PNPInverse Weekly Development Writeup
Week of February 16, 2026

Jake Weinstein (Project Notes Consolidated)

February 18, 2026

Weekly Focus

This week focused on converting the inverse workflow from isolated scripts into a modular framework, then building a new experiment-style inference path for Robin boundary transfer coefficients from ϕ_{applied}-vs-steady-flux data.
The center of gravity was the new Robin curve inference path: data generation, steady-state detection, adjoint gradient extraction point-by-point, SciPy-based optimization, and recovery logic for forward-solve failures.

Outline of What Was Implemented

1. Unified inverse interface that can plug into compatible forward solvers and infer multiple parameter targets.
2. Always-resilient optimization logic that can recover from nonlinear forward-solve failures.
3. Robin experimental pipeline where the observable is steady-state flux on the Robin boundary.
4. New gradient-based Robin κ inference from flux-curve mismatch using Firedrake adjoints + SciPy.
5. Real-time fit diagnostics: live curve plot with target/current/best and per-evaluation trajectories.

1. Unified Inverse Interface

What was built

A shared inference core was established so new inverse problems can be configured through:

- a forward-solver adapter,

- a parameter target definition,
- a request object containing true value, initial guess, noise, and optimizer settings.

This unifies diffusion, Dirichlet BC, and Robin BC inference under one architecture while preserving compatibility wrappers for existing helper imports.

Mathematical template

For state $u(m)$ produced by the forward solve under control/parameter m, the generic objective template is:

$$J(m) = \frac{1}{2} \int_{\Omega} \|\mathcal{O}(u(m)) - d\|^2 \, dx,$$

where $\mathcal{O}$ selects the measured field(s) (concentration, potential, or derived observable), and d is synthetic/experimental target data.

2. Resilient Optimization for Forward-Solve Failures

What was built

The optimization loop was made always-resilient by adding staged recovery rather than hard-failing on the first diverged PDE solve. Recovery phases were implemented in this order:

1. Increase `snes_max_it`.
2. Apply anisotropy reduction to the trial parameter vector (with reset of relaxed tolerances).
3. Relax nonlinear/linear tolerances (`snes_atol`, `snes_rtol`, `ksp_rtol`) and line-search strategy.

The restart logic uses the last known feasible parameter state, so retries do not repeatedly jump back to the original initial guess.

Anisotropy reduction concept

For a multi-component parameter vector m, anisotropy is treated as a max/min magnitude ratio. If too large, the vector is blended toward an isotropic surrogate based on geometric-mean magnitude:

$$m_{\text{new}} = (1 - \beta)m + \beta m_{\text{iso}},$$

where $\beta \in [0, 1]$ controls flattening strength.

3. Robin Experimental Flux Pipeline

What was built

A new workflow mirrors an experiment where:

- control input is applied voltage ϕ_{applied},
- measured output is steady-state flux through the Robin boundary.

Utilities were added for:

- steady-state probe sweeps,
- synthetic ϕ_{applied}-flux data generation,
- CSV serialization and reuse across fitting runs.

Steady-state flux definition

The Robin boundary law in this implementation is:

$$J_i \cdot n = \kappa_i(c_i - c_{\infty,i}).$$

Species flux across the Robin electrode boundary is assembled as

$$F_i = \int_{\Gamma_{\text{electrode}}} \kappa_i(c_i - c_{\infty,i})\, ds.$$

The default scalar observable used this week is total species flux:

$$F_{\text{obs}} = \sum_i F_i.$$

Steady-state criterion

At each time step n, with per-species boundary flux $F_i^{(n)}$:

$$\Delta_i^{(n)} = |F_i^{(n)} - F_i^{(n-1)}|,$$

$$\text{rel}^{(n)} = \max_i \frac{\Delta_i^{(n)}}{\max(|F_i^{(n)}|, |F_i^{(n-1)}|, \varepsilon_{\text{abs}})}, \quad \text{abs}^{(n)} = \max_i \Delta_i^{(n)}.$$

A step is steady if

$$\text{rel}^{(n)} \le \varepsilon_{\text{rel}} \quad \text{or} \quad \text{abs}^{(n)} \le \varepsilon_{\text{abs}},$$

and steady state is declared after the required number of consecutive steady steps.

