DIAGNOSING HOW OPTIMISERS SHAPE LEARNING THROUGH OPERATOR PROJECTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Loss and accuracy track progress, but do not distinguish what an optimiser can change from the direction it chooses or the structure that changes during learning. We introduce an operator-projection framework for bias-free piecewise-linear networks. At each training state, it separates locally attainable operator change, steering within that space, and parameter motion invisible to the sampled operators to first order. Deep linear and MNIST-1D experiments use this reference to examine existing optimisers and a reference-following control. SGD exhibits low alignment despite nearly complete estimated reachability. Increasing reference alignment does not monotonically improve test accuracy, and the control reduces input-dependent gating even when activation slopes remain nonzero. Most strikingly, its pronounced decline in raw operator diversity after fitting nearly disappears after removing the common-logit component that softmax ignores. Across three activations, this component accounts for 86–98% of the squared net operator displacement between the measured snapshots. Finite-step checks further distinguish local steering from realised operator change when gates vary. The framework provides a diagnostic instrument for interpreting optimiser behaviour and the evolution of the learned function, rather than a proposed optimisation algorithm.

1 INTRODUCTION

Two parameter updates can produce the same first-order decrease in training loss while changing different parts of a network's function. Conversely, they can produce the same local functional change while moving the parameters in different directions. A further ambiguity arises when interpreting the resulting structure: an operator can appear to simplify because a component irrelevant to prediction has changed. Understanding an optimiser therefore requires more than a loss curve or a structural statistic. It requires separating the change the current network can realise, the change selected by the update, and the meaning of the structure measured along its trajectory.

For deep linear networks, this conversion can be analysed through the end-to-end matrix. Gradient descent on the factors induces a depth-dependent transformation of the end-to-end gradient (Saxe et al., 2014; Arora et al., 2018; 2019). Bias-free piecewise-linear networks retain a related description: for each input, the network computes $f_W(x) = P_W(x)x$, where $P_W(x)$ is its input Jacobian almost everywhere. The collection of these local operators describes both how the network responds to input perturbations and how its linear action varies across inputs. Following their changes offers a common setting in which to examine SGD, adaptive methods and matrix-preconditioned updates.

The difficulty is that the operators cannot move independently. They share the same weights, and their attainable changes depend on the current gates and factors. An update may therefore fail to follow an operator-gradient target because part of that target is locally inaccessible, because the optimiser favours a different attainable direction, or both. Low alignment alone cannot identify which explanation applies. The parameter update adds another ambiguity: some of its motion may leave all sampled operators unchanged to first order. Treating these effects as one undifferentiated "bias" obscures how an optimiser acts.

We address this problem with an *operator-projection diagnosis*. We linearise the map from parameters to sampled operators, project an operator-gradient target onto its range, and use the minimum-norm lift of that projection as a reference. This separates local reachability, steering within the reachable space, and locally invisible parameter motion. A worked example shows why the same first-order

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

loss decrease can admit different diagnoses. The reference gives a precise way to read an update at a training state; a reference-following control then tests how following this local direction relates to the function learned over time.

The experiments develop three connected findings. First, SGD can have low operator alignment at states with nearly complete estimated reachability: the diagnosis distinguishes steering from an inaccessible target. Second, increasing the control's reference alignment does not monotonically improve test accuracy, while its gates become less input-dependent even with nonzero activation slopes. These observations separate local reference following from the structure and predictive performance reached during training. Third, the control's apparent post-fitting compression changes meaning after class-centring: the pronounced decline in raw operator diversity nearly disappears once the common-logit component, which softmax ignores, is removed. This component accounts for most of the squared net operator displacement between the measured snapshots.

Exact optimiser-state replay, finite-step checks, and variations in architecture and training history qualify these findings. Each comparison describes an update rule at the states it visits. Together, the results show why interpreting learning requires distinguishing local update geometry, finite motion and prediction-relevant structure.

2 From parameter updates to functional changes

2.1 The operator target

We consider a bias-free chain $W = (W_1, \dots, W_L)$ with piecewise-linear, positively homogeneous activations and a linear output. At an input $x \in \mathbb{R}^d$, its operator is

$$P_W(x) = W_L D_{L-1}(x) W_{L-1} \cdots D_1(x) W_1 \in \mathbb{R}^{C \times d}, \tag{1}$$

where $D_\ell(x)$ is the activation gate. Thus $f_W(x) = P_W(x)x$, and $P_W(x)$ equals the input Jacobian away from activation boundaries. Linear networks have $D_\ell = I$; ReLU and leaky ReLU have diagonal gates, with slopes in $\{0, 1\}$ and $\{0.1, 1\}$ in our experiments. CReLU concatenates positive and negative rectified responses (Shang et al., 2016), so its gate is rectangular and doubles the feature dimension. We keep pre- and post-activation dimensions distinct; Appendix A gives the shapes and the convention at zero pre-activations.

For a batch $\{(x_i, y_i)\}_{i=1}^B$, write $\mathcal{L} = B^{-1} \sum_i \ell(f_W(x_i), y_i)$. We equip sampled operator collections with the average Frobenius inner product,

$$\langle H, K \rangle_{\mathcal{H}} = \frac{1}{B} \sum_i \operatorname{tr}(H_i^\top K_i), \tag{2}$$

and parameter increments with the sum of layerwise Frobenius inner products, denoted $\langle \cdot, \cdot \rangle_{\mathcal{W}}$. If $g_i = \nabla_{f_i} \ell(f_i, y_i)$, the loss gradient with respect to independently varying sampled operators is $G_i = g_i x_i^\top$. We take the corresponding gradient step as the target:

$$b = (b_i)_{i=1}^B, \qquad b_i = -\eta G_i, \qquad \eta > 0. \tag{3}$$

For half-squared loss, $g_i = f_i - y_i$; for cross-entropy, $g_i = \operatorname{softmax}(f_i) - e_{y_i}$. Input rescaling or a different metric would change this request.

