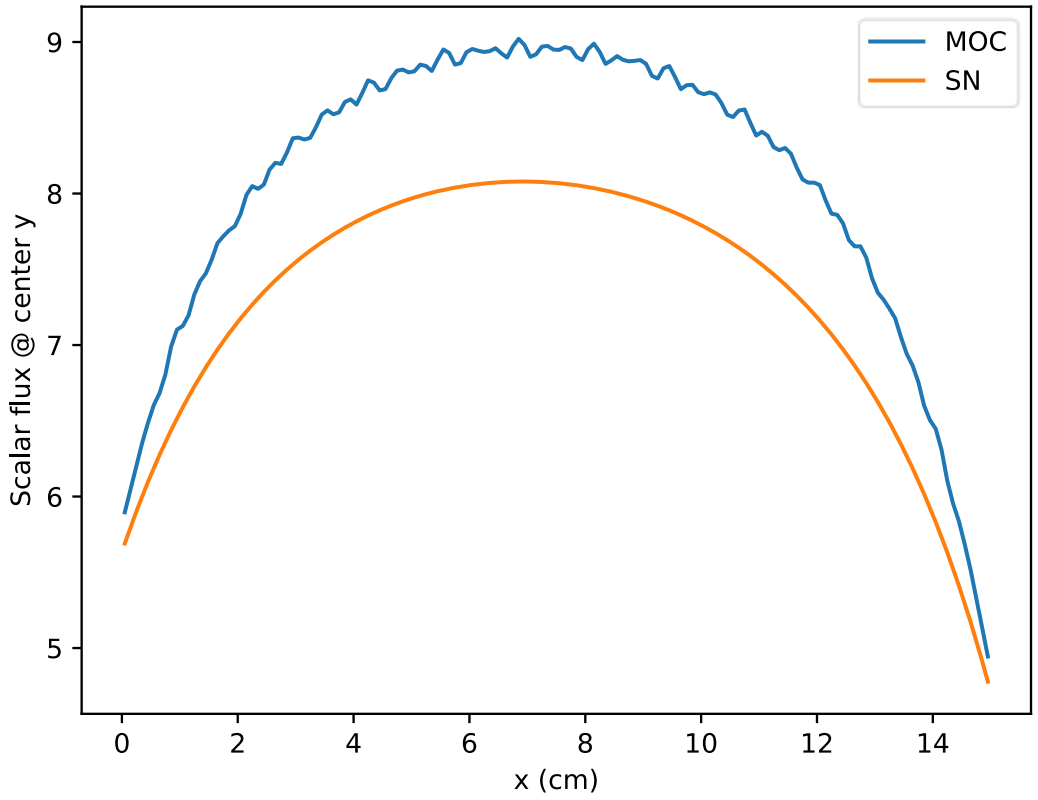
1D: Scatterer



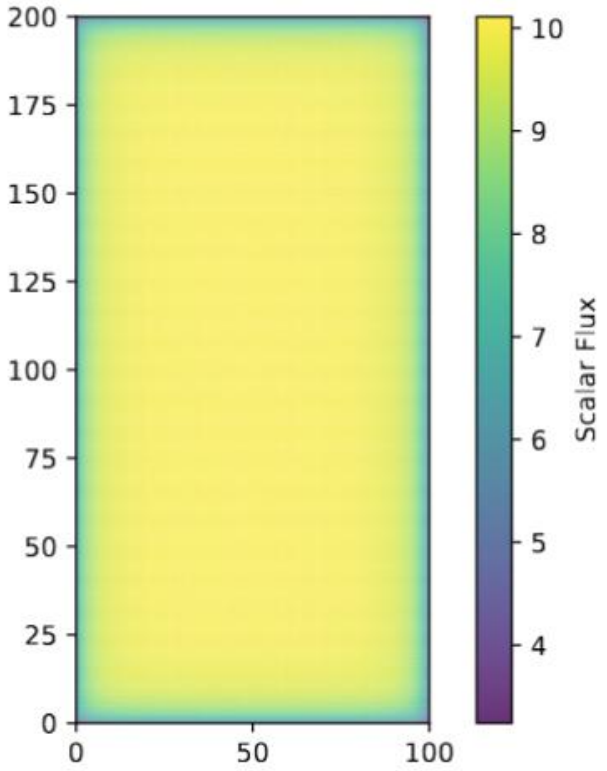


1D Isotropic Source

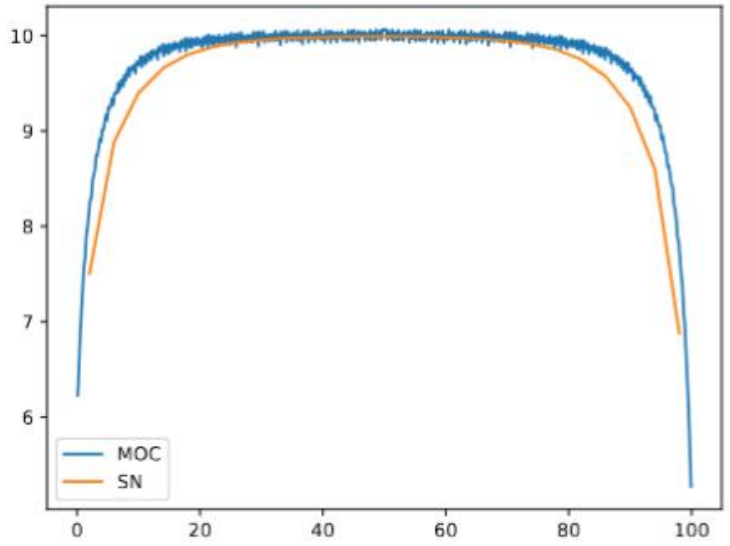




1D Isotropic Source

