CONSTRAINT CLEANING

Solving PDEs with constraints

Can machine learning control constraint violation inside a solver's time loop?

Murman Gurgenidze

in collaboration with Alberto Roper Pol

Groundwork for the learned-projection hackathon · November 2026

TODAY

Four parts, one question

The problem

Constraints drift. Classical cleaning fixes the constraint.

Our testbed

Relativistic MHD with displacement current, in the Pencil Code.

Where ML fits

Three approaches, and what would count as better.

The hackathon

Levels, data, conventions and the open questions.

THE PROBLEM

Evolved fields must also satisfy a constraint

System	Evolution	Constraint
Incompressible flow	$\partial_t u + (u \cdot \nabla)u = -\nabla p/\rho + \nu \nabla^2 u$	$\nabla \cdot u = 0$
MHD / Maxwell	induction, Ampère–Maxwell	$\nabla \cdot B = 0, \quad \nabla \cdot E = \rho_e$
Yang–Mills	gauge-field evolution	Gauss law (nonlinear)
Numerical relativity	3+1 evolution	Hamiltonian, momentum (nonlinear)

> The continuum dynamics preserve the constraint. Discretization error does not.

> Violations feed back as unphysical forces and growing modes.

THE PROBLEM

The constraint is defined by a discrete operator

"$\nabla \cdot B = 0$" only means something once we fix the derivative operator D.

- Official D: the Pencil Code sixth-order first-derivative stencil
- Its symbol $s(k)$ vanishes at the Nyquist wavenumber
- Exact projection = FFT with $s(k)$: passes the check to round-off

THE RESIDUAL WE REPORT

$$r = \frac{\|D \cdot u\|}{\|Du\|}$$

CLASSICAL REMEDIES

Four standard fixes, and where they stop

Method	Idea	Cost per call	Exactness
Constrained transport	discretize so $\nabla \cdot B = 0$ holds	built into update	exact by design
Projection (FFT / Poisson)	remove the gradient part	$\mathrm{O}(N \log N)$, global	exact or to tolerance
Hyperbolic (GLM) cleaning	propagate and damp the error	$\mathrm{O}(N)$, local	approximate
Constraint damping	terms that drive violation to 0	$\mathrm{O}(N)$, local	approximate

Work well for linear constraints with known structure.

Expensive, hand-tuned or unavailable for nonlinear constraints.

STARTING POINT

Hard projection on a neural operator

Reproduced Duruisseaux et al. (2024): FNO time-stepper, 2D Kolmogorov flow, Re = 500, 64^2, output projected in Fourier space.

Method	Rel. L^2 ($\times 10^{-2}$)	RMS $\nabla \cdot u$
Baseline FNO	3.30	1.6×10^{-1}
+ projection at inference	3.22	5.5×10^{-6}
+ projection fine-tuning	3.26	5.5×10^{-6}
Trained with projection	3.19	5.5×10^{-6}

~97%

of the error is already divergence-free.

Projection fixes the constraint for free, but cannot touch the rest.

PART 2

Our testbed: relativistic MHD with displacement current

Where Gauss's law becomes a dynamical constraint, and what we learned from running it.

OUR TESTBED

Why keep the displacement current

$$\partial_t E = \nabla \times B - J, \quad \nabla \cdot E = \rho_e \quad \Longrightarrow \quad \partial_t \rho_e + \nabla \cdot J = 0$$

Gauss becomes dynamical

$\nabla \cdot E = \rho_e$ stays true only if the current conserves charge at every step.

Ohm's law breaks it

J comes from Ohm's law, not its own equation, so nothing in the update enforces continuity.

The charge is evolved

3D runs advance ρ_e separately (Brandenburg & Protiti 2023), so $\nabla \cdot E$ and ρ_e drift apart.

OUR TESTBED

A fully relativistic Ohm's law, with J^0 evolved

$$J = J^0 u + \gamma\sigma \left[E + u \times B - (u \cdot E)\, u\right]$$
$$J^0 = \gamma\left(\rho_e + \sigma\, u \cdot E\right)$$
$$\partial_t J^0 = -\nabla \cdot J$$

$$\nabla \cdot E = J^0 \quad \text{(enforced by damping)}$$

J^0 is the lab-frame charge density; it is evolved, not diagnosed.

Four-force on the fluid enters the momentum equation.

