ArtificialGate

ArtificialGate Ltd. Research Preprint

The Direction–Execution Gap in Physics-Informed Neural Network Optimization

Barak Or
ArtificialGate Ltd., Israel
barak@artificialgate.ai

Abstract

PINN optimization is often interpreted through component-gradient geometry, yet training executes a finite, stateful update shaped by momentum, adaptive scaling, or conflict resolution. We introduce the *direction–execution gap*: a favorable local component projection followed by deterioration after the complete step. Our audit measures this event on identical frozen points before and after execution, eliminating collocation replacement; a paired-estimator analysis establishes its variance advantage. A Hessian-Lipschitz bound then converts one directional Hessian-vector product into two-sided finite-step sign margins. In 111 paired runs across four PDE families and three update rules, first-order sign accuracy was 86.39% and reversal prevalence 13.56%. The diagnostic was attached to only 80 of 6000 optimizer updates per run (1.33%), yet it exposed the failed local-progress claims and identified directional curvature as their mechanism. Directional curvature raised accuracy to 99.63% and reduced the median normalized magnitude-error ratio from the first-order baseline of 1.0 to 1.284×10^{-3}—a 99.87% reduction. A guarded-update control reduced reversals but neither improved final relative L_2 error nor offset its 3.65-fold runtime. The result is an optimizer-aware reliability audit separating directional promise, realized component progress, and solver accuracy.

Keywords. physics-informed neural networks; finite-step reliability; adaptive optimization; directional curvature; multi-objective learning.

1 The Direction–Execution Gap

Physics-informed neural networks (PINNs) approximate solutions of partial differential equations (PDEs) by jointly optimizing interior residual, boundary, initial, or observational losses [1]. Training failures have been linked to operator stiffness, gradient imbalance, ill-conditioned loss landscapes, and unreliable solver-level accuracy [2–7]. Because the component terms define a multi-objective problem, related responses include adaptive PINN loss balancing and Pareto- or projection-based gradient aggregation [8–10].

PDEs with closed-form or manufactured solutions remain useful PINN benchmarks because they provide exact error references and isolate optimization effects under controlled physics. The practical motivation is broader: PINNs combine differential constraints with sparse observations in inverse, data-assimilation, and geometrically complex settings where an analytic solution is unavailable. The canonical equations used here therefore test the optimizer mechanism before it is transferred to those harder regimes.

DCGD and ConFIG reconcile component gradients, while other work studies second-order alignment, loss-landscape curvature, residual-surface flattening, and metric-aware or quasi-Newton updates [5, 11–16]. PCGrad also connects gradient conflict to aggregate multi-task curvature across successive iterates [10]. These methods improve weighting, direction construction, or global step control. They do not test whether every frozen-set PINN component that looked

This research was conducted under ArtificialGate Ltd. The author serves as Academic Director at the Google & Reichman Tech School, and as an External Lecturer at Technion–Israel Institute of Technology and Reichman University.

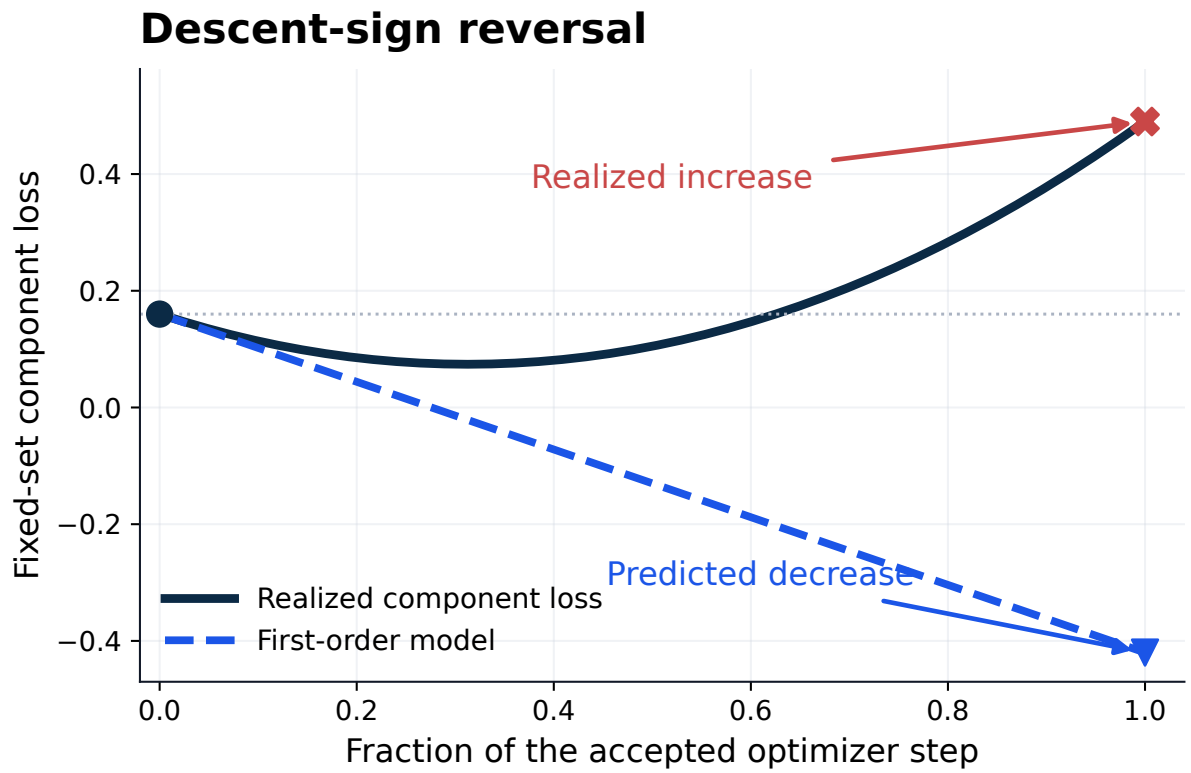

Figure 1: Direction–execution gap: a locally favorable first-order prediction can end at a higher loss after the accepted finite optimizer step.