How κ Shapes the ϕ_{applied}-Flux Curve

To isolate κ-dependence, a no-noise overlay was generated for:

$$\kappa \in \{[2,2],[1,1],[2,1],[1,2],[1,5],[5,1]\}.$$

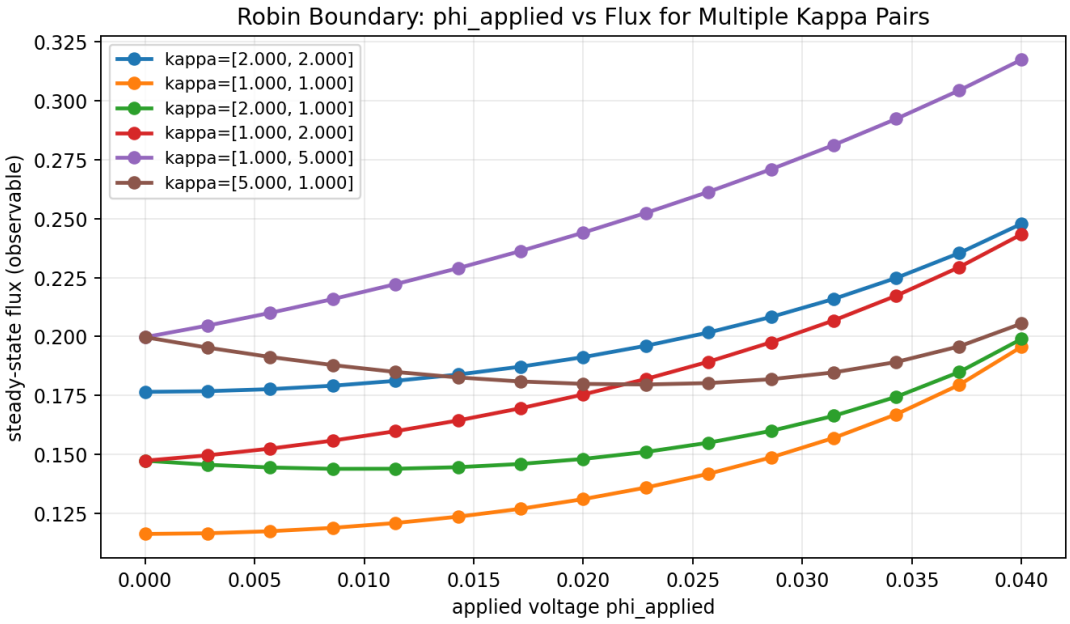

Figure 1: No-noise ϕ_{applied}-vs-steady-flux curves for multiple Robin κ pairs.

Interpretation from these curves:

- Increasing both κ values raises the overall flux level (e.g. $[1,1] \to [2,2]$).
- The two species do not affect voltage sensitivity equally: $[1,5]$ is much steeper than $[5,1]$, even though both have similar zero-voltage flux intercept.
- This asymmetry is consistent with charge-coupled transport in PNP: each species-specific Robin transfer coefficient multiplies a concentration field that responds differently to applied voltage.

Using endpoint slope $\big(F(0.04) - F(0)\big)/0.04$, representative values are:

$$[1,1]:1.984, \quad [2,2]:1.782, \quad [2,1]:1.290, \quad [1,2]:2.400, \quad [1,5]:2.942, \quad [5,1]:0.144.$$

4. Robin κ Inference from Flux Curves

What was built

A dedicated script and helper pipeline were implemented for inferring κ from a full ϕ_{applied}-vs-steady-flux curve.
Per voltage point ϕ_j, a steady-state forward solve is run and compared to target flux $F_j^\star$.

Objective and gradient

Per-point loss:

$$L_j(\kappa) = \frac{1}{2}\left(F_j(\kappa) - F_j^\star\right)^2.$$

Global loss over m sweep points:

$$J(\kappa) = \sum_{j=1}^{m} L_j(\kappa).$$

Firedrake adjoint is used at each point to compute

$$\nabla_\kappa L_j(\kappa),$$

and gradients are accumulated over converged points:

$$\nabla J(\kappa) = \sum_{j \in \mathcal{C}} \nabla_\kappa L_j(\kappa).$$

SciPy minimize pipeline

SciPy receives:

- `fun`: curve loss $J(\kappa)$,
- `jac`: analytic adjoint gradient $\nabla J(\kappa)$,
- bounds on κ,
- method/tolerance/options (e.g. `L-BFGS-B`, `ftol`, `gtol`, `maxiter`),
- callback for iteration diagnostics and plot updates.

