LArTPC Detector Response Calculations
Using **B**oundary **E**lement **M**ethod

Brett Viren

Physics Department

BROOKHAVEN NATIONAL LABORATORY

μBoone Sim

12 Jul 2016

Method Overview

Boundary Element Method (BEM)

1. Discretize (mesh) boundary electrode surfaces.
2. Define (Dirichlet) scalar potential on each mesh element (triangle).
3. Calculate (Neumann) vector normal potential.
4. Integrate Laplace equation $\nabla^2\phi = 0$, evaluate at boundary.
5. Evaluate solution across a **grid of points** in the volume.
   $\rightarrow$ **electrostatic drift** and **Shockley-Ramo weighting** potentials.

Stepping method:

- Instantaneous Shockley-Ramo current in k^{th} wire:

$$i_k = q \times \mu \times (\vec{E}_{weight,k} \cdot \vec{E}_{drift})$$

- Adaptive Runge-Kutta+Cash/Karp stepping through velocity field:

$$\vec{v} = \mu \times \vec{E}_{drift}$$

- Raster each linear step to a scale comparable to **grid points**.
- Discretize current (i_k) samples to make "digitized" waveforms.

Element Methods: Boundary vs. Finite

BEM

- Meshes the surfaces.
- Fast for low surface-to-volume.
- Performance relies on newish mathematical developments.
- Relatively few software implementations.

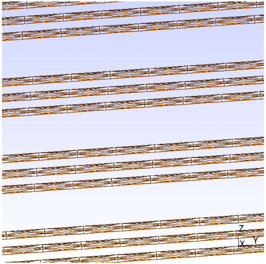
FEM

- Meshes the volume.
- Fast for high surface-to-volume.
- Adaptive meshes can improve performance.
- Many implementations, heavily used in industry.

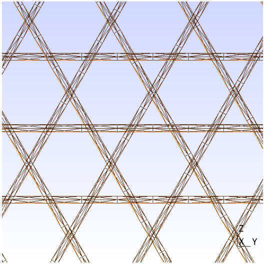
Can also consider a unified/hybrid BEM/FEM calculation.

- BEM in the volume,
- FEM in near the surface.

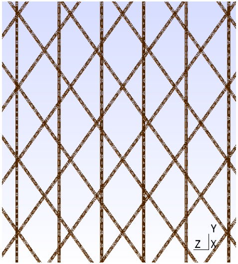
Wire Meshes



“Parallel”:
3mm pitch and gap
all wires parallel



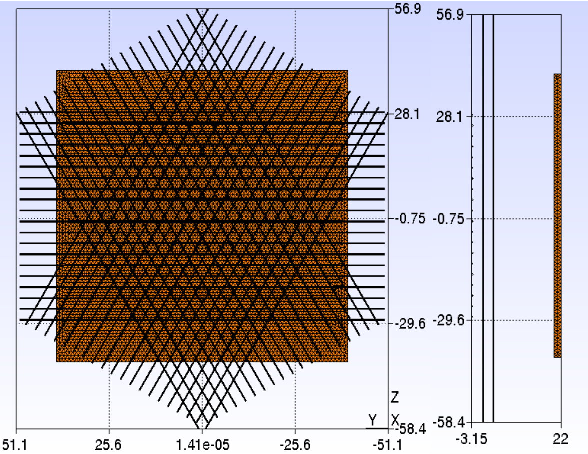
“MicroBooNE”:
3mm pitch and gap
60° angles for U/V.



“DUNE”:
5mm pitch and gap
35.7° angles for U/V.

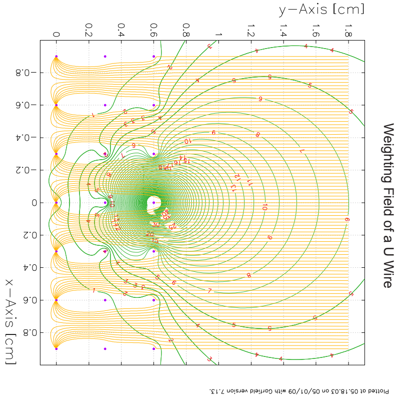
- “Parallel” used to reproduce 2D calculations.
- Geometry parameterized to facilitate exploring different configurations.

MicroBooNE Patch

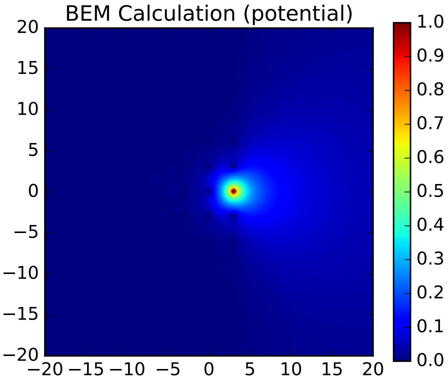


- "Cathode" is close to wires with voltage adjusted to give 500V/cm.
- Do calculations near center, far enough away from edge effects.

Weighting Potential - 2D vs “2D”



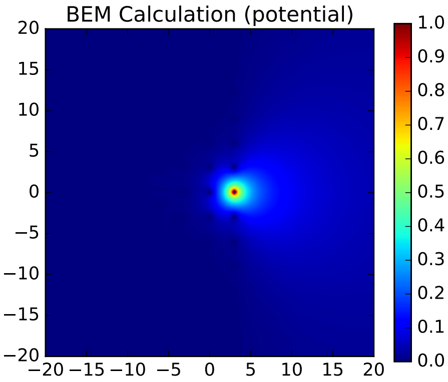
Garfield 2D calculation from Bo



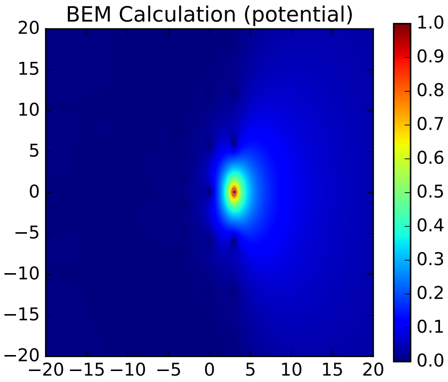
3D BEM, parallel wires, sliced at Y=0.