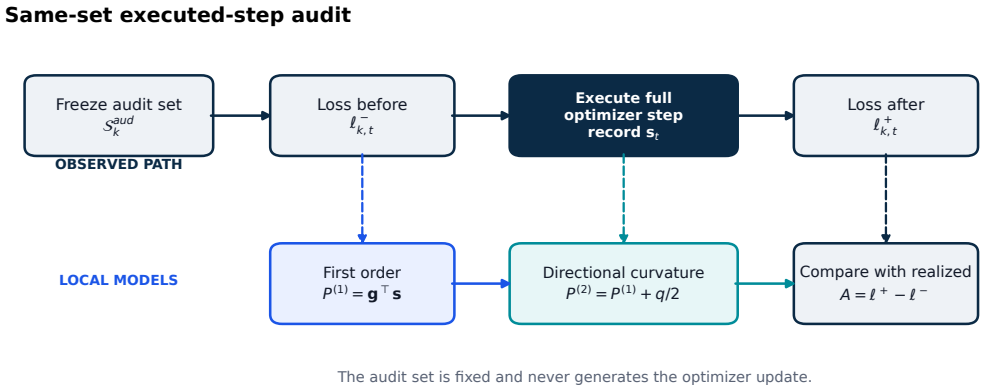

Figure 2: Same-set audit workflow. Straight data paths distinguish observed before/after losses from first- and second-order predictions; the frozen set never generates the update.

favorable at $\boldsymbol{\theta}_t$ actually improved after a stateful optimizer executed its complete step. Momentum, adaptive scaling, and conflict resolution can make that finite displacement differ materially from the nominal direction.

We call this distinction the *direction–execution gap*. If a component derivative has a favorable projection on the recorded update but the same component loss increases on the same points, a *descent-sign reversal* has occurred. Figures 1 and 2 separate the event from the measurement needed to distinguish it from collocation resampling.

This work makes three linked contributions. First, it formalizes the direction–execution gap as a component-wise event defined on the *complete* optimizer displacement, rather than on a nominal direction or update-generating batch. Second, it introduces a paired same-set audit and derives its variance advantage over independently resampled pre/post measurements. Third, it turns one directional Hessian-vector product into a two-sided finite-step sign margin and uses that margin to identify curvature as the mechanism behind the observed reversals. The contribution is the optimizer-aware combination of these established ingredients into a measurable finite-step reliability audit, supported by systematic evidence at the step lengths PINN training actually executes.

The remainder of this paper develops the audit in Section 2, specifies the experiments in Section 3, reports the results in Section 4, discusses scope in Section 5, and provides statistical details in Appendix A.

2 A Same-Set Executed-Step Audit

2.1 Component Losses and Update-Generating Batches

Let $u_{\boldsymbol{\theta}} : \Omega \subset \mathbb{R}^{d_z} \to \mathbb{R}^{d_u}$ be a PINN with parameter vector $\boldsymbol{\theta} \in \mathbb{R}^p$. Here d_z is the input dimension, including spatial and, when present, temporal coordinates; d_u is the number of predicted physical fields; and p is the number of trainable parameters. For component $k \in \{1, \dots, K\}$, the operator $\mathcal{R}_k$ evaluates a PDE, boundary, or initial-condition residual with target b_k. On a finite sample set $\mathcal{S}$, the raw component loss is

$$\mathcal{L}_k(\boldsymbol{\theta}; \mathcal{S}) = \frac{1}{|\mathcal{S}|} \sum_{z \in \mathcal{S}} \|\mathcal{R}_k[u_{\boldsymbol{\theta}}](z) - b_k(z)\|_2^2. \tag{1}$$

A sample item z can be a single coordinate or a tuple of coordinates required by a coupled boundary operator.

At optimization step t, the update-generating objective is

$$\mathcal{L}_{\mathrm{tr},t}(\boldsymbol{\theta}) = \sum_{k=1}^{K} w_{k,t} \mathcal{L}_k(\boldsymbol{\theta}; \mathcal{B}_{k,t}), \tag{2}$$

where $\mathcal{B}_{k,t}$ is the batch that generates the update and $w_{k,t} > 0$ is its component weight. The batch index is explicit because training points may change between updates, whereas a before-and-after audit must use an unchanged set.

2.2 Frozen Audit Losses and the Executed Step

Each component is assigned a fixed audit set $\mathcal{S}_k^{\mathrm{aud}}$. It is drawn once per run, is disjoint from the independent solution-evaluation set, and is never used to calculate the training update. It is therefore a diagnostic probe, not a validation set or a source of optimization data. Its purpose is to isolate parameter-induced change from collocation resampling. The losses immediately before and after update t are

$$\ell_{k,t}^{-} = \mathcal{L}_k(\boldsymbol{\theta}_t; \mathcal{S}_k^{\mathrm{aud}}), \tag{3}$$

$$\ell_{k,t}^{+} = \mathcal{L}_k(\boldsymbol{\theta}_{t+1}; \mathcal{S}_k^{\mathrm{aud}}). \tag{4}$$

The executed parameter step is

$$\boldsymbol{s}_t = \boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t, \tag{5}$$

including the complete effect of momentum, adaptive preconditioning, or conflict resolution. The realized component-loss change is

$$A_{k,t} = \ell_{k,t}^{+} - \ell_{k,t}^{-}. \tag{6}$$

Thus $A_{k,t} < 0$ denotes improvement on the audit set, whereas $A_{k,t} > 0$ denotes deterioration.