2.2 What the current network can realise

Let $A_{\ell,i}$ and $B_{\ell,i}$ be the products after and before W_ℓ, including the appropriate gates, so that $P_W(x_i) = A_{\ell,i} W_\ell B_{\ell,i}$. At fixed gates, the local parameter-to-operator map and its adjoint are

$$(Mu)_i = \sum_{\ell=1}^L A_{\ell,i} u_\ell B_{\ell,i}, \tag{4}$$

$$(M^* H)_\ell = \frac{1}{B} \sum_i A_{\ell,i}^\top H_i B_{\ell,i}^\top. \tag{5}$$

In particular, $\nabla_W \mathcal{L} = M^* G$. When all sampled pre-activations are nonzero, a sufficiently small neighbourhood preserves their signs and $P_{W+u} - P_W = Mu + O(\|u\|_{\mathcal{W}}^2)$. A gate crossing can make the operator jump, even though the output remains continuous. The diagnosis therefore concerns the local derivative; finite-step agreement is a separate empirical check.

The range of M contains the jointly attainable operator changes. For example, inputs sharing a complete gate pattern have identical operators and local step maps, so the network cannot independently fulfil different operator requests on those inputs. Width or a large parameter count alone does not remove this coupling. Let Π be orthogonal projection onto $\mathrm{range}(M)$. We define

$$r = \Pi b, \qquad u^{\mathrm{ref}} = M^+ b = \operatorname*{arg\,min}_{u:\, Mu = r} \|u\|_{\mathcal{W}}. \tag{6}$$

The projected target r is the closest locally attainable request; u^{ref} selects its minimum-norm parameter representative.

For a deep linear chain, every input shares the same operator. With $\bar{b} = B^{-1} \sum_i b_i$ and the single-operator map M_0, the reference has the exact linearised expression

$$u^{\mathrm{ref}} = M_0^* T^+ \bar{b}, \qquad T(H) = M_0 M_0^* H = \sum_\ell A_\ell A_\ell^\top H B_\ell^\top B_\ell. \tag{7}$$

Appendix B derives this reduction and its rank conditions. In particular, the first-order reference is zero at all-zero weights of a linear chain of depth greater than one.

3 Reading an optimiser through the diagnosis

3.1 Reachability, steering and parameter motion

For an optimiser's candidate increment u, define

$$\alpha = \frac{\|r\|_{\mathcal{H}}}{\|b\|_{\mathcal{H}}}, \qquad c_{\mathrm{op}} = \cos_{\mathcal{H}}(Mu, r),$$
$$c_{\mathrm{w}} = \cos_{\mathcal{W}}(u, u^{\mathrm{ref}}), \qquad q = \frac{\|(I - M^+ M)u\|_{\mathcal{W}}}{\|u\|_{\mathcal{W}}}. \tag{8}$$

Reachability α asks how much of the requested change is attainable at this state. Operator alignment c_{op} asks how closely the optimiser follows that attainable direction. The kernel fraction q measures parameter motion invisible to the sampled operators to first order. Weight alignment c_{w} compares the full parameter route with the minimum-norm route.

Proposition 1 (Separating constraint from steering)**.** *For any increment u,*

$$\|Mu - b\|_{\mathcal{H}}^2 = \|(I - \Pi)b\|_{\mathcal{H}}^2 + \|Mu - r\|_{\mathcal{H}}^2. \tag{9}$$

When the vectors defining the cosines are nonzero,

$$\cos_{\mathcal{H}}(Mu, b) = \alpha c_{\mathrm{op}}. \tag{10}$$

The first error term comes from an inaccessible target component; the second comes from the candidate update's choice within the reachable space. The same distinction appears in the instantaneous loss change:

$$D\mathcal{L}(W)[u] = -\frac{\|b\|_{\mathcal{H}} \|Mu\|_{\mathcal{H}}}{\eta}\, \alpha c_{\mathrm{op}}. \tag{11}$$

A loss slope thus combines the strength of the learning signal, the magnitude of functional motion, reachability and alignment. Measuring the factors distinguishes behaviours that their product hides.

The parameter decomposition gives a complementary view. Write $u_R = M^+ M u$ and $u_K = u - u_R$. Since u^{ref} lies in $\mathrm{range}(M^*)$, orthogonality yields

$$c_{\mathrm{w}} = \sqrt{1 - q^2}\, \cos_{\mathcal{W}}(u_R, u^{\mathrm{ref}}) \tag{12}$$

when the displayed cosines exist. A small weight cosine can therefore come from steering within the effective parameter subspace, kernel motion, or both. Kernel directions are invisible only to the sampled operators and only to first order; they can still affect other inputs and later geometry.

The same loss decrease, three different diagnoses. Consider the illustrative map $M(u_1, u_2, u_3) = (u_1, 2u_2, 0)$ and target $b = (1, 1, 1)$. Here $r = (1, 1, 0)$ and $\alpha = \sqrt{2/3}$. The updates below all satisfy $\langle b, Mu \rangle = 2$ and hence have the same first-order loss effect.

Update	u	Mu	Diagnosis
Reference	$(1, \frac{1}{2}, 0)$	$(1, 1, 0)$	$c_{\mathrm{op}} = 1,\ q = 0$
Rescaled gradient	$(\frac{2}{5}, \frac{4}{5}, 0)$	$(\frac{2}{5}, \frac{8}{5}, 0)$	$c_{\mathrm{op}} = 5/\sqrt{34},\ q = 0$
Reference plus kernel	$(1, \frac{1}{2}, \frac{\sqrt{5}}{2})$	$(1, 1, 0)$	$c_{\mathrm{op}} = 1,\ q = 1/\sqrt{2}$

The gradient overweights the second attainable coordinate because the parameterisation amplifies it. The last update adds parameter motion without changing the first-order sampled-operator effect. None can realise the third target component. Appendix A.4 derives the example and connects it to a factored linear model.