5. Noise Model Clarification

Noise is Gaussian with standard deviation tied to RMS signal magnitude:

$$\sigma = \left(\frac{p}{100}\right) \mathrm{RMS}(F_{\mathrm{clean}}),$$

where p is `noise_percent`.
So 5% means $\sigma = 0.05 \cdot \mathrm{RMS}$, not a hard $\pm 5\%$ pointwise cap. Individual points can exceed 5% in magnitude.

6. Representative Results and Plots

Summary Tables

Case	Seed	Mean $\lvert\Delta F/F\rvert$ (%)	Max $\lvert\Delta F/F\rvert$ (%)	Best κ_0	Best κ_1	Final Loss
0% noise	20260220	0.000	0.000	1.0015	1.9980	1.4901×10^{-8}
2.5% noise	20260222	1.790	8.263	1.0165	1.9927	1.6312×10^{-4}
5% noise	20260221	3.908	7.918	1.4533	1.7969	4.0573×10^{-4}

Case	Target κ_0	Target κ_1	Fit κ_0	Fit κ_1	RMSE (Flux)	MAE (Flux)
0% noise	1.0000	2.0000	1.0015	1.9980	4.4573×10^{-5}	3.8524×10^{-5}
2.5% noise	1.0000	2.0000	1.0165	1.9927	4.6636×10^{-3}	3.0610×10^{-3}
5% noise	1.0000	2.0000	1.4533	1.7969	7.3551×10^{-3}	5.3404×10^{-3}

Case	Objective Evals	SciPy Success	Termination
0% noise	20	True	Proj. grad $\leq$ PGTOL
2.5% noise	20	True	Proj. grad $\leq$ PGTOL
5% noise	16	True	Proj. grad $\leq$ PGTOL

Data Generation Curves

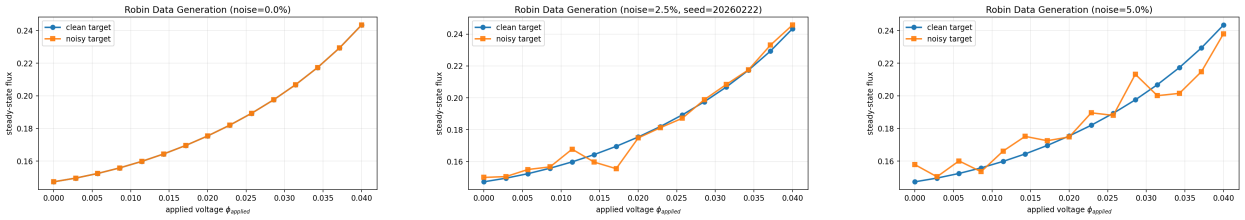

Figure 2: Synthetic Robin flux-curve data generation ordered from least to most noise: 0%, 2.5%, 5%.

Inference Fits

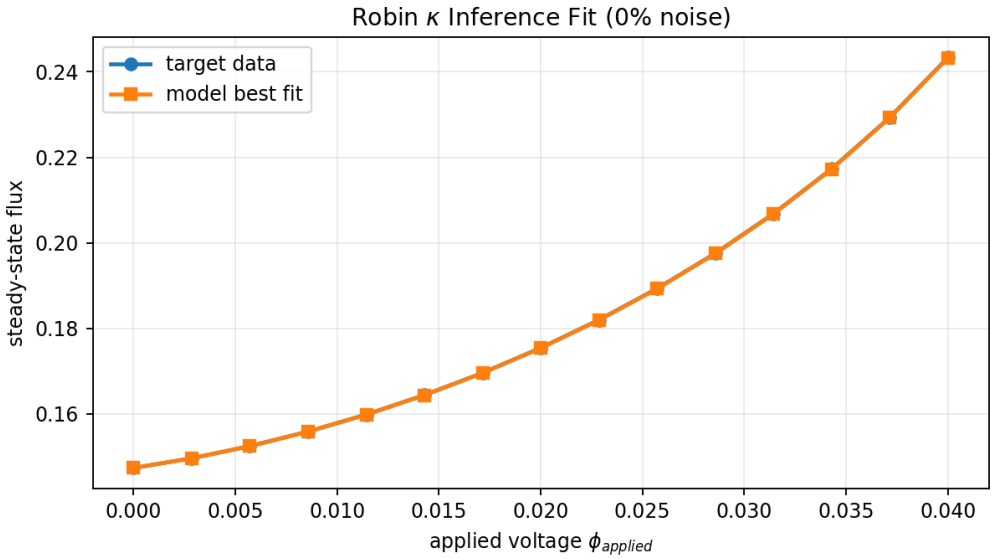

Figure 3: Robin κ curve fit at 0% noise.