2.3 Motivation for Same-Set Pairing

Proposition 1 (paired-set isolation) Let $f_k^{-}(Z)$ and $f_k^{+}(Z)$ denote the pointwise component losses at $\boldsymbol{\theta}_t$ and $\boldsymbol{\theta}_{t+1}$. Under any unbiased randomized sampling design, write $\bar{f}_k^{\pm}(\mathcal{S})$ for the sample mean on a random set $\mathcal{S}$. Compare $\widehat{\Delta}_{\mathrm{same}} = \bar{f}_k^{+}(\mathcal{S}) - \bar{f}_k^{-}(\mathcal{S})$ with $\widehat{\Delta}_{\mathrm{ind}} = \bar{f}_k^{+}(\mathcal{S}^{+}) - \bar{f}_k^{-}(\mathcal{S}^{-})$, where $\mathcal{S}^{+}$ and $\mathcal{S}^{-}$ are independent draws from the same design. Both estimate the population component change without bias, and

$$\mathrm{Var}(\widehat{\Delta}_{\mathrm{same}}) = \mathrm{Var}(\widehat{\Delta}_{\mathrm{ind}}) - 2\,\mathrm{Cov}(\bar{f}_k^{+}(\mathcal{S}), \bar{f}_k^{-}(\mathcal{S})). \tag{7}$$

The identity is obtained by expanding the variance of the paired difference. Appendix A gives the iid specialization and explains how the randomized scrambled Sobol design is used without making the algebraic identity depend on iid sampling. Whenever the two sample means are nonnegatively correlated—the typical local-update regime—pairing has no larger variance. Once the audit set is frozen, Eqs. (3)–(6) additionally measure one fixed empirical objective exactly, so an observed sign cannot be produced by changing collocation points. Pairing is therefore a statistical isolation principle, not merely an implementation detail.

2.4 From Local Direction to Executed Step

At the pre-update parameters, define

$$\boldsymbol{g}_{k,t} = \nabla_{\boldsymbol{\theta}} \mathcal{L}_k(\boldsymbol{\theta}_t; \mathcal{S}_k^{\mathrm{aud}}), \qquad \boldsymbol{H}_{k,t} = \nabla_{\boldsymbol{\theta}}^2 \mathcal{L}_k(\boldsymbol{\theta}_t; \mathcal{S}_k^{\mathrm{aud}}). \tag{8}$$

The first-order prediction for the change caused by the executed step is

$$P_{k,t}^{(1)} = \boldsymbol{g}_{k,t}^{\top} \boldsymbol{s}_t, \tag{9}$$

A negative value predicts a decrease. The directional quadratic form and local second-order prediction are

$$q_{k,t} = \boldsymbol{s}_t^{\top} \boldsymbol{H}_{k,t} \boldsymbol{s}_t, \tag{10}$$

$$P_{k,t}^{(2)} = P_{k,t}^{(1)} + \frac{1}{2} q_{k,t}. \tag{11}$$

The product $\boldsymbol{H}_{k,t}\boldsymbol{s}_t$ is computed by differentiating the scalar $\boldsymbol{g}_{k,t}^\top \boldsymbol{s}_t$ with respect to $\boldsymbol{\theta}$, treating the recorded step vector as constant [18, 19]. This Hessian-vector product uses vectors with p entries and never forms a dense $p \times p$ Hessian.

The link to the realized change follows from Taylor's theorem with integral remainder [17]. For $\phi_{k,t}(\alpha) = \mathcal{L}_k(\boldsymbol{\theta}_t + \alpha \boldsymbol{s}_t; \mathcal{S}_k^{\text{aud}})$ and $\alpha \in [0,1]$, this one-dimensional restriction traces the loss along the accepted parameter segment. Integrating $\phi''_{k,t}(\alpha)$ twice over $[0,1]$, with $\phi'_{k,t}(0) = \boldsymbol{g}_{k,t}^\top \boldsymbol{s}_t$, gives

$$A_{k,t} = P_{k,t}^{(1)} + \int_0^1 (1-\alpha)\, \boldsymbol{s}_t^\top \nabla^2 \mathcal{L}_k(\boldsymbol{\theta}_t + \alpha \boldsymbol{s}_t; \mathcal{S}_k^{\text{aud}}) \boldsymbol{s}_t \, d\alpha. \tag{12}$$

The weight $1-\alpha$ accounts for how much curvature encountered at position α contributes to the endpoint. The first-order error is therefore accumulated directional curvature along the accepted segment. Equation (11) replaces the path-varying Hessian by its value at the start of that segment.

Proposition 2 (executed-step sign certificate) Assume that the component Hessian is locally $\rho_{k,t}$-Lipschitz along the executed segment, so $\|\nabla^2 \mathcal{L}_k(\boldsymbol{\theta}_t + \alpha \boldsymbol{s}_t; \mathcal{S}_k^{\text{aud}}) - \boldsymbol{H}_{k,t}\|_2 \le \rho_{k,t} \alpha \|\boldsymbol{s}_t\|_2$ for $\alpha \in [0,1]$. Equation (12) then gives the deterministic bound

$$\left| A_{k,t} - P_{k,t}^{(2)} \right| \le r_{k,t} \triangleq \frac{\rho_{k,t}}{6} \|\boldsymbol{s}_t\|_2^3. \tag{13}$$

Indeed, subtracting Eq. (11) from Eq. (12) and integrating $\alpha(1-\alpha)$ yields the factor $1/6$. Define the two-sided certificate margins

$$M_{k,t}^{\downarrow} = -(P_{k,t}^{(2)} + r_{k,t}), \qquad M_{k,t}^{\uparrow} = P_{k,t}^{(2)} - r_{k,t}. \tag{14}$$

A positive $M_{k,t}^{\downarrow}$ certifies realized descent, while a positive $M_{k,t}^{\uparrow}$ certifies realized increase; otherwise the local certificate is unresolved. Equivalently, $|P_{k,t}^{(2)}| > r_{k,t}$ certifies its sign, while $|P_{k,t}^{(1)}| > |q_{k,t}|/2 + r_{k,t}$ certifies the first-order sign.

Corollary (quadratic reversal threshold) In the locally quadratic case $r_{k,t} = 0$, a predicted descent with $P_{k,t}^{(1)} < 0$ and $q_{k,t} > 0$ is reversed exactly when $q_{k,t}/(2|P_{k,t}^{(1)}|) > 1$. Thus the audit, certificate margin, and curvature ratio answer different questions: what happened, what is guaranteed locally, and why the first-order sign can fail. No global smoothness constant is required for the empirical audit itself.