For SGD on the probed batch, $u = M^* b$ and $Mu = MM^* b$: singular directions are weighted by their squared singular values. Thus SGD can steer strongly away from r even when $\alpha = 1$ and $q = 0$. The exact reference also has $q = 0$, but cancels this weighting on the reachable target. Their difference describes the effect of parameterisation on gradient following.

3.2 Measuring a run and intervening on its trajectory

The diagnosis can be evaluated at a stored training state without replacing the optimiser. Products with M and M^* are computed through the contexts; LSQR (Paige & Saunders, 1982) approximates the reference without forming the full matrix. We evaluate candidate increments on the probe batch, retaining each rule's optimiser state. The resulting profile describes the rule at the weights and gates reached along its own trajectory. Isolating the update rule from that trajectory would require common weights and explicitly specified optimiser histories.

The diagnostic probes solve

$$u_\rho = \arg\min_v \{\|Mv - b\|_{\mathcal{H}}^2 + \rho^2 \|v\|_{\mathcal{W}}^2\}, \qquad \rho = 10^{-7} \widehat{\sigma}_{\max}, \tag{13}$$

using the augmented system $[M; \rho I]$. Here $\widehat{\sigma}_{\max} = \|Mv\|_{\mathcal{H}} / \|v\|_{\mathcal{W}}$ after one power step estimates the operator scale. Estimated scores carry hats; probes with undamped normal residual $\mathrm{ne} = \|M^*(Mu_\rho - b)\| / \|M^* b\| > 10^{-2}$ are retained and flagged. A small residual does not certify minimum norm or projection accuracy in an ill-conditioned system. Any nonzero $y = Mv$ nevertheless gives the lower bound $\alpha \geq |\langle b, y \rangle_{\mathcal{H}}| / (\|b\|_{\mathcal{H}} \|y\|_{\mathcal{H}})$, without requiring convergence (Appendix C).

To intervene on the learning process, we also train a *reference-following control*, denoted Ref. It applies k zero-initialised, undamped LSQR iterations to

$$b_i^{\mathrm{Ref}} = -\eta \frac{g_i}{\max(\|g\|_F, 10^{-12})} x_i^\top, \qquad \|g\|_F^2 = \sum_i \|g_i\|_2^2, \tag{14}$$

with $k = 50$ in the principal comparison. The target size η is tuned on validation data. This control tests how approximate reference following relates to fitting, gating and operator structure. A single LSQR iteration follows the gradient direction with a solver-dependent scalar, so the $k = 1$ control does not reproduce fixed-rate SGD.

Local diagnosis and finite-step motion. The finite-step check replays seven rules to their fitting snapshots and measures each rule's next increment on the probe batch, using two seeds. For Ref on ReLU, the mean cosine with the estimated projected target falls from .887 for Mu to .028 for $P(W + u) - P(W)$; the tested step changes gates on 94% of probe inputs (Table 3). These measurements distinguish local steering from realised motion. They do not isolate gate changes from higher-order fixed-gate effects, and the estimated target retains numerical uncertainty. The separate no-gate-change scaling check supports the local limit; dense SVD checks assess solver accuracy and reveal greater sensitivity in parameter space under ill-conditioning (Appendix C).

Table 1: Local update geometry alongside predictive performance, at each rule's own trajectory. Accuracy is mean ± seed standard deviation. REF uses the validation-selected $\eta = 1$, $k = 50$ cell in each architecture. Stateful quantities use replayed optimiser states; $\widehat{q}$ remains a numerical projection estimate. †: re-solved at 30000 iterations, with $3/5$, $5/5$ and $3/3$ seeds passing the .01 normal-residual threshold; ReLU reachability is a lower bound. Appendix D.3 reports uncertainties.

Model	Rule	$\widehat{\alpha}$	$\widehat{c}_{\mathrm{op}}$	$\widehat{q}$	Test accuracy
ReLU	SGD	1.000	.228	.0000	$.531 \pm .010$
	Adam†	$\geq .874$	.030	.742	$.634 \pm .013$
	REF	1.000	.861	.0000	$.565 \pm .013$
CReLU	SGD	1.000	.253	.0000	$.504 \pm .016$
	Adam†	1.000	.035	.557	$.612 \pm .071$
	REF	1.000	.931	.0000	$.592 \pm .011$
Leaky	SGD	1.000	.204	.0000	$.535 \pm .011$
	Adam†	1.000	.066	.616	$.654 \pm .018$
	REF	1.000	.860	.0000	$.567 \pm .010$

4 WHAT THE DIAGNOSIS REVEALS ABOUT LEARNING

4.1 STUDY DESIGN

We use deep linear regression, whose end-to-end operator can be drawn exactly, and MNIST-1D (Greydanus & Kobak, 2024), a nonlinear classification task with 40 input features and 10 classes. The principal networks have eight weight layers, 128 pre-activation units per hidden layer, no biases and He initialisation. ReLU, leaky ReLU and CReLU separate input-dependent gating from effects requiring zero slopes. Training uses cross-entropy, batches of 100, 12000 updates and $3000/1000/1000$ training/validation/test examples.

Metrics are probed every 250 updates. The first probe at which training accuracy reaches .999 is the *fitting anchor*; it and the next seven probes form the *fitting phase*. Hyperparameters are selected on mean validation accuracy over this phase; test accuracy is only reported. Principal summaries use five seeds for ReLU/CReLU and three for leaky ReLU, each generating its own dataset. Every principal seed crosses the threshold, but Adam need not remain above it. Operator diagnostics are computed every 2000 updates on a fixed 100-input batch in float64, with up to 3000 LSQR iterations, and read at the first such probe after the anchor; structural statistics use the first weight snapshot (saved every 4000 updates) after the anchor (Appendix D).