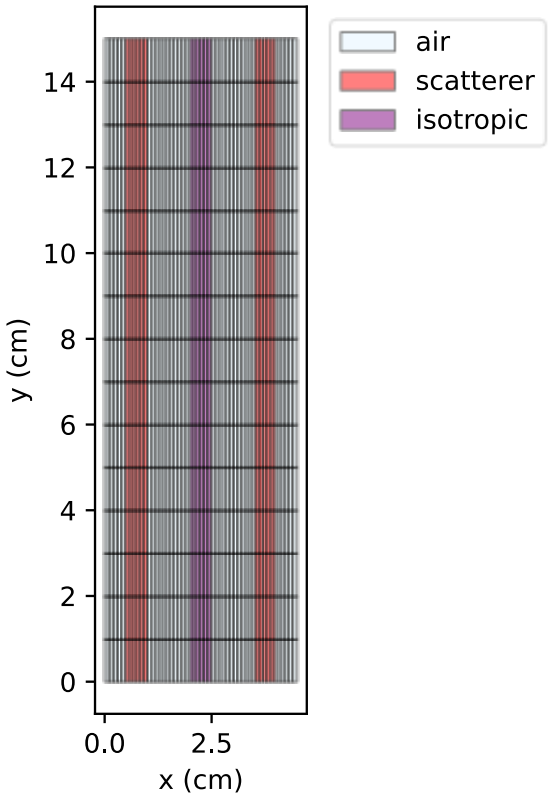


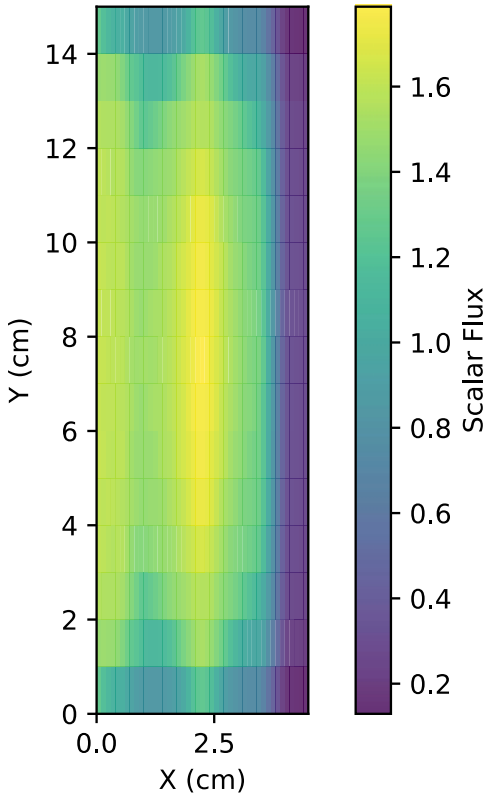
(a) 2D Flux

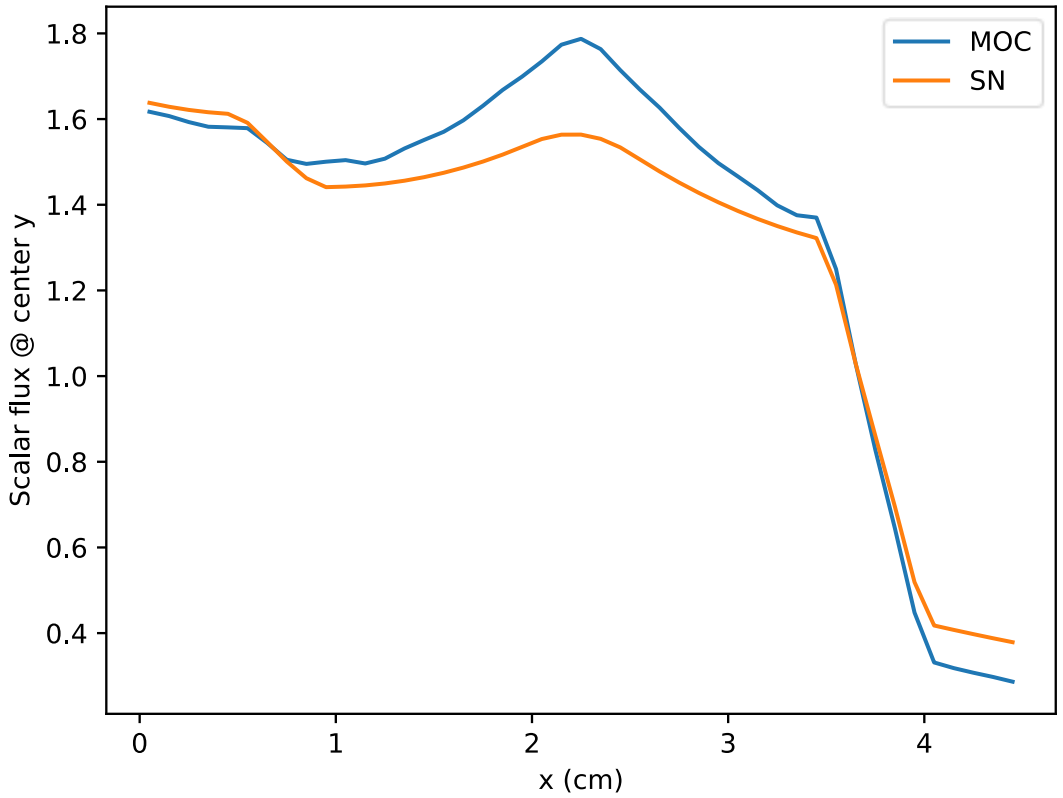


(b) Flux along centerline vs. S_n

1D Varied Material