2.5 Audit Measures

To compare components with different scales, let $c_{k,t} = |\ell_{k,t}^-| + \delta$ and define $\widehat{A}_{k,t} = A_{k,t}/c_{k,t}$ and $\widehat{P}_{k,t}^{(j)} = P_{k,t}^{(j)}/c_{k,t}$. The floor $\delta > 0$ and a sign tolerance $\varepsilon > 0$ exclude roundoff-level changes.

A first-order descent-sign reversal is

$$\widehat{P}_{k,t}^{(1)} < -\varepsilon, \qquad \widehat{A}_{k,t} > \varepsilon. \tag{15}$$

We distinguish two frequencies. *Prevalence* is the fraction of all audited component events that satisfy Eq. (15). *Conditional risk* uses only events for which first order predicted a decrease in the denominator. The distinction matters because an optimizer can predict descent more or less often without changing the reliability of those predictions.

For model $j \in \{1,2\}$, let $\mathcal{D}^{(j)}$ contain events for which both $|\widehat{A}_{k,t}| > \varepsilon$ and $|\widehat{P}_{k,t}^{(j)}| > \varepsilon$. The run-level sign-prediction accuracy is

$$\text{Acc}^{(j)} = \frac{1}{|\mathcal{D}^{(j)}|} \sum_{(k,t)\in\mathcal{D}^{(j)}} \mathbb{I}\left[\text{sgn}(\widehat{P}_{k,t}^{(j)}) = \text{sgn}(\widehat{A}_{k,t})\right]. \tag{16}$$

The normalized magnitude error and curvature-to-linear ratio are

$$E_{k,t}^{(j)} = |\widehat{P}_{k,t}^{(j)} - \widehat{A}_{k,t}|, \tag{17}$$

$$\chi_{k,t} = \frac{|q_{k,t}|/2}{|P_{k,t}^{(1)}| + \delta}. \tag{18}$$

Values $\chi_{k,t} \ll 1$ indicate linear dominance; values near or above one indicate that the quadratic correction can match or overturn the linear term. Because $\chi_{k,t}$ is constructed from the same local terms as Eq. (11), it is an explanatory finite-step scale rather than an independently calibrated failure probability.

2.6 Audit Procedure and Cost

Unlike supervisory training schemes that use secondary probes to accept or roll back optimizer proposals [25], Algorithm 1 observes an update but does not modify it. If an ordinary training step costs C_{tr}, one audit checkpoint adds before-and-after component evaluations, one gradient calculation per component, and one Hessian-vector product per component. For K components and D audit checkpoints over T training steps, the cost is approximately

$$\mathcal{O}(TC_{\text{tr}} + DK(C_{\text{eval}} + C_{\text{grad}} + C_{\text{HVP}})). \tag{19}$$

Additional storage is $\mathcal{O}(p)$ rather than $\mathcal{O}(p^2)$. In the primary runs, $D = 80$ and $T = 6000$, so the audit sampled only 1.33% of optimizer steps.

Algorithm 1 Optimizer-aware audit of an accepted PINN update

Require: $\boldsymbol{\theta}_t$; batches $\{\mathcal{B}_{k,t}\}$; fixed audit sets $\{\mathcal{S}_k^{\text{aud}}\}$; optimizer state

1: Record $\ell_{k,t}^-$ on each audit set using Eq. (3).
2: Execute one complete optimizer step and record $\boldsymbol{s}_t$ using Eq. (5).
3: Re-evaluate the same sets to obtain $A_{k,t}$ using Eqs. (4)-(6).
4: Restore $\boldsymbol{\theta}_t$ temporarily; compute $P_{k,t}^{(1)}$ and the Hessian-vector product using Eqs. (9)-(11).
5: Classify the sign outcomes using Eqs. (15)-(16).
6: Restore $\boldsymbol{\theta}_{t+1}$ and continue training.

Table 1: PDE settings and sample counts. Counts are interior/boundary/initial; "–" indicates no initial-condition term.

PDE	Two settings	Train	Audit	Eval.
Poisson 2-D	$(m, n) = (1, 1), (4, 5)$	768/256/–	384/128/–	16k
Burgers	$\nu = 0.05/\pi, 0.01/\pi$	768/160/160	384/80/80	16k
Allen-Cahn	$\epsilon = 10^{-3}, 10^{-4}$	768/160/160	384/80/80	16k
Helmholtz	$(m, n) = (1, 2), (5, 7), \kappa = 10$	1024/256/–	512/128/–	16k

3 Experimental Protocol

The primary grid spans Poisson 2-D, Burgers, Allen-Cahn, and Helmholtz equations, each with two predefined settings that vary frequency, viscosity, or diffusivity. For Helmholtz, the wave number is set to $\kappa = 10$. Burgers and Allen-Cahn are evaluated against independent numerical reference solvers whose coarse-to-fine differences remain below 1%; Poisson and Helmholtz use manufactured solutions. The 1% value is a conservative design tolerance chosen so that reference-grid uncertainty is small relative to the solver differences being interpreted; it is not proposed as a universal PINN threshold and can be tightened when the application requires it.

All runs use a five-hidden-layer multilayer perceptron with 64 tanh units per layer and Xavier-normal initialization [20]. Inputs are scaled coordinatewise to $[-1, 1]$. Scrambled Sobol sequences generate deterministic, paired low-discrepancy streams for training, audit, and solution-evaluation sets [21, 22]. The architecture is fixed to isolate update geometry. Within every PDE-setting-seed block, methods share initialization, sampling streams, audit sets, and evaluation points.

The primary grid uses Adam, plain SGD, and ConFIG followed by Adam. Adam uses $(\beta_1, \beta_2) = (0.9, 0.999)$ and $\epsilon_A = 10^{-8}$ [23]; SGD uses zero momentum [24]. Raw component weights in Eq. (2) are one for Adam and SGD. ConFIG constructs a conflict-aware direction from the same unweighted component gradients before Adam applies its adaptive transform. Learning rates are declared per PDE and range from 5×10^{-5} to 10^{-3}. Each run performs 6000 FP64 updates with 80 audit checkpoints and no gradient clipping.