4.2 LOW ALIGNMENT CAN COEXIST WITH HIGH REACHABILITY

The linear experiment first illustrates why an endpoint does not identify the route taken to it. All eight displayed rules finish within .003 of the target operator, yet follow different paths (Figure 2, Appendix D.1). In a separate width-eight series, 26.9% of Adam's finite steps at depth 32 oppose the instantaneous negative operator gradient, versus 0.0% for gradient descent and 0.1% for the reference.

Table 1 separates two possible explanations of low alignment. SGD combines estimated reachability 1.000 with operator cosine .204–.253 and passing residual checks. This is consistent with steering of largely reachable targets: $Mu = MM^*b$ weights attainable directions unequally, even though the same-batch gradient has no kernel component. These measurements also evolve: CReLU REF's alignment falls from .931 at the fitting phase to .710 in the last quarter (Appendix D.3).

At 3000 iterations, Adam's reference solves leave mean normal residuals of .077, .037 and .022. Exact replay recovers its optimiser state at each fitting snapshot, and increasing the budget to 30000 iterations gives small estimated alignments (.030, .035, .066). The guaranteed reachability lower bound rises to .874 on ReLU and rounds to 1.000 on CReLU and leaky ReLU; $3/5$, $5/5$ and $3/3$ seeds pass the residual threshold (Appendix D.2). The small estimated alignment persists under the increased budget, although two ReLU probes remain unresolved. The corresponding kernel-fraction estimates are $\widehat{q} = .742$, .557 and .616; Appendix D.17 describes their numerical evaluation and the reported checks.

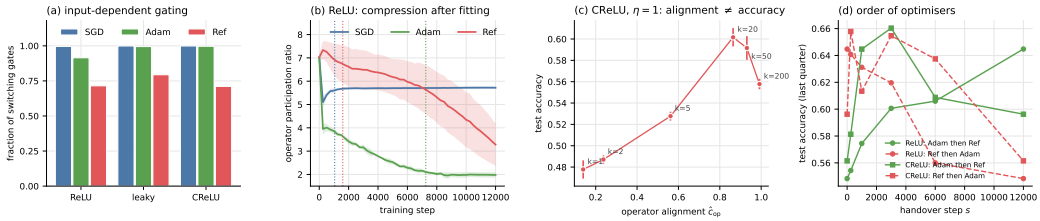

Figure 1: Local alignment, gating and operator structure describe different aspects of learning. (a) Fraction of hidden gates that switch across 256 training inputs at the first snapshot after fitting (seed means). (b) ReLU pointwise operator participation ratio during training on the 64-input probe (mean $\pm$ seed standard deviation, five seeds); dotted lines mark each rule's mean fitting step. Class-centring changes the interpretation of REF's decline (Section 4.4); the 512-input snapshot statistics there are a separate series. (c) CReLU truncation sweep at $\eta = 1$: test accuracy against operator alignment at the fitting phase for $k \in \{1, 2, 5, 20, 50, 200\}$ (mean $\pm$ seed standard deviation; three seeds, five at $k = 50$). (d) Optimiser handover: Adam for the first s updates and then REF (solid), or the reverse (dashed), on ReLU (circles) and CReLU (squares); last-quarter test accuracy, mean of three seeds. Settings are in Appendices D.7, D.8, D.6 and D.12.

The expanded comparison shows that predictive performance does not order these local profiles (Appendix D.17). On ReLU, heavy ball (Polyak, 1964) and Adam (Kingma & Ba, 2015) score .642 and .634 at estimated alignments .150 and .030; Muon (Jordan et al., 2024) scores .475 at .039. Shampoo and SOAP provide further profiles (Gupta et al., 2018; Vyas et al., 2025). K-FAC (Martens & Grosse, 2015) scores .562 at alignment .251 on ReLU and .594 at .361 on CReLU, with its best rate at the edge of the tested grid. Table 27 also reports distinct kernel-fraction estimates at replayed states. Their interpretation depends on projection accuracy and the geometry visited by each rule; the comparison does not isolate a mechanism intrinsic to the optimiser. Matched-loss comparisons show that different fitting progress does not alone explain the principal accuracy ordering: Adam leads at every reported threshold, with REF usually between Adam and SGD; leaky ReLU at loss 10^{-1} is a small exception, where SGD exceeds REF (Appendix D.4). For context, validation-selected 10-nearest neighbours scores .573 $\pm$ .015, near REF's .565–.592, while logistic regression scores .333 $\pm$ .016 (Appendix D.5).

4.3 CLOSER REFERENCE FOLLOWING NEED NOT IMPROVE PREDICTION

The truncation sweep tests whether following the reference more closely leads to better prediction. In CReLU, all $k \in \{1, 2, 5, 20, 50, 200\}$ settings fit: alignment rises from .140 to .992, while accuracy peaks at .602 for $k = 20$ and falls to .558 at $k = 200$ (Figure 1c; Appendix D.6). Seed deviations are .004–.011 (three seeds, five at $k = 50$), and all probes pass the numerical threshold. ReLU separates this comparison from fitting failure: no seed fits at $k = 1$ and only one of three at $k = 2$, while accuracy again falls from $k = 50$ to 200. Thus local reference alignment and predictive performance are distinct outcomes even within the control family.

Smoothing REF's weight increments lowers alignment and gives nonpositive mean accuracy changes in all twelve tested settings (Appendix D.14). It does not reproduce Adam's advantage, although this does not rule out momentum's contribution to Adam. Two-seed isotropic perturbations instead yield small, phase-dependent gains, with fitting-phase accuracy .560 $\rightarrow$ .573 at relative size .3, while size 3 prevents fitting; range noise at size .1 gives a similar gain to isotropic noise (Appendix D.20). These controls show that the outcome depends on how the trajectory is perturbed.