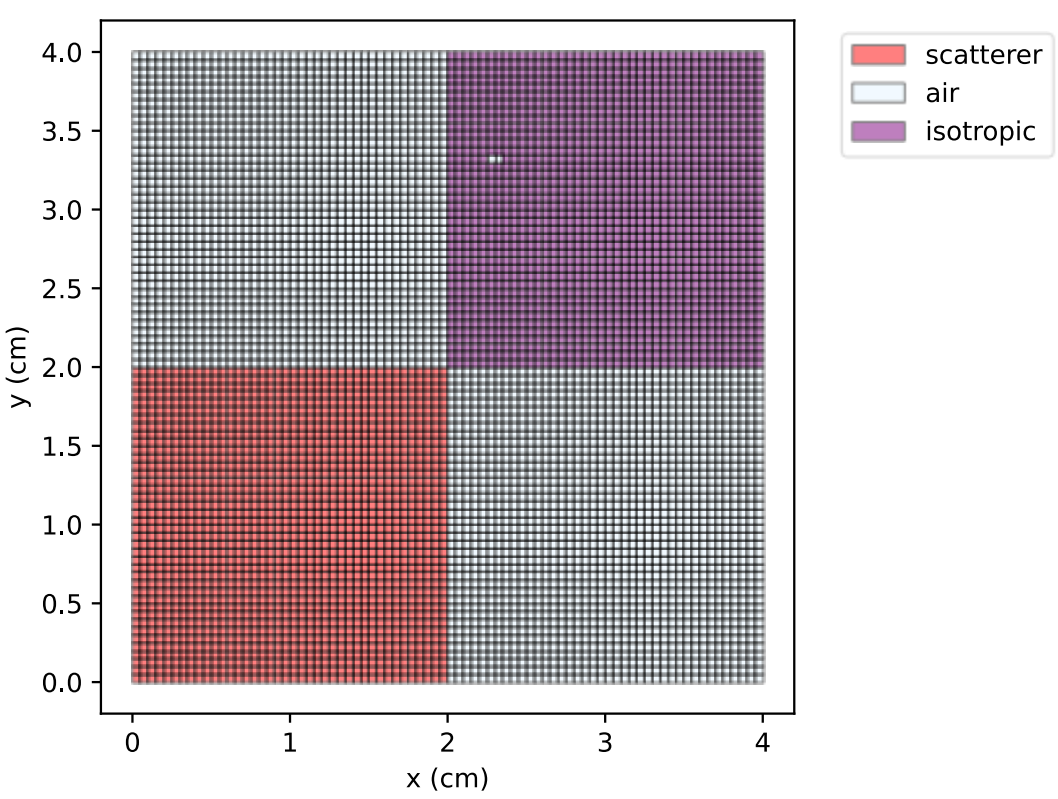


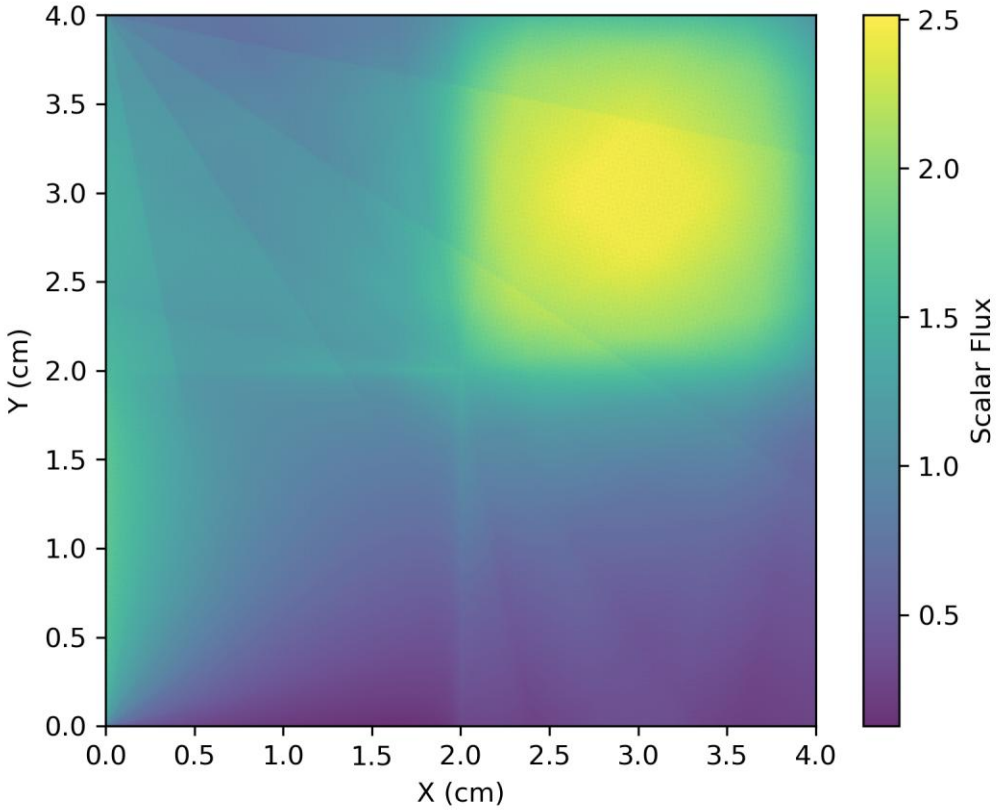
My 2D MOC code agrees somewhat well with my 1D Sn code

- Shapes are mostly correct
- Flux at left and right boundaries was pretty accurate
- Expected symmetry
- Potential room for improvement:
  - Reflecting boundary conditions
  - Increase size of Y domain
  - Problems calculating streaming?
  - Rays that hit corners of cells?