The controls isolate mechanisms: Adam tests stateful adaptive execution; zero-momentum SGD, unpreconditioned steps; ConFIG+Adam, conflict resolution followed by adaptive transformation; and half-rate Adam, simple shrinkage. A prespecified sweep adds 45 matched AdamW runs on hard Allen-Cahn, Burgers, and Helmholtz at weight decays 10^{-6}, 10^{-4}, and 10^{-2}. Together they cover five update mechanisms with matched initialization and samples.

For FP64 measurements, $\delta = 10^{-14}$ and $\varepsilon = 10^{-9}$. Statistical comparisons use the run, not the checkpoint or component event, as the inferential unit. First- and second-order outcomes are paired by PDE, setting, optimizer, and seed and assessed with the Wilcoxon signed-rank test [26]; this nonparametric test asks whether the paired run-level differences are centered at zero without assuming Gaussian differences. Holm's procedure controls family-wise error over the primary endpoints [27]. The estimand is the median paired change over this predefined experimental grid. The resulting p-values are not interpreted as population-level evidence over unseen PDE families or architectures.

4 Measuring the Gap

The primary analysis contains 111 completed runs and 24,960 fixed-set component events. Across the primary grid and auxiliary controls, 186 runs completed in approximately 6.0 aggregate GPU-hours.

For the results below, prevalence divides reversals by all audited events, whereas conditional risk divides reversals only by events for which first order predicted descent. Sign accuracy uses events whose realized and predicted normalized changes exceed the numerical tolerance, and normalized magnitude error is the absolute difference in Eq. (17).

4.1 The Gap Persists Across PDEs and Update Rules

Figure 3 shows that the gap is neither isolated to one PDE nor absent under ConFIG+Adam. Under Adam, reversals accounted for 10.1% of audited component events in Allen-Cahn, 10.8% in Burgers, 19.3% in Helmholtz, and 19.9% in Poisson 2-D for the challenging settings. ConFIG+Adam produced corresponding rates of 4.3%, 11.5%, 25.0%, and 25.1%. This does not contradict ConFIG's favorable projections for its constructed training-batch direction: Adam subsequently transforms that direction, and the audit evaluates the resulting finite step on frozen objectives. The distinction is between direction construction and executed-step progress.

Across all completed primary runs, including both settings and stable SGD runs, first-order sign-prediction accuracy averaged 86.39%. Pooled reversal prevalence was 13.56%, and conditional risk among predicted decreases was

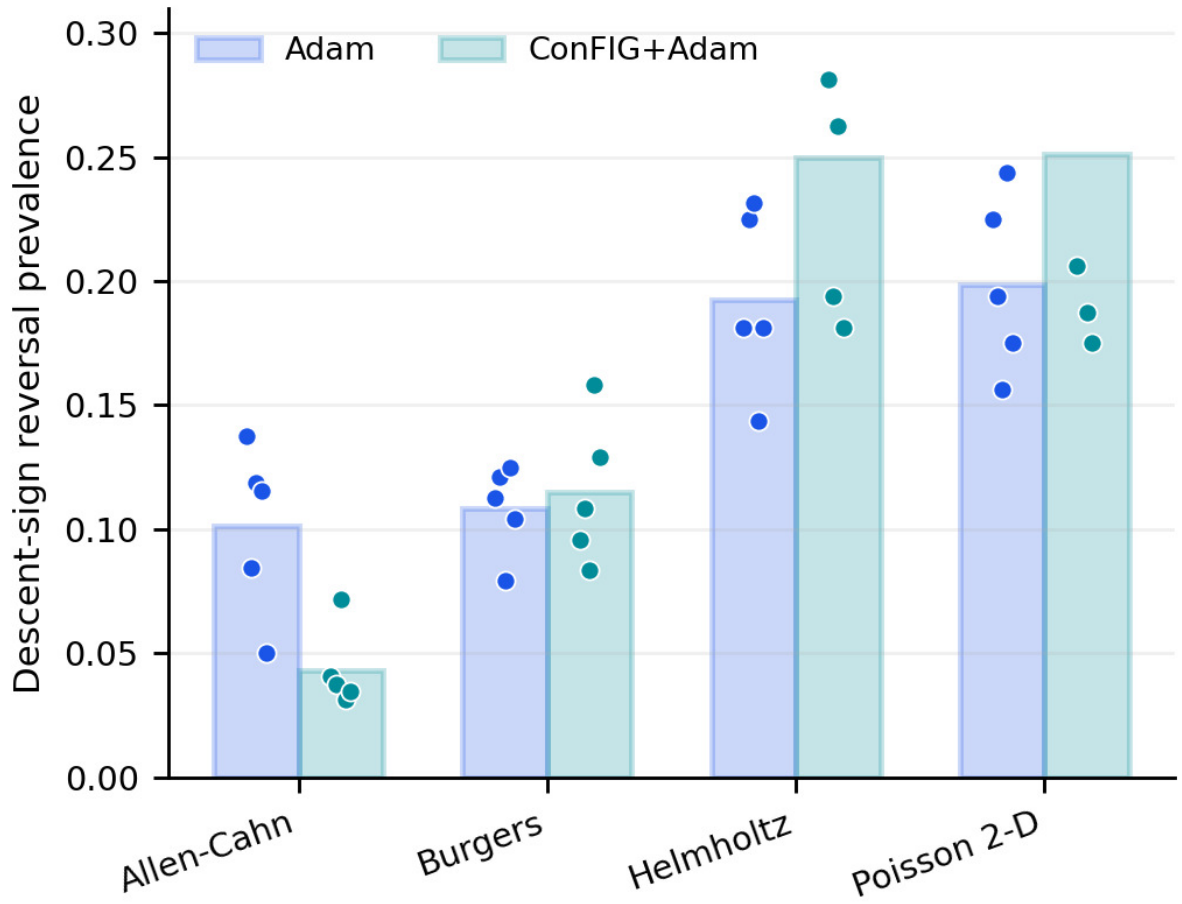

Figure 3: Direction–execution gap in the challenging settings, measured by first-order descent-sign reversal prevalence. Bars show means; points show paired seeds.

17.85%. Plain SGD generally made smaller updates and exhibited fewer reversals. The pattern is consistent with the finite-step interpretation: a favorable directional derivative guarantees descent only within a sufficiently small neighborhood.