Damping: $\nabla \cdot E$ is fed back into $\nabla \cdot J$, driving the Gauss violation to zero.

Roper Pol & Midiri 2025, arXiv:2501.05732,

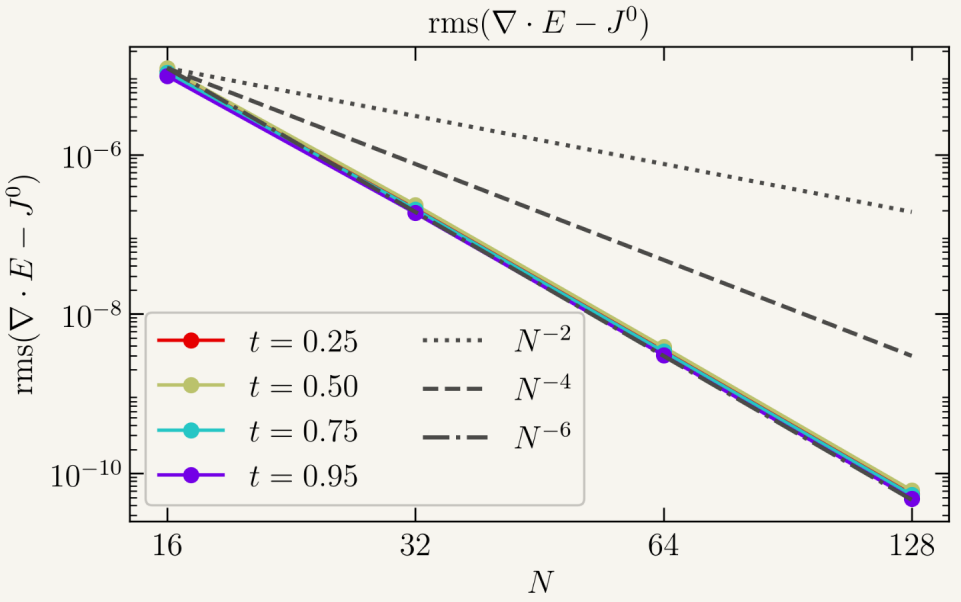
OUR TESTBED

Implemented in Pencil, verified by convergence

5.7–6.0

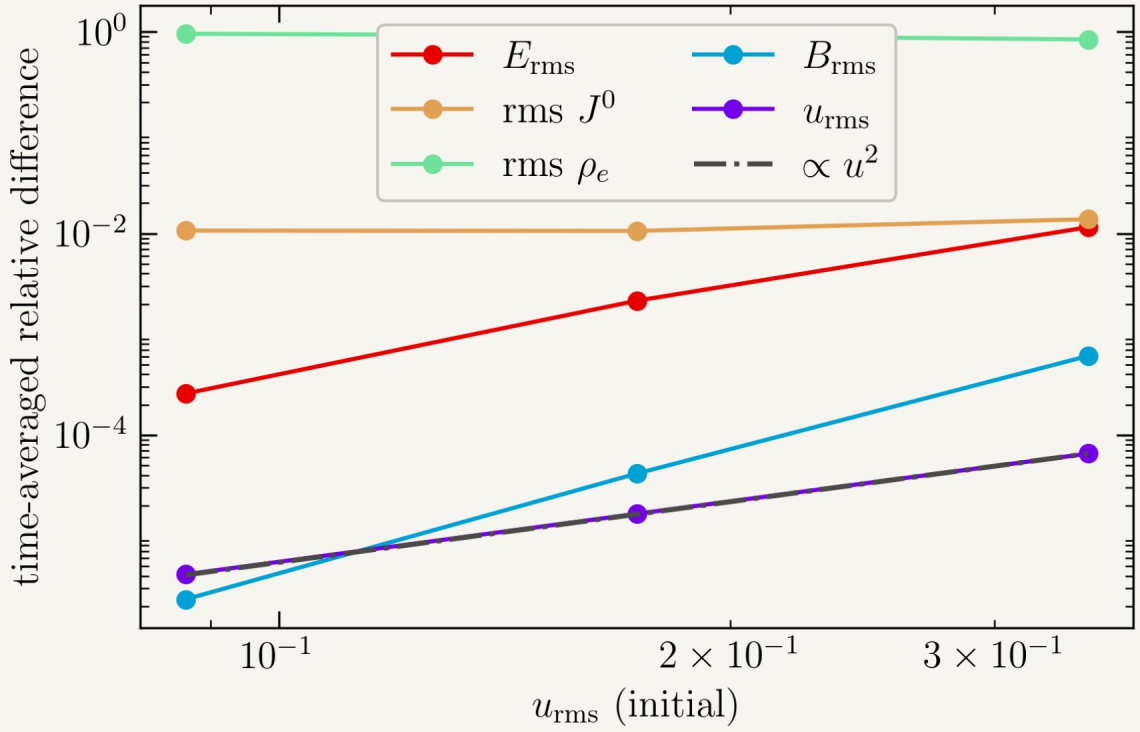
convergence order of the Gauss and spectral residuals, 16^3 to 128^3: pure truncation error

