The Machine Learns Physics

LArTPC Field Response and Detector Design Optimization

Brett Viren

October 13, 2026

Topics

- Learning Laplace
- Applications to wires and pixels
- Going further

Field response construction

Drift potential

$\Phi_d(\mathbf{r})$ potential given geometry and applied voltages.

- Velocity field $\mathbf{v}_d = \mu \mathbf{E}_d = -\mu \nabla \Phi_d$ and drift paths.

Weighting potential

$\Phi_{w,k}(\mathbf{r})$ potential for electrode k at 1 volt and all else grounded.

Field response

Induced current $I_k(t_i) = e\mathbf{v}_d \cdot \mathbf{E}_{w,k}$ with $\mathbf{E}_{w,k} = -\nabla \Phi_{w,k}$ along drift path steps of duration t_i

Solving potentials: Laplace equation with boundary conditions.

$\nabla^2 \Phi(\mathbf{r}) = 0 \quad \text{in } \Omega, \qquad \Phi(\mathbf{r}) = g(\mathbf{r}) \quad \text{on } \partial\Omega.$

Traditional Laplace solutions methods

It is fair to say that BNL is the center for field responses for LArTPC.

- **GARFIELD** provides all LArTPC **2D wire** models.
  - **FEM**, ST CPU, FORTRAN. Yichen, Xuyang.
- **pochoir** provides **hybrid 2D/3D strips, 3D pixels**.
  - **FDM**, MT CPU/GPU, PyTorch. Rado, Yousen, Sergey, Brett.
- **larf** earlier 3D wires, discontinued when MicroBooNE showed 2D is enough.
  - **BEM**, ST CPU, Python/C++. Brett.
- **drifiers** Garfield++, never quite reached escape velocity.
  - **FEM**, ST CPU, C++. Brett.

BNL also provides the primary consumers:

- **Wire-Cell Toolkit** 2D **simulation** and **signal processing**.
- **TRED** GPU-accelerated 2D/3D **simulation**.

A new (to me) method: learning Laplace

$$\theta = \arg\min_{\theta} \left[\lambda_\Omega \left\langle \left(\nabla^2 \Phi_\theta \right)^2 \right\rangle_\Omega + \lambda_\partial \left\langle \left(\Phi_\theta - g \right)^2 \right\rangle_{\partial\Omega} \right]$$

Training finds parameters θ that minimize Laplace equation and the boundary comparison.

- Sampling $\mathbf{r}$: half from the full volume and half in the electrodes.

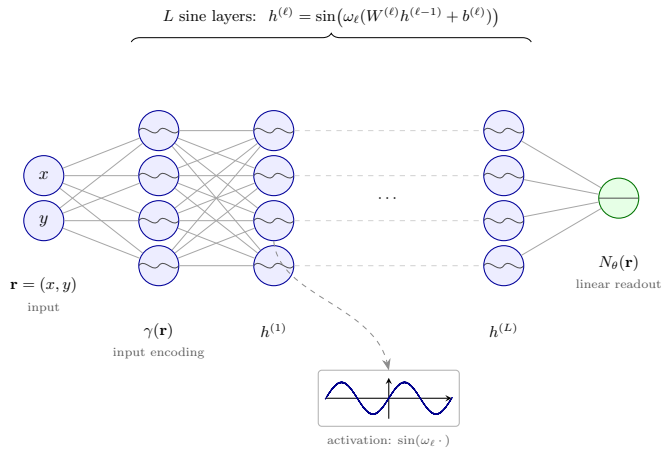
Conceptually simple procedure but what is the form of Φ_θ? Defined as a partition:

$$\Phi_\theta(\mathbf{r}; c) \equiv B_c(\mathbf{r}) + \sigma N_\theta(\mathbf{r})$$

- B_c exact, analytical solution to a different but similar problem.
- N_θ a **network** to learn a correction to the actual problem.

Partitioning minimizes what must be learned, accommodates network architecture limitations.

SIREN[1] is a **multi-layer perceptron** (MLP) with **sinusoidal activations**.