4.2 Directional Curvature Explains the Gap

The curvature-corrected prediction increases mean run-level sign accuracy from 86.39% to 99.63% (Fig. 4). The median within-run gain is 11.88 percentage points. Accuracy improves in 86 runs, is unchanged in 25, and declines in none. The paired run-level comparison remains significant after multiplicity correction ($p = 7.96 \times 10^{-16}$). Second-order reversal prevalence falls to 0.128%, with conditional risk 0.245%. These outcomes span all 23 nonempty PDE–setting–method cells; they are not produced by one PDE or optimizer stratum. Figure 6 reports the corresponding accuracy by PDE, difficulty, and update rule.

Magnitude recovery is still more consistent. Taking the first-order normalized magnitude error as 1.0, the median second-to-first-order error ratio is 1.284×10^{-3}; equivalently, curvature correction removes 99.87% of that baseline error. Every one of the 111 completed primary runs has lower median magnitude error under the curvature-corrected model ($p = 5.99 \times 10^{-20}$). Figure 5 shows why: reversal frequency rises sharply when the quadratic contribution becomes comparable with the linear term, precisely where the second-order correction can cancel or overturn the first-order sign.

4.3 Local Audit Reliability Versus Solver Accuracy

To test actionability, an auxiliary guard scales an Adam proposal below the positive local quadratic root only when a separate frozen guard set predicts a reversal. A fixed 20% root margin and lower scale bound of $1/16$ are used across the challenging Burgers and Allen-Cahn settings. Five paired seeds and two controls—standard Adam and half-rate Adam—are run for 6000 FP64 updates. The guard set is distinct from training, audit, and solution-evaluation sets.

Across the 10 paired runs, the guard reduced median reversal prevalence from 11.61% under Adam and 8.59% under half-rate Adam to 3.44%. Median conditional risk fell from 18.83% and 14.29%, respectively, to 5.62%. The reductions were directionally consistent in all paired runs and remained significant after Holm correction ($p = 0.0156$ for each reversal comparison). Final relative L_2 error, however, did not differ detectably from Adam ($p_{\text{Holm}} = 1.000$) or the half-rate control ($p_{\text{Holm}} = 1.000$), and median runtime increased by a factor of 3.65. On Allen-Cahn, reversal prevalence fell from 10.13% to 0.88% without rescuing the solution, confirming that local sign control is insufficient

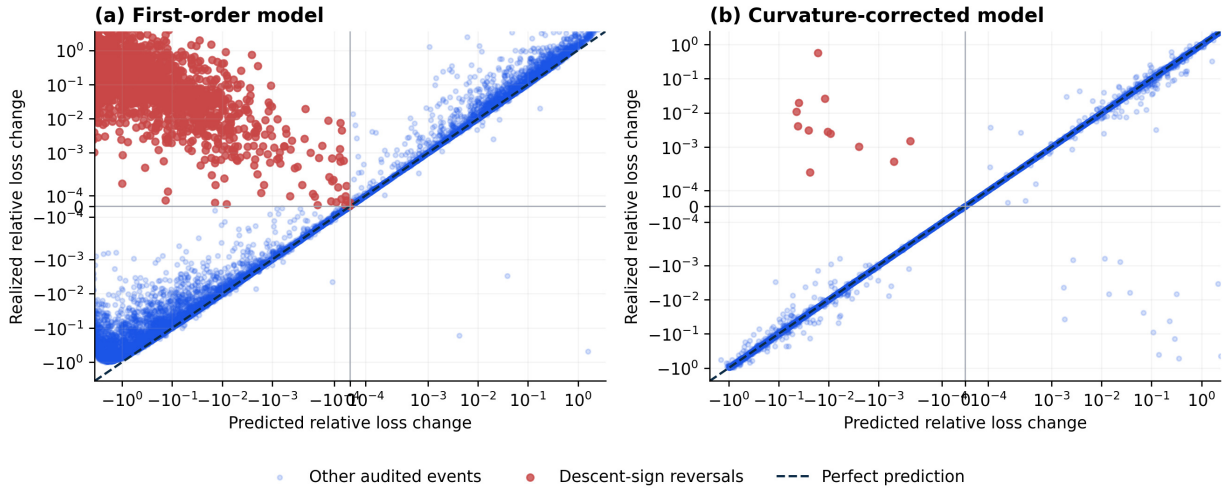

Figure 4: Predicted versus realized same-set changes. First order exposes the direction–execution gap; adding local directional curvature concentrates events near the identity line.

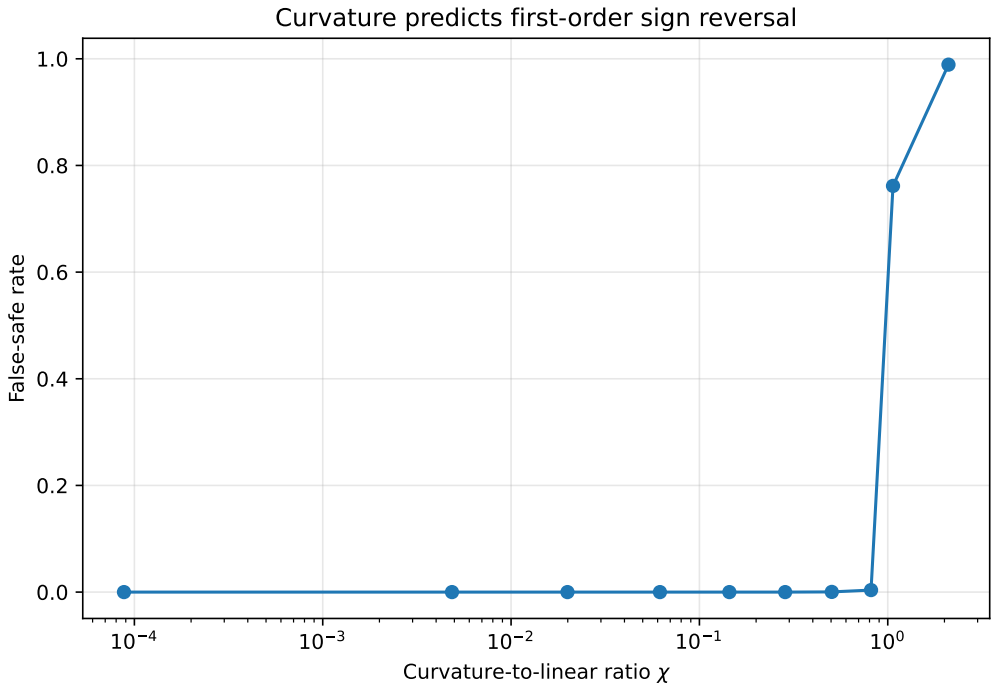

Figure 5: False-safe rate versus the curvature-to-linear ratio. Risk rises when the quadratic contribution becomes comparable with the linear term.

for PDE accuracy.