4.4 APPARENT COMPRESSION DEPENDS ON WHAT THE OPERATOR REPRESENTS

We measure pointwise operator complexity by the singular-value participation ratio

$$\mathrm{PR}(P) = \frac{(\sum_j \sigma_j(P))^2}{\sum_j \sigma_j(P)^2}. \tag{15}$$

Snapshot estimates average over 512 inputs; no zero operator occurs. Across-input diversity is the PR of the row-centred matrix of vectorised operators. These statistics measure different forms of spectral concentration: REF has the highest snapshot pointwise PR in every architecture, but the highest raw diversity only on leaky ReLU (Appendix D.8). Neither quantity is an algebraic rank.

The control exhibits substantial structural change after fitting. On the 512-input ReLU probe, between steps 4000 and 12000, REF's PR falls from 6.38 to 3.29 and its diversity from 191.5 to 109.8 (Table 11). The separate 64-input trajectory in Figure 1b shows the same pattern. SGD changes little over this interval; Adam also changes, with PR $2.79 \to 1.98$ and diversity $190.6 \to 161.4$ (Tables 12–13). Adam's nearly constant selected-snapshot summaries start later, at its own post-fitting snapshots. To interpret the control's decline, we must also specify what the operator represents: $P(x)$ describes logits, some changes to which cannot affect prediction. With $Q = I - \mathbf{1}\mathbf{1}^\top/C$,

$$\mathrm{softmax}(P(x)x) = \mathrm{softmax}(QP(x)x). \tag{16}$$

Class-centring removes a prediction-invariant component from the structural measurement. The update diagnostics retain the declared full-logit map M; their values should therefore be interpreted as full-logit geometry, without assuming they are unchanged for the class-centred map QM.

Class-centring changes the interpretation of the apparent compression. ReLU REF's PR decreases from 6.47 to 5.35, but diversity changes only from 221.0 to 218.8 (Appendix D.8). CReLU and leaky ReLU show the same distinction: class-centred diversity changes from 220.2 to 220.1 and 219.6 to 216.7. Thus the pronounced decline in raw diversity does not persist in the operator governing class differences, although some pointwise concentration remains. A direct decomposition of the net displacement supports this reading: between the two snapshots, $86 \pm 8\%$ (ReLU), $98 \pm 2\%$ (CReLU) and $92 \pm 5\%$ (leaky) of $\sum_i \|P_{\mathrm{end}}(x_i) - P_{\mathrm{fit}}(x_i)\|_F^2$ lies in the common-logit component $(I - Q)$, against 1% for SGD. These fractions describe the net change between snapshots, not accumulated per-step motion. A large change in the raw operator can therefore coexist with nearly stable diversity in its prediction-relevant class differences.

The supporting controls show where this interpretation applies. Raw PR declines after fitting in 54 of the 55 fitting $k = 50$ runs, whereas five of six $k = 200$ runs do not decline; the exception at $k = 50$ is CReLU $\eta = 10$, which fits only at update 7750. A decline is the first one-unit drop from the fitting-anchor PR on the 64-input probe, with median lag 3250 updates. A single unnormalised-target control at $\eta = .5$ retains PR 6.27, versus 1.59 with normalisation, but unequal losses prevent a matched-progress causal conclusion (Appendix D.8). Layerwise PCA also records concentration: REF's output top-two-component share rises from 74% to 93% on ReLU and 69% to 99% on CReLU (Appendix D.9). Because this PCA uses raw logits, it likewise requires the distinction between logit structure and class differences.

4.5 REDUCED GATE SWITCHING DOES NOT REQUIRE DEAD UNITS

We next ask whether the control's changes in gating require units to become inactive. A gate switches when its state varies across a finite input set. Across 256 training inputs, this fraction for REF is .715/.794/.711 on ReLU/leaky/CReLU, against SGD's .996/.999/.999 (Figure 1a). Both leaky slopes are nonzero, yet .107 of REF's leaky units stay on the positive-preactivation branch and .099 on the nonpositive branch throughout the set. Reduced switching therefore does not require dead units. Over all 3000 inputs, the fractions rise to .791/.856/.783, while SGD remains at least .996 (Appendix D.7). Fitting-phase disagreement between paired inputs (gate diversity) likewise falls from SGD's .425/.451/.388 to REF's .236/.272/.253.

The partition analysis explains why a gate count is incomplete. On sampled data planes, REF concentrates partition edges nearer the training anchors than either baseline in all twelve seed-level contrasts, across three networks per architecture (Appendix D.10). Fewer switching units can thus coexist with more locally concentrated boundaries.

The gate measurements also provide an exploratory comparison with predictive performance. On 64 inputs, density records positive gate states, diversity records disagreement between paired inputs, and churn records changes between probes 250 updates apart. Across 53 fitting cells, after removing architecture–optimiser means, their pooled correlations with test accuracy are $-.58$, $+.51$ and $+.56$. Operator alignment has a weaker association ($+.15$, interval including zero); weight alignment has a small negative association ($-.14$), and training loss is also associated (Appendix D.18). These results

concern variation within the tested hyperparameter grids; they do not establish the same relationship for every optimiser or a cause of generalisation.

4.6 Architecture and learning history qualify the diagnosis

Architecture and training history test how far a diagnosis at one setting transfers. At CReLU width 16, REF fits two of three seeds and Adam all three, while SGD has no fitting cell; REF's endpoint accuracy increases across widths 16–128, giving no smaller-network advantage. At width 32, it exceeds Adam under both endpoint and fitting-phase selection, reversing the principal width-128 ordering (Appendix D.11). Across depths, its ReLU advantage over SGD at depth 8 reverses at depth 32 for $k = 50$. Raising k to 200 improves phase accuracy $.445 \rightarrow .504$ despite lower estimated alignment (Appendix D.16). This comparison concerns separately trained states. At one fixed depth-32 checkpoint, increasing the solve budget $50 \rightarrow 3000$ instead raises alignment $.71 \rightarrow .9999$, illustrating why solver convergence and comparisons between trajectories must be distinguished.