- Switch lsolve_J0 in the displacement-current module
- No Γ in the evolved equations: curl curl A computed directly
- $\sigma\, W \cdot \nabla\gamma$ term is ~39% of $\nabla \cdot J$ even at $|u| < 0.25$



WHAT WE FOUND · PHYSICS

Relativistic corrections grow steeply with u



GALILEAN VS RELATIVISTIC OHM'S LAW

$$E \sim u^3, \quad B \sim u^4$$

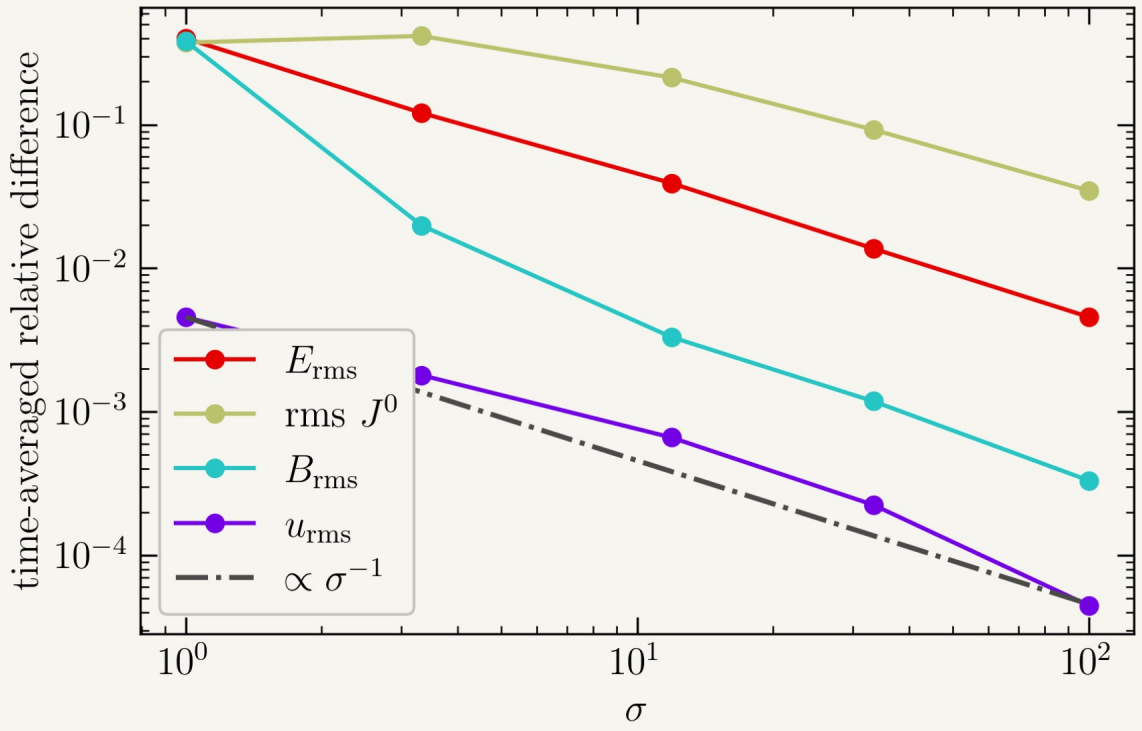
At $u_{\mathrm{rms}} = 0.35$, E differs by 1.2% and J^0 by 1.4%.

A partial $\gamma = 1$ model is worse than Galilean (J^0 8.5%).

Resolution error in these comparisons ~10^{-8}.