Solution quality is stratified. In hard Burgers, median final relative L_2 errors were 0.336, 0.103, and 0.147 for Adam, half-rate Adam, and the guard, while reversals remained measurable. In hard Allen-Cahn, the corresponding medians were 0.993, 0.996, and 0.997, so that family serves only as a stress test. The gap is therefore not confined to a completely failed solver, and local reliability is not solution accuracy.

In the 45-run robustness sweep, weight decay did not consistently improve reliability or final error across three PDEs and three decay strengths; exact adaptive/decay step decomposition passed the supplied unit test. This broadens the executed-step mechanism without treating AdamW as a tuned baseline.

5 Meaning and Scope of the Gap

The Taylor identity in Eq. (12) is standard; the new object is the reliability of the *stateful optimizer map* after it has converted local gradient information into the step actually executed. Two separations make that object measurable. Proposition 1 removes pre/post collocation replacement from the estimator, and Proposition 2 separates a favorable derivative from a certified finite-step sign. The nontrivial empirical result is then the frequency and scale at which practical PINN steps cross that boundary, not the generic fact that a second-order approximation can be more accurate.

The curvature-to-linear ratio in Eq. (18) gives a direct scale for the observed failure. When the quadratic term is small, first-order signs are reliable; when it approaches the linear term, reversals become common. Gradient alignment

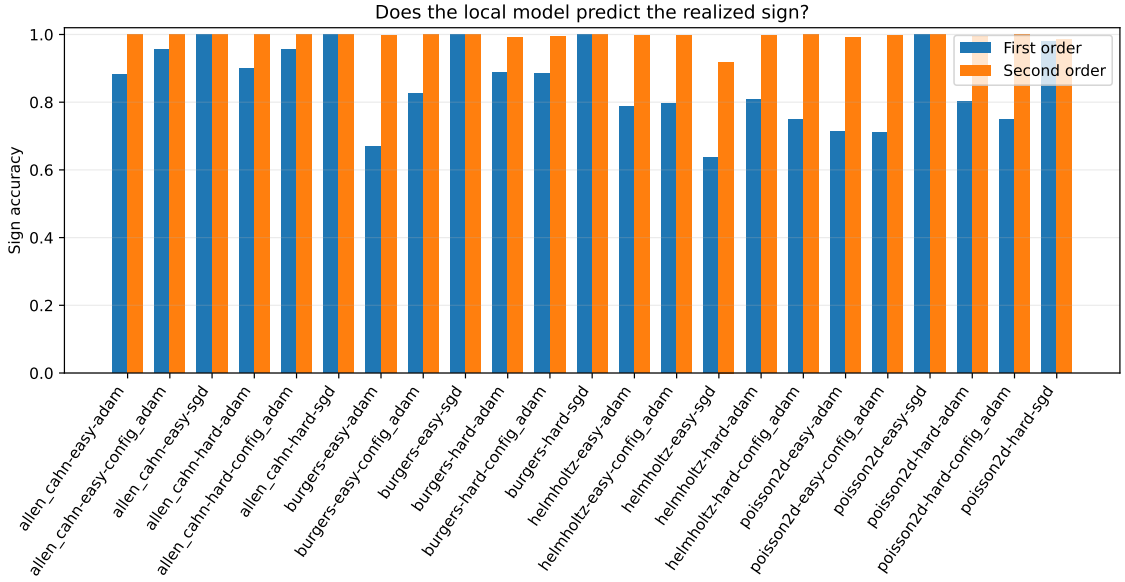

Figure 6: First- and second-order sign accuracy across PDE, difficulty, and update-rule regimes.

and conflict resolution remain useful properties of a proposed direction, but they do not by themselves certify the full adaptive step. In particular, the ConFIG results should be read as a post-optimizer finite-step audit, not as a test of ConFIG's training-batch projection theorem.

The PINN-specific audit needs only differentiable component losses and flags local-progress claims unsupported by execution; here the target is component-wise diagnosis during ordinary training. The guard reduced reversals at 3.65 times Adam's runtime but did not improve final error. Local reliability and solver accuracy are therefore distinct, as in physical-admissibility gates where passing a monitored envelope does not certify task success [7].

Practical use. At a sparse checkpoint, freeze a diagnostic set, record component losses, execute the ordinary step, and reevaluate the same losses. The dot product $\boldsymbol{g}_{k,t}^{\top}\boldsymbol{s}_t$ tests the linear claim; one Hessian-vector product supplies the curvature explanation and quadratic margin. A reversal warns about the step, not the final solution. This three-question audit—prediction, outcome, and explanation—mirrors runtime-assurance boundaries between learned proposals and consequential execution [28].

5.1 Limitations

Evidence covers one tanh MLP, four low-dimensional forward PDEs, five paired seeds, and fixed sampled audit sets; generality to other architectures, inverse problems, and neural operators remains untested. The experimental grid is not a PDE population, and the curvature ratio is not a calibrated failure probability. Guard hyperparameter sensitivity and independent validation remain future work.

6 Conclusion

This work defines the direction–execution gap for the complete optimizer step and combines a realized-displacement definition, a paired estimator with explicit variance advantage, and two-sided finite-step margins. In the tested grid, the gap is frequent, persists after conflict-aware direction construction followed by Adam, and is explained by directional curvature. The guard further separates local reliability from final PDE accuracy. The actionable message is narrow: audit the path actually taken before calling a favorable local direction realized component progress.