Handover experiments test whether an optimiser's effect depends on the state it inherits. With 6000 updates of each rule, Adam followed by REF scores .606 on ReLU against .560 for the reverse order, higher in all three paired seeds (differences .034, .073, .031). On CReLU the order has no consistent effect (.609 against .637; paired differences .000, $-.096$, .009). Adam reaches its usual accuracy after a REF start given enough updates (.620 and .655 after 9000 Adam updates). Order dependence is therefore moderate and architecture-specific rather than an irreversible imprint (Figure 1d; Appendix D.12).

Longer training further limits conclusions from one fitting phase. In single test-selected ReLU continuations to 300000 steps, Adam's accuracy falls to .477 as its training loss rises to .69, below SGD (.527) and REF (.498) (Appendix D.13). Adam's 104 loss spikes coincide with higher gate churn, .0348 against .0058 otherwise.

4.7 Predictive meaning and practical scope

Under a margin-based PGD attack, REF has the highest robust/clean accuracy ratios on ReLU and CReLU; at radius .1, CReLU gives 44.8%, against SGD's 30.9% and Adam's 22.5% (Appendix D.15). The CIFAR-10 extension tests feasibility rather than confirming the MNIST-1D findings (Appendix D.19). No reported cell fits the train set completely with the given budget, and the two one-seed REF trials ($\eta = 0.3$ and 1) reach test accuracy .441 and .362 at training losses 1.35 and 3.99, far from the baselines'. Its 32-input probe already contains 983040 target entries, against 40000 on MNIST-1D. A float64 probe of a depth-eight MNIST-1D network takes about four seconds (Appendix F), so the diagnosis suits targeted analysis of selected states rather than continuous monitoring at scale.

5 Related work

Deep linear models connect parameter dynamics with end-to-end learning. Exact dynamics for whitened quadratic regression with balanced, aligned initialisation (Saxe et al., 2014), and the induced preconditioner under balanced gradient flow (Arora et al., 2018), explain how factorisation changes learning. Matrix-factorisation analyses investigate low-rank tendencies and the limits of norm-based implicit-bias descriptions (Gunasekar et al., 2017; Arora et al., 2019; Razin & Cohen, 2020; Li et al., 2021). Our framework builds on this operator perspective to diagnose a finite nonlinear training state. Fixed-gate Jacobian analysis studies depth scaling and spectral separation (Haas et al., 2026); conditional rank-bias results describe other structural effects (Timor et al., 2023; Jacot, 2023). We distinguish these state properties from the attainable motion and selected direction that produce their evolution.

The reference $-\eta(M^*M)^+\nabla_W\mathcal{L}$ connects the diagnosis to natural-gradient and Gauss–Newton geometry (Amari, 1998; Martens, 2020). In deep linear regression with whitened inputs and an isotropic quadratic output metric, the output-Jacobian Gram is proportional to the operator Gram, and Moore–Penrose representatives agree up to scale. Other generalised inverses can induce the same operator velocity with different parameter updates (Bernacchia et al., 2018); cross-entropy does not

inherit the isotropic-metric identity. Layerwise least-squares methods also use geometric reference steps to assess approximations (Dufort-Labbé et al., 2026).

Minimum-perturbation analyses prescribe output changes on a batch, with exact single-layer expressions under appropriate assumptions and multilayer bounds (Evans & Tanner, 2026). Our target specifies complete local operators at sampled inputs. In the homogeneous setting, an output change constrains the action of an operator change along its input; the operator target constrains the full matrix. Jacobian-spectrum and dynamical-isometry studies (Wang et al., 2016; Pennington et al., 2017) motivate measuring the input–output map, while its parameter derivative M is the object needed to diagnose reachability and update steering.

Function-space views describe how a parameter update moves a network's outputs through a Gram operator: the neural tangent kernel (Jacot et al., 2018) plays for outputs the role that MM^* plays for sampled operators. Optimisers can also be read as steepest descent under different parameter norms (Bernstein & Newhouse, 2024). These views characterise an update by its metric. The diagnosis instead measures, at a given state, how much of a declared functional request is attainable, how far an optimiser's first-order functional change departs from it, and how much of its parameter motion leaves the sampled operators unchanged. Training dynamics concentrate in a low-dimensional subspace of top Hessian directions (Gur-Ari et al., 2018); the kernel fraction q gives a complementary, operator-defined split of parameter motion. Parameter symmetries and conservation laws identify directions that leave the whole function unchanged (Kunin et al., 2021; Marcotte et al., 2023), whereas $\ker M$ is defined on a finite input set and can contain directions that change the function elsewhere.

Adaptive and non-adaptive optimisers can reach solutions that generalise differently (Wilson et al., 2017); our experiments ask which local and structural statistics accompany such differences, rather than ranking optimisers. The gate measurements relate to the study of linear regions, whose number and density near the data characterise the function a piecewise-linear network realises (Hanin & Rolnick, 2019), and to gated linear networks, which separate the role of the gates from that of the weights (Lippl et al., 2022). We relate these gate statistics to the operator motion that each update produces.

6 Discussion and conclusion

Operator projection separates local reachability, steering and parameter motion invisible to the sampled operators. The experiments establish three diagnostic distinctions: a largely reachable target can coexist with low alignment; closer reference following need not improve prediction; and pronounced raw-operator compression can coexist with nearly stable class-centred diversity. The last finding makes the represented function part of the diagnosis: a structural change can primarily concern a component that prediction ignores.