Order of Convergence

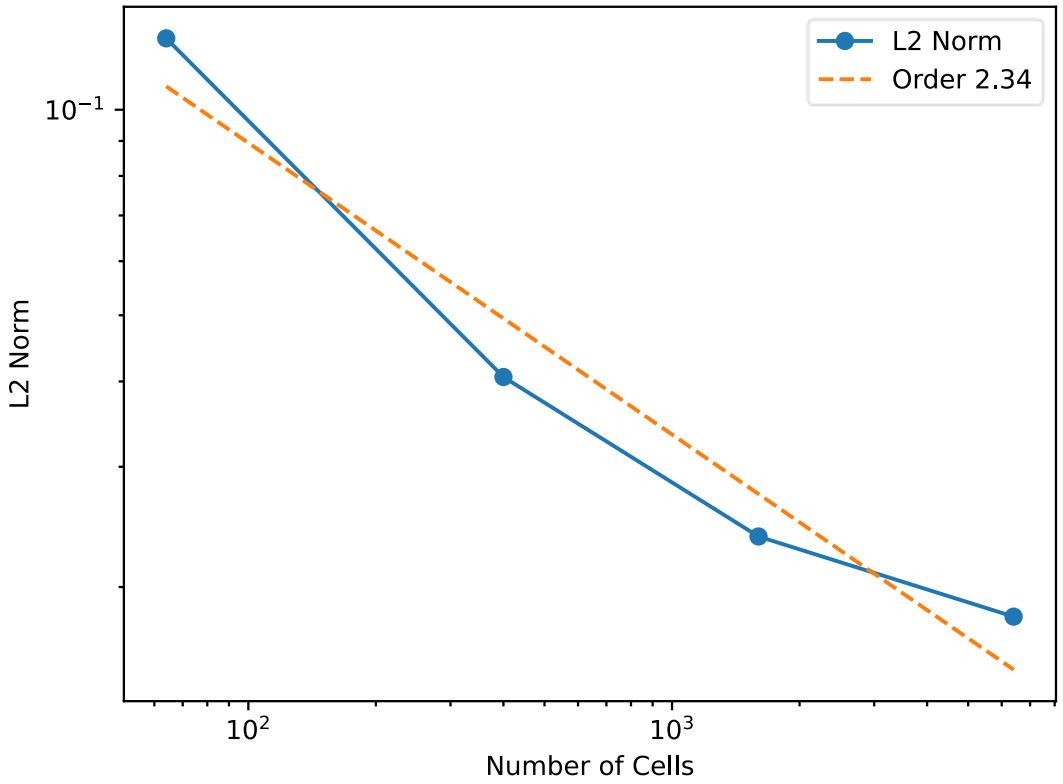
Initial Problem





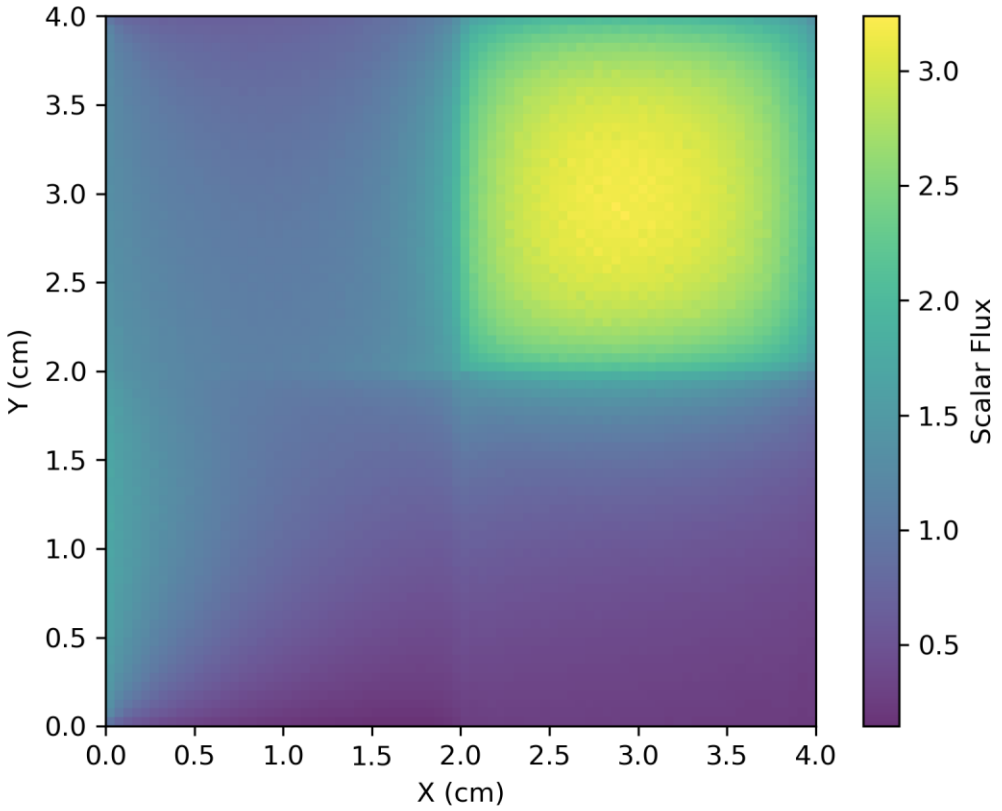
My code exhibits and order of convergence of approximately two with space

- Started with 160,000 cells
- 8 Azimuthal and 4 Polar Angles
- Base solution took ~25 minutes to solve
- Potential sources of error
  - Issues with interpolation?
  - Dynamic ray widths?
  - Ray effects