WHAT WE FOUND · PHYSICS

The displacement current matters at low σ



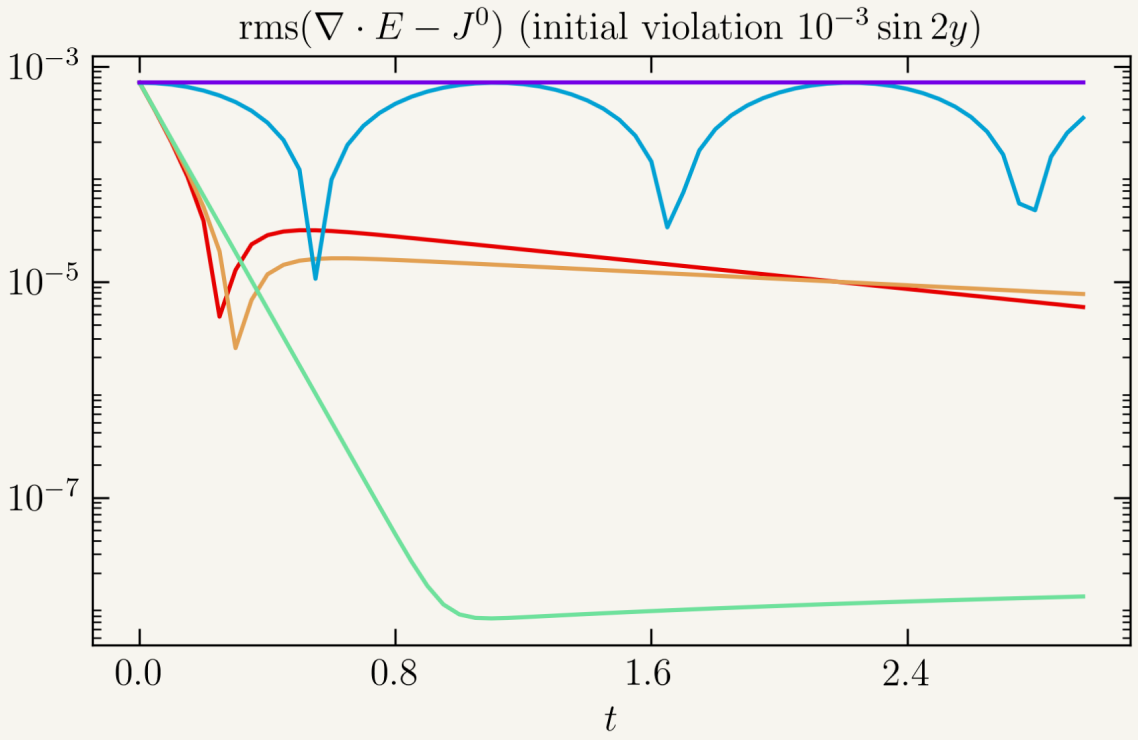
DROPPING $\partial_t E$

$$\sim 1/\sigma$$

$u_{\rm rms}$ changes by 4.6×10^{-3} at $\sigma = 1$, falling to 4.5×10^{-5} at $\sigma = 100$.

At $\sigma = 1$, E, J^0 and B all change by ~40%.