The finite-step and gate controls further connect this geometry to learning: realised alignment can be much smaller than first-order alignment, and reduced switching does not require dead units. These findings support using the framework to interpret the relationship between update direction, finite motion and learned structure, with each conclusion tied to the state and measurement that supports it.

Limitations. The diagnosis is local, probe- and metric-dependent, and tested mainly on small bias-free MNIST-1D networks; CIFAR-10 establishes feasibility only. Finite-step checks do not isolate gate effects. Two ReLU Adam reference solves remain above the residual threshold, and kernel fractions retain numerical approximation error. Full-logit update geometry differs from class-centred structure. Comparisons use each rule's own trajectory; gate correlations are exploratory. K-FAC's best rate lies at its grid boundary, long continuation uses test-selected runs, and robustness concerns the stated final-iterate attack.

AI USE STATEMENT

In this work, we used generative AI tools for coding and figure generation as well as for polishing writing. We have not used generative AI tools for the scientific content, original ideas and the general structure of the paper. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies how optimisation rules change the functions computed by neural networks. Its contribution is a diagnostic framework and a small-scale empirical analysis. The measurements do not provide guarantees of model reliability, fairness or robustness and should not replace evaluation of those properties in deployment.

REPRODUCIBILITY STATEMENT

Sections 2–3.2 and Appendices A–C define the operator geometry and its numerical approximation; Appendix D gives the experimental settings. Anonymised code is available at `https://anonymous.4open.science/r/operator-projection-diagnostics-79B2`, with configurations, scripts for the tables and figures, summary tables and unit tests. Replaying the stored configurations recovers the optimiser states used for the stateful measurements and reproduces their saved weights exactly. The noise runs have a separate seed-reproducibility limitation, stated in Appendix D.20.

REFERENCES

Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998. doi: 10.1162/089976698300017746.

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 244–253, 2018.

Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Alberto Bernacchia, Máté Lengyel, and Guillaume Hennequin. Exact natural gradient in deep linear networks and its application to the nonlinear case. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Jeremy Bernstein and Laker Newhouse. Old optimizer, new norm: An anthology. arXiv:2409.20325, 2024.

Simon Dufort-Labbé, Pierre-Luc Bacon, Razvan Pascanu, Simon Lacoste-Julien, and Aristide Baratin. Layerwise LQR for geometry-aware optimization of deep networks. arXiv:2605.04230, 2026.

Bethan Evans and Jared Tanner. Theory of minimal weight perturbations in deep networks and its applications for low-rank activated backdoor attacks. arXiv:2601.16880, 2026.

Sam Greydanus and Dmitry Kobak. Scaling down deep learning with MNIST-1D. In *International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 16404–16415, 2024.

Suriya Gunasekar, Blake E. Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nathan Srebro. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1842–1850. PMLR, 2018.

Guy Gur-Ari, Daniel A. Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. arXiv:1812.04754, 2018.

Nathanaël Haas, François Gatine, Augustin Cosse, and Zied Bouraoui. Why deep Jacobian spectra separate: Depth-induced scaling and singular-vector alignment. arXiv:2602.12384, 2026.

Boris Hanin and David Rolnick. Complexity of linear regions in deep networks. In *International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2596–2604, 2019.

Arthur Jacot. Implicit bias of large depth networks: a notion of rank for nonlinear functions. In *International Conference on Learning Representations*, 2023.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks. `https://kellerjordan.github.io/posts/muon/`, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Daniel Kunin, Javier Sagastuy-Brena, Surya Ganguli, Daniel L. K. Yamins, and Hidenori Tanaka. Neural mechanics: Symmetry and broken conservation laws in deep learning dynamics. In *International Conference on Learning Representations*, 2021.

Zhiyuan Li, Yuping Luo, and Kaifeng Lyu. Towards resolving the implicit bias of gradient descent for matrix factorization: Greedy low-rank learning. In *International Conference on Learning Representations*, 2021.

Samuel Lippl, L. F. Abbott, and SueYeon Chung. The implicit bias of gradient descent on generalized gated linear networks. arXiv:2202.02649, 2022.

Sibylle Marcotte, Rémi Gribonval, and Gabriel Peyré. Abide by the law and follow the flow: Conservation laws for gradient flows. In *Advances in Neural Information Processing Systems*, volume 36, pp. 63210–63221, 2023.

James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020.

James Martens and Roger Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2408–2417. PMLR, 2015.

Christopher C. Paige and Michael A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, 1982. doi: 10.1145/355984.355989.

Jeffrey Pennington, Samuel S. Schoenholz, and Surya Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Boris T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964. doi: 10.1016/0041-5553(64)90137-5.

Noam Razin and Nadav Cohen. Implicit regularization in deep learning may not be explainable by norms. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21174–21187, 2020.

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *International Conference on Learning Representations*, 2014.

Wenling Shang, Kihyuk Sohn, Diogo Almeida, and Honglak Lee. Understanding and improving convolutional neural networks via concatenated rectified linear units. In *International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 2217–2225, 2016.

Nadav Timor, Gal Vardi, and Ohad Shamir. Implicit regularization towards rank minimization in ReLU networks. In *International Conference on Algorithmic Learning Theory*, volume 201 of *Proceedings of Machine Learning Research*, pp. 1429–1459, 2023.

Nikhil Vyas, Depen Morwani, Rosie Zhao, Mujin Kwun, Itai Shapira, David Brandfonbrener, Lucas Janson, and Sham Kakade. SOAP: Improving and stabilizing Shampoo using Adam for language modeling. In *International Conference on Learning Representations*, 2025.

Shengjie Wang, Abdel-rahman Mohamed, Rich Caruana, Jeff Bilmes, Matthai Philipose, Matthew Richardson, Krzysztof Geras, Gregor Urban, and Özlem Aslan. Analysis of deep neural networks with extended data Jacobian matrix. In *International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 718–726, 2016.