[1] "Implicit Neural Representations with Periodic Activation Functions", Sitzmann, et al. Proc. NeurIPS, 2020. https://arxiv.org/abs/2006.09661

SIREN layers

$$
\begin{aligned}
h^{(0)} &= \gamma(\mathbf{r}) \in \mathbb{R}^{n_0}, && \text{(input encoding)} \\
h^{(\ell)} &= \sin\Big(\omega_\ell\big(W^{(\ell)}h^{(\ell-1)} + b^{(\ell)}\big)\Big) \in \mathbb{R}^{n_\ell}, && \ell = 1, \ldots, L, \\
N_\theta(\mathbf{r}) &= W^{(L+1)}h^{(L)} + b^{(L+1)} \in \mathbb{R}. && \text{(linear readout)}
\end{aligned}
$$

Input is an analytic function and may encode symmetries of a model.
The hidden layers:

- First layer's frequency ω_1 sets a kind of "input bandwidth", deeper layers learn harmonics.
- Learned weights $W^{(\ell)}$ and biases $b^{(\ell)}$, ω_1 frequencies fixed.[2]

Linear readout combines last layer to produce a scalar.

- A vector readout could also be defined.

[2] Typically set $\omega_{\ell>1} = 1.0$. Learned are frequencies possible but they help convergence, not richness.

SIREN limitations and benefits

SIREN limitations

- Effectively has a *bandwidth limit*, poorly represents sharp peaks ($\Phi(\mathbf{r}) \sim \ln \rho$ near wires).
- A global function, but $\Phi(\mathbf{r})$ is undefined outside the boundary (*eg* inside electrodes).

The $B_c(\mathbf{r})$ supplies these problematic parts so SIREN can learn a smooth, global correction.

SIREN benefits

- It is an analytical function (along with B_c) so solution $\Phi(\mathbf{r})$ can be evaluated for any $\mathbf{r}$.
- Differentiable, so we may use fast *back-propagation* training techniques.

We choose $N_\theta \equiv$ SIREN.

Application: wires

Consider a few geometry models and domains:

2D-aligned	the usual model used by GARFIELD.
phase slices	family of 2D slices of 3D along wire direction.
crossed planes	quasi-3D excluding wire-wire charge induction.
full 3D	no approximation tricks.

Notes:

- $B_c(\mathbf{r})$ for wires is generally enough, no SIREN needed.
  - This changes for strips/pixels.
- We may average the last three to get to the standard 2D form.
- The *full 3D* has similar errors as other two, slower but not better.

Wire field errors

solve errors
domain limitation errors

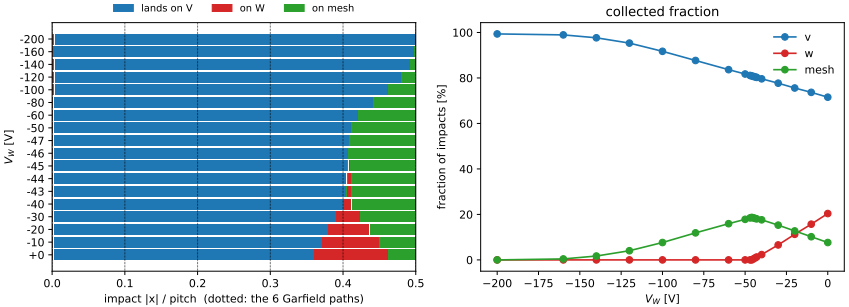
Compute performance

CPU and GPU speed

What can be done with these new types of fields?

- Determine the floating W-plane voltage of “bad” PDHD APA.
- Understand V-plane quasi-collection in 2D and 3D models.

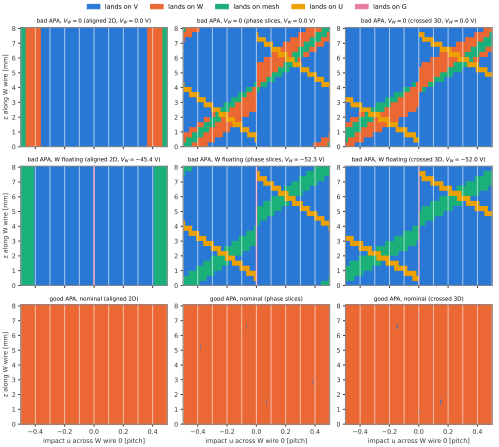
Collecting planes: **2D-aligned** vs W-plane voltage



W-plane collects electrons and charges up until it becomes purely induction-type.

- Resolve fine-grained transverse detail not sampled by Garfield.
- Equilibrium voltage at -47 V depends on geometry details, **and domain dimensionality**.

Collecting planes: **3D / quasi-3D**



Even more high-resolution 3D complexity not available to Garfield 2D.

- **rows**: $V_W = 0$, V_W floating, V_W nominal.
- **columns**: 2D aligned, phase slices, crossed planes.

Application: pixels an grid design optimization

Modified loss functions, results.

Going further

other PDEs
high rez sim for PDHD floating W.
pixel nd design optimization also FD3?