Initial, qualitative agreement. More checks needed.

Weighting Potential - 2D vs. 3D wire pattern



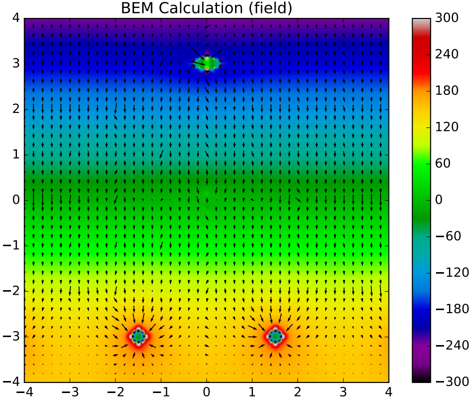
Parallel wires



MicroBooNE wires.

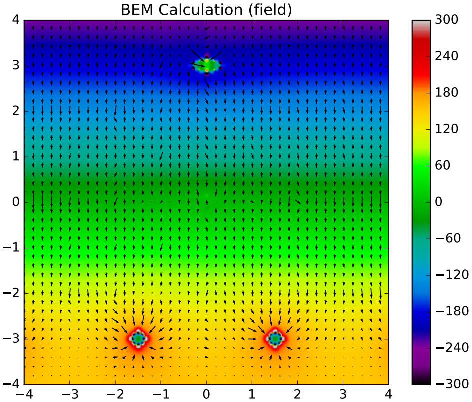
Clear distortion in extent and shape. Not surprising.

An Initial E-Field Calculation



- Square wires? Large fluctuations near wires?
- Weird discontinuities/artifacts in the volume?

An Initial E-Field Calculation



- Square wires? Large fluctuations near wires?
- Weird discontinuities/artifacts in the volume?

What's going on?! $\longrightarrow$

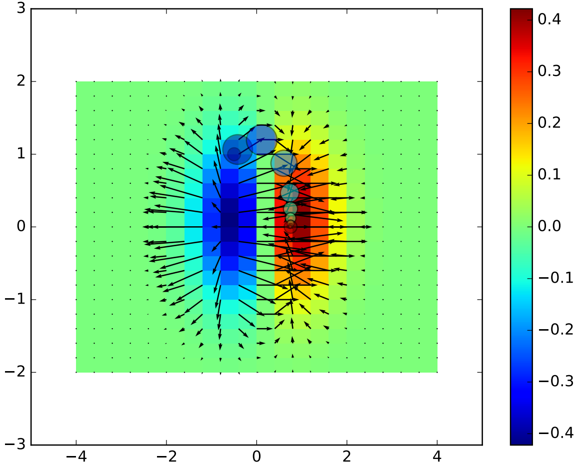
What are These Problems?

Ultimately they are all due to **mesh granularity**.

- Near-surface problems:
  - Mesh approximates real shape with triangles
    - $\Rightarrow$ circular cross-section wires $\rightarrow$ square/hexagon
  - Mesh has sharp edges and points
    - $\Rightarrow$ large local fields (as it should be!)
- In-volume problems:
  - BEM evaluates a sum over pairs of mesh triangles.
  - Triangles are sub-sampled, more for nearby pairs, less for far.
  - As one goes from "close", "medium" and "far" pairs, different number of sub-samples used.
  - $\Rightarrow$ visible artifacts at borders of these different volume domains.

There are "knobs" that **can and must be tuned** to control this.
All trade off running time for accuracy.

Intial Stepping Tests



Just a made-up test "velocity potential". Arrows show local velocity. Circles are steps with time as color and step distance as area.

The Software - Main Dependencies

BEM++

- Solves Laplace, Helmholtz and Maxwell eqns using BEM
- C++ with Python interface.
- General, but low-level. Requires some understanding.
- Multithreaded.

GMSH

- 3D Grid generator and visualization.
- Supports bot volume and surface meshes.
- Interactive and scripted geometry definition.

Python:

- NumPy, Matplotlib, MayaVi, Sqlalchemy/Sqlite3, Click
- all wrapped together by **LARF** $\rightarrow$

LARF - **L**iquid **Ar**gon TPC **F**ield Calculator

Some features:

- Standard Python installation.
- Command line and Python module interface.
- Simple config file specifies the many input parameters.
- Handles everything from geometry to solving to stepping.
- Results stored in Sqlite3 database file.
- Multiple data export methods (Numpy `.npz`, VTK)
- Built-in visualization (2D/matplotlib, 3D/mayavi).

Code and docs on GitHub:

https://github.com/brettviren/larf.

LARF status:

- About **90% feature complete**
    - (the remaining 90% will still take a lot of work!)
    - "done": geometry, meshing, ES potential, R-S potential, mobility/velocity, volume rastering, drift stepping, and current sampling.
- Lots of documentation and some unit tests.

→ Brave users could give it a try.

To do:

- Lots of intermediate diagnostic plots to add.
- Systematic end-to-end validation:
    - Various "does it look right" visualizations.
    - Comparison to 2D Garfield and with Leon's 3D FEM.
- Study response *vs.* step starting positions.

→ Apply to MicroBooNE data to see if removes known problems!

- Tune precision knobs. If required, consider hybrid BEM/FEM.
    - BEM++ hooks into FEniCS FEM.