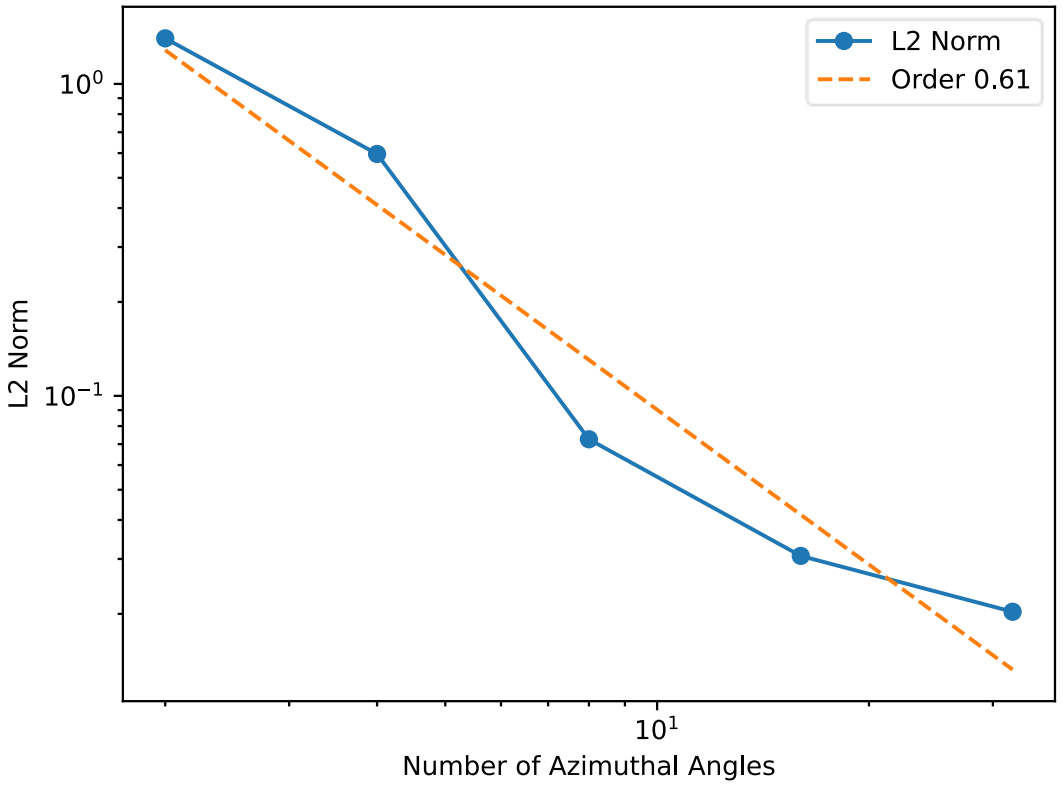


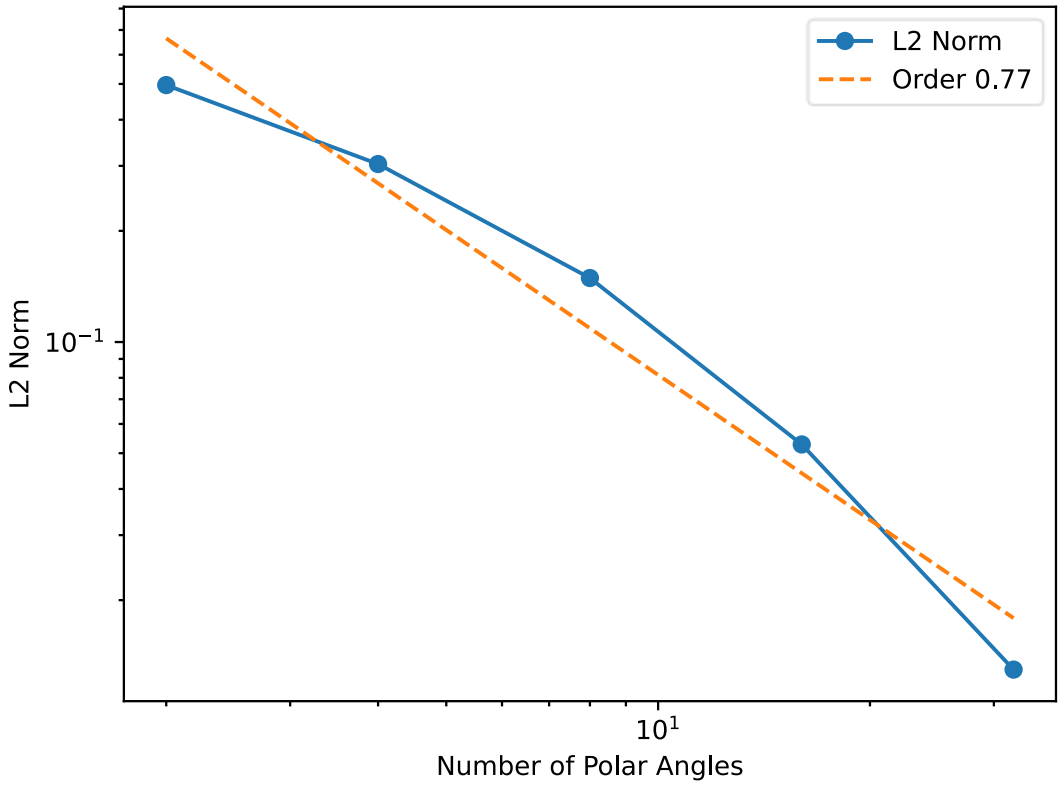
I used a coarser mesh with many angles as the ground truth for angular convergence

- 64,000 cells
- 64 azimuthal and 64 polar angles
- Base solution took ~30 minutes to solve
- No noticeable ray effects in truth
- Varied number of azimuthal and polar angles independently



The order of convergence with each angle is approximately one





I tried many things to speed up my code

- Pre-generate rays (Infinite speed up)
- Threading on ray generation (major slow down to 4x speed up depending on the task)
- Batch processing rays (3-4x speed up on ray generation)
- Compiling the streaming calculation (10x speed boost on iteration)
- Threading ray solves (no speed up)
- Ask Copilot to speed up my code
  - Vectorization – no noticeable speed up
  - Avoiding repeating calculations (~2x speed boost)

Issues encountered

- Shapely did not return intersecting cells in the right order so I had to calculate the order
- Shapely has a lot of overhead (complicated geometry), but I think batching intersection calculations ended up saving me computational time
- Initial fluxes were annoying (no reflecting boundary conditions)
- Typos in the streaming calculation
- Deep copies of objects for some libraries for threading
- Running out of memory