Damping cleans at a rate set by σ



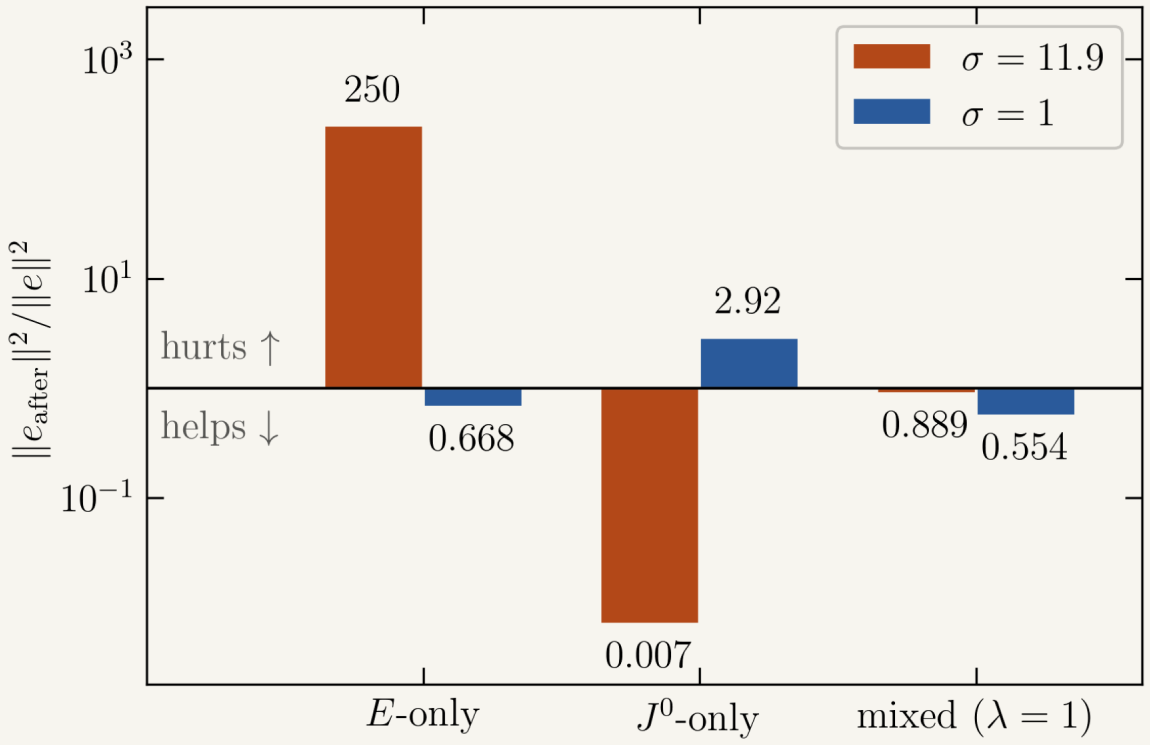
rms($\nabla \cdot E - J^0$), initial violation $10^{-3} \sin 2y$

- $w = 0$ + damping: decays to 10^{-8}
- $w = 2$ + damping: stalls ~10^{-5}
- $w = 1$ + damping: stalls ~10^{-5}
- $w = 2$, no damping: oscillates
- No enforcement: conserved

Fields themselves change $< 10^{-9}$. With Γ dropped, damping is the only enforcement left.

WHAT WE FOUND · WHERE THE ERROR LIVES

Which field carries the violation depends on σ



$$e = x_{64} - P\,x_{128}$$

Coarse run minus the spectrally restricted 128^3 reference; cleaning applied to the coarse state.

$\sigma = 11.9$: violation sits in J^0. Fix J^0; touching E costs $250\times$.

$\sigma = 1$: violation sits in $E\|$. Fix E or both; J^0-only costs $3\times$.

Ceiling: cleaning can reach only 10^{-4}–10^{-3} of the state error with enforcement, 0.10–0.17 without. It never touches u.

THE LESSON SO FAR

Satisfying the constraint is not the same as recovering the true solution.

FNO: ~97% of the error is divergence-free.

Pencil: static cleaning never touches the error in u.

PART 3

Where machine learning fits

Three ways in, and how we will know if one helps.

WHERE ML FITS

Three approaches to compare

1

Learned projection

A network replaces the Poisson solve or FFT; or corrects the state before an exact projection.

Hackathon focus

2

Learned damping

Damping terms trained end-to-end through a differentiable solver.

3

Learned parameterization

Generalize $B = \nabla \times A$ to systems where no potential is known.

Goal: corrections that recover the true solution, not just ones that satisfy the constraint.

WHERE ML FITS

What counts as better

Metrics

- Constraint residual $\|D \cdot u\| \,/\, \|Du\|$
- Error vs reference, per $|k|$ shell
- Idempotent; correction is curl-free
- Physics: magnetic spectrum $E(k)$, top 1% of $|J|$
- Cost per call; drift over 1,000 steps

Baselines

- Exact FFT projection with $s(k)$: the ceiling
- CG on $D \cdot D$, multigrid-preconditioned, tolerance sweep
- Plain compact-Laplacian multigrid
- GLM cleaning and damping

A negative result, well measured, is a valid result.

PART 4

The hackathon

Learning the divergence-free projection in MHD.

THE HACKATHON

Replace the solver's projection with one learned step

$$B^* \xrightarrow{\mathcal{F}} \hat{B}^* \xrightarrow{\delta_{ij} - k_i k_j / k^2} \hat{B} \xrightarrow{\mathcal{F}^{-1}} B$$

learn $B = \mathcal{T}_\theta[B^*]$

Can a learned operator match the projection's accuracy while costing less, and when does it stop being worth it?