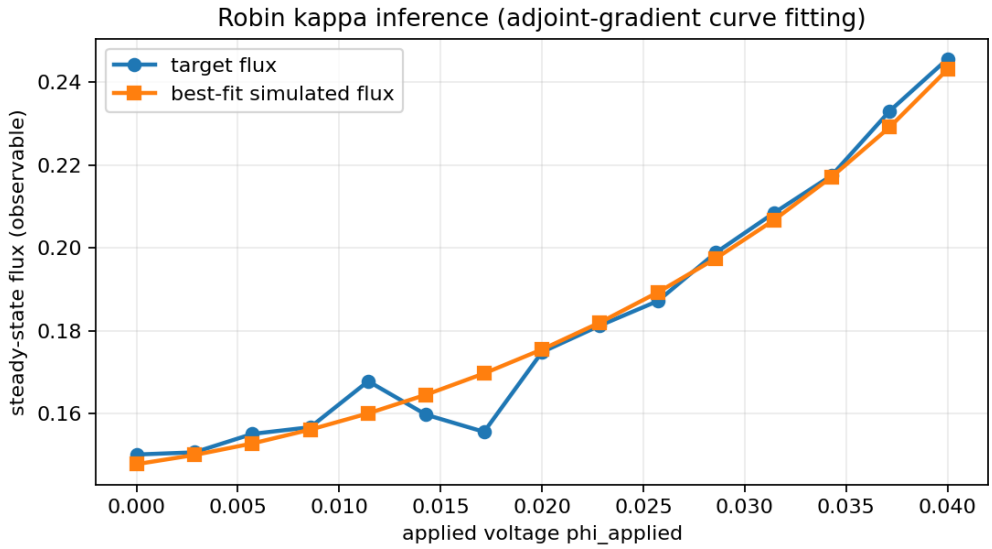

Figure 4: Robin κ curve fit at 2.5% noise.

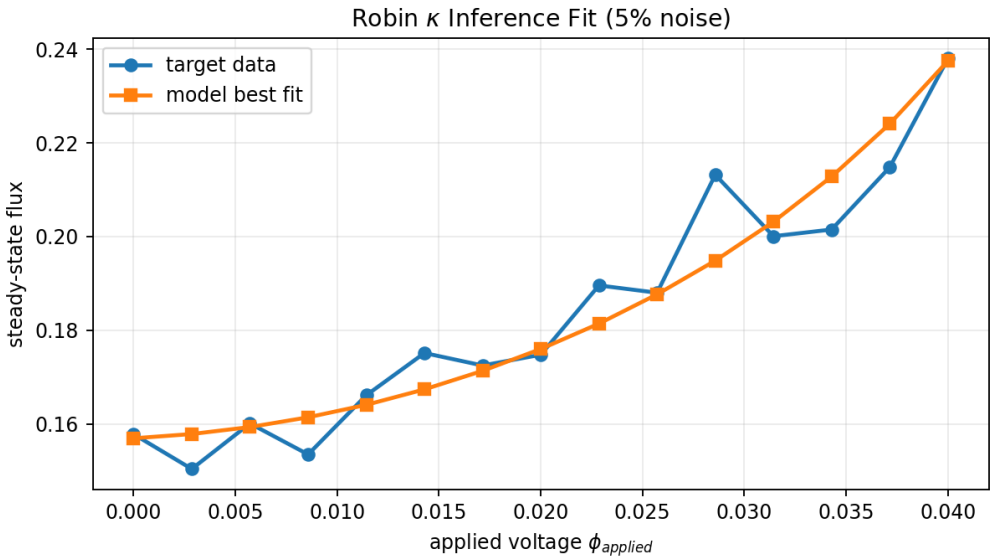

Figure 5: Robin κ curve fit at 5% noise.

7. Practical Takeaways from This Week

- The Robin flux-based inverse workflow is now fully implemented end-to-end and experimentally interpretable.
- The unified interface significantly reduced duplication and made target/solver swapping straightforward.
- Resilient retry mechanics now handle many nonlinear solve failures that previously aborted optimization.
- In noiseless synthetic tests, recovered κ is close to truth; with 5% RMS noise, solutions remain feasible but can shift to different local minima.
- Live plotting materially improved debugging visibility during fitting.