A Paired-Set Variance and Statistical Inference

For iid points, write the pre- and post-step pointwise losses as $X = f_k^-(Z)$ and $Y = f_k^+(Z)$. With n shared points, the paired mean difference has variance $[\mathrm{Var}(X) + \mathrm{Var}(Y) - 2\,\mathrm{Cov}(X,Y)]/n$; independent pre/post sets omit the covariance term. This gives Eq. (7) directly. A randomized scrambled Sobol design changes the sampling variance but not this algebraic same-design identity. Once the points are frozen, the audit compares two deterministic empirical losses on exactly those points.

For each primary endpoint, the signed-rank analysis first forms one paired difference per matched PDE–setting–optimizer–seed run. Zero differences are omitted, absolute nonzero differences are ranked with average ranks for ties, and the signed ranks are summed. The two-sided Wilcoxon test evaluates whether the paired-difference distribution is centered at zero without requiring Gaussian differences; under the usual symmetry condition, this center is the median [26]. For m primary endpoints, Holm's procedure orders their p-values and compares the jth ordered value with $\alpha/(m-j+1)$, preserving family-wise error control [27].

Acknowledgment

OpenAI Codex [29] assisted language and structural editing. The authors verified all claims, data, and final wording and take responsibility for the manuscript.

References

[1] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, 2019.

[2] A. Krishnapriyan, A. Gholami, S. Zhe, R. Kirby, and M. W. Mahoney, "Characterizing possible failure modes in physics-informed neural networks," in *Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 26548–26560.

[3] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient flow pathologies in physics-informed neural networks," *SIAM J. Sci. Comput.*, vol. 43, no. 5, pp. A3055–A3081, 2021.

[4] S. Wang, X. Yu, and P. Perdikaris, "When and why PINNs fail to train: A neural tangent kernel perspective," *J. Comput. Phys.*, vol. 449, Art. no. 110768, 2022.

[5] P. Rathore, W. Lei, Z. Frangella, L. Lu, and M. Udell, "Challenges in training PINNs: A loss landscape perspective," in *Proc. 41st Int. Conf. Mach. Learn.*, 2024, pp. 42159–42191.

[6] T. Służalec, D. Wójcik, C. Uriarte, M. Łoś, A. Paszyńska, and M. Paszyński, "Reliable physics-informed neural networks for Navier–Stokes simulations: Can we trust AI-generated numerical simulations?" *J. Comput. Sci.*, vol. 95, Art. no. 102817, 2026.

[7] B. Or, "Can predicted dynamics exist in the physical world?" arXiv:2606.00089, 2026.

[8] R. Bischof and M. A. Kraus, "Multi-objective loss balancing for physics-informed deep learning," *Comput. Methods Appl. Mech. Eng.*, vol. 439, Art. no. 117914, 2025.

[9] O. Sener and V. Koltun, "Multi-task learning as multi-objective optimization," in *Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 527–538.

[10] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn, "Gradient surgery for multi-task learning," in *Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 5824–5836.

[11] Y. Hwang and D.-Y. Lim, "Dual cone gradient descent for training physics-informed neural networks," arXiv:2409.18426, 2024.

[12] Q. Liu, M. Chu, and N. Thuerey, "ConFIG: Towards conflict-free training of physics-informed neural networks," in *Proc. Int. Conf. Learn. Representations*, 2025.

[13] S. Wang, A. Bhartari, B. Li, and P. Perdikaris, "Gradient alignment in physics-informed neural networks: A second-order optimization perspective," in *Adv. Neural Inf. Process. Syst.*, vol. 38, 2025.

[14] F. Liu, R. Saluja, S. Kwak, R. Wang, R. Deng, H. Kim, J. C. Paetzold, and M. R. Sabuncu, "MAdam: Metric-aware multi-objective Adam," arXiv:2606.03904, 2026.

[15] R. Z. Moayedi, M. Abbaszadeh, and M. Dehghan, "CuPINN: Optimizing PINNs through curvature minimization and residual landscape flattening," *Comput. Methods Appl. Mech. Eng.*, vol. 445, Art. no. 118180, 2025.

[16] E. Kiyani, K. Shukla, J. F. Urbán, J. Darbon, and G. E. Karniadakis, "Optimizing the optimizer for physics-informed neural networks and Kolmogorov–Arnold networks," *Comput. Methods Appl. Mech. Eng.*, vol. 446, Art. no. 118308, 2025.

[17] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.

[18] B. A. Pearlmutter, "Fast exact multiplication by the Hessian," *Neural Comput.*, vol. 6, no. 1, pp. 147–160, 1994.

[19] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: A survey," *J. Mach. Learn. Res.*, vol. 18, no. 153, pp. 1–43, 2018.

[20] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[21] I. M. Sobol', "On the distribution of points in a cube and the approximate evaluation of integrals," *USSR Comput. Math. Math. Phys.*, vol. 7, no. 4, pp. 86–112, 1967.

[22] A. B. Owen, “Scrambling Sobol’ and Niederreiter–Xing points,” *J. Complexity*, vol. 14, no. 4, pp. 466–489, 1998.

[23] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. Int. Conf. Learn. Representations*, 2015.

[24] L. Bottou, F. E. Curtis, and J. Nocedal, “Optimization methods for large-scale machine learning,” *SIAM Rev.*, vol. 60, no. 2, pp. 223–311, 2018.

[25] B. Or, “Automatic stability and recovery for neural network training,” arXiv:2601.17483, 2026.

[26] F. Wilcoxon, “Individual comparisons by ranking methods,” *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, 1945.

[27] S. Holm, “A simple sequentially rejective multiple test procedure,” *Scand. J. Stat.*, vol. 6, no. 2, pp. 65–70, 1979.

[28] B. Or, “Silent failures in physical AI: A literature review of runtime action authorization for autonomous systems,” arXiv:2606.00090, 2026.

[29] OpenAI, *Codex*, software system, 2026.