> Why it is hard: the exact projection is nonlocal, its kernel decays like $1/r^3$ in 3D.

> The rule: no FFT and no call to the exact projector inside the model.

THE HACKATHON

Five levels, one model interface

Level 1 · day 1

Random fields

Synthetic spectra → exact Leray projection

Level 2 · day 2

Real MHD B

Corrupted Pencil fields → clean B

Level 2.1 · day 3

Worth it?

Cost–accuracy Pareto against solver methods

Level 3 · day 4 A

$\nabla \cdot u = f$

Constraint with a source: 3a smooth, 3b localised

Level 4 · day 4 B

Gauss's law

4b: drifted $E \to E$ with $\nabla \cdot E = J^0$

Levels unlock in order. Days 1–3 are the core path; on day 4 each team picks one advanced track.

THE HACKATHON

Fixed conventions and the model interface

- PyTorch, tensors [batch, comp, x, y, z]
- Periodic box of side 2π, no ghost zones
- D = Pencil sixth-order stencil
- Laplacian = $D \cdot D$ (null at Nyquist)
- Mean mode preserved; unit RMS per snapshot
- Data and scoring in float32; harness validated in float64

```
class Projector(torch.nn.Module):
    def forward(self, u, f=None, a=None):
        # u: [B,3,N,N,N]
        # f: source (Levels 3, 4)
        return corrected_u
```

```
python -m kit.evaluate my_team --level 2
```

THE HACKATHON · CORE PATH

Levels 1 and 2: from noise to turbulence

Level 1 · random fields

- Spectra $|k|^{-s}$, $s \in \{0, 5/3, 3\}$, made on the fly
- Baselines: exact FFT; local stencil radius 1–8; linear FNO (reference only)
- Local stencils stay at 60–70% error on smooth fields

Pass: $< 1\%$ error on white noise; beat the radius-8 stencil.

Level 2 · real MHD

- Pencil 512^3 turbulence, restricted to 32^3–128^3; $B = D \times A$
- Corruptions: gradient noise, regridding, truncation
- Train on decaying runs; forced run held out
- Diagnostics: magnetic spectrum $E(k)$, top 1% of $|J|$, mean field

Pass: $< 1\%$ error held out; residual $< 10^{-4}$.

Thresholds are starting values; final numbers are set after all baselines are scored.

Level 2.1 decides; Levels 3 and 4 generalize

2.1 · is it worth it?

- Pareto plot: error vs time per call
- Against spectral, PCG + multigrid, plain multigrid, GLM
- Small kinematic induction loop
- Break-even number of calls, training included

3 · $\nabla \cdot u = f$

$$u' = u - D\phi, \quad D \cdot D\phi = D \cdot u - f$$

- 3a: smooth sources; $f = 0$ recovers Level 2
- 3b: localised sources, $\nabla\varphi \sim 1/r^2$
- Correction must be curl-free

4b · Gauss's law

$$E' = E - D\phi, \quad D \cdot D\phi = D \cdot E - J^0$$

- Data: E and J^0 from our Pencil runs without enforcement, 32^3
- Same interface as Level 3

THE HACKATHON · SCHEDULE

Four days, a checkpoint each day

Day	Goal	Checkpoint
0, before	Working setup	Baseline on the leaderboard
1	Level 1: no-FFT projector	$< 1\%$ on white noise
2	Level 2: real MHD	Held-out checks passed
3	Level 2.1: worth it?	Pareto point, trade-off table
4	Track A (L3) or B (L4)	Report, 5-minute talk

OPEN QUESTIONS

What we do not know yet

Resolution transfer

Does a projector trained at 64^3 hold at 128^3 and beyond?

Long rollouts

Is it stable over 1,000 steps, or does its error accumulate?

Accuracy, not just the constraint

Can any correction reach the error that projection leaves behind?

SUMMARY

What we know so far

1. Constraint and accuracy are different targets: most error lives where projection cannot reach.
2. In relativistic MHD the violation's carrier depends on σ; damping toward $\nabla \cdot E$ cleans at a rate $\sim\sigma$.
3. The hackathon asks one measurable question: is a learned projection cheaper at equal accuracy?

Questions?