Ashia C. Wilson, Rebecca Roelofs, Mitchell Stern, Nathan Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

A Operator geometry and proofs

A.1 The operator and its local derivative

For each input x_i, let $h_0 = x_i$, $z_\ell = W_\ell h_{\ell-1}$ and $h_\ell = D_\ell(z_\ell)z_\ell$ for $\ell < L$, with a linear output layer. Writing the gates explicitly gives

$$P_i = W_L D_{L-1}(x_i) W_{L-1} \cdots D_1(x_i) W_1, \qquad f_W(x_i) = P_i x_i. \tag{17}$$

For ReLU, leaky ReLU and CReLU, this identity follows coordinate by coordinate from the definition of the activation. Away from activation boundaries, $P_i = \partial f_W(x_i)/\partial x_i$. At a zero pre-activation, a gate convention still gives the factorisation in Equation (17), but the ordinary input Jacobian need not exist.

The dimensions of a gate need not be square. If $z \in \mathbb{R}^d$, CReLU has

$$D(z) = \begin{bmatrix} \operatorname{diag}(\mathbf{1}[z > 0]) \\ -\operatorname{diag}(\mathbf{1}[z < 0]) \end{bmatrix} \in \mathbb{R}^{2d \times d}. \tag{18}$$

The next weight matrix therefore has $2d$ input columns. For this strict-sign convention, $D(z)^\top D(z) = I$ only when every coordinate of z is nonzero; in general it equals $\operatorname{diag}(\mathbf{1}[z \neq 0])$. If the next weight matrix is $[O \mid -O]$, its action on the CReLU activations is nevertheless Oz for every z, including coordinates equal to zero. Away from zero pre-activations, $[O \mid -O]D(z) = O$. A chain built from such paired blocks represents the corresponding linear product at initialisation. It is an orthogonal operator only when that product is square and orthogonal; rectangular input and output layers require their own rank and conditioning assumptions.

Define the contexts, with input dependence understood, by

$$\begin{aligned} B_{1,i} &= I, \quad B_{\ell+1,i} = D_\ell(x_i) W_\ell B_{\ell,i}, \\ A_{L,i} &= I, \qquad A_{\ell,i} = A_{\ell+1,i} W_{\ell+1} D_\ell(x_i). \end{aligned} \tag{19}$$

Then $P_i = A_{\ell,i} W_\ell B_{\ell,i}$ for every layer, with compatible dimensions also for CReLU.

Suppose that all pre-activations on the finite probe batch are nonzero. By continuity, there is a neighbourhood of W on which all of those gate signs remain fixed. In that neighbourhood, P_i is a polynomial in the weights. Expanding the product gives

$$P_i(W + u) - P_i(W) = \sum_{\ell=1}^{L} A_{\ell,i} u_\ell B_{\ell,i} + O(\|u\|_{\mathcal{W}}^2). \tag{20}$$

The linear term is the step map of Equation (4). The bound is local to this fixed-gate neighbourhood. It does not cover a finite step that crosses an activation boundary: the network output is continuous there, but its input Jacobian, and hence the operator P_i, can jump. At a boundary, the frozen-gate map is an explicitly chosen local model rather than a generally valid derivative of $W \mapsto P_i(W)$.

A.2 Batch geometry and the adjoint

Let $\mathcal{W}$ be the product of the layer matrix spaces, with $\langle u, v \rangle_{\mathcal{W}} = \sum_\ell \operatorname{tr}(u_\ell^\top v_\ell)$. For a batch of B inputs, the operator space is $\mathcal{H} = (\mathbb{R}^{C \times d})^B$, equipped with

$$\langle H, K \rangle_{\mathcal{H}} = \frac{1}{B} \sum_{i=1}^{B} \operatorname{tr}(H_i^\top K_i). \tag{21}$$

The trace identity $\langle AuB, H \rangle_F = \langle u, A^\top H B^\top \rangle_F$ then gives

$$(M^* H)_\ell = \frac{1}{B} \sum_{i=1}^{B} A_{\ell,i}^\top H_i B_{\ell,i}^\top. \tag{22}$$

Here M^* denotes the adjoint for these two inner products. This convention keeps the batch-average loss and its gradient consistent.

In particular, for $\mathcal{L}(W) = B^{-1} \sum_i \ell(f_W(x_i), y_i)$, put $g_i = \nabla_f \ell(f_W(x_i), y_i)$ and $G_i = g_i x_i^\top$. Then $\nabla_W \mathcal{L} = M^* G$ wherever the preceding derivative is valid. Squared loss means $\ell(f, y) = \frac{1}{2}\|f - y\|^2$, for which $g_i = f_W(x_i) - y_i$; for softmax cross-entropy, $g_i = \operatorname{softmax}(f_W(x_i)) - e_{y_i}$. Thus the gradient-descent update with step size η is exactly $M^* b$ for the target $b = -\eta G$. This identity requires the update and the diagnostic to use the same batch and loss reduction.

A.3 THE REFERENCE AND THE DECOMPOSITION

All statements in this subsection are finite-dimensional and hold for an arbitrary linear step map M. Let b be the target in Equation (3), and let $\Pi = MM^+$ be the orthogonal projector in $\mathcal{H}$ onto $\mathrm{range}(M)$. If

$$M\psi_j = \sigma_j \phi_j, \qquad M^*\phi_j = \sigma_j \psi_j, \qquad \sigma_j > 0, \tag{23}$$

is a singular system in the specified inner products, then

$$r := \Pi b = \sum_j \langle b, \phi_j \rangle_{\mathcal{H}} \phi_j, \quad u^{\mathrm{ref}} = M^+ b = \sum_j \frac{\langle b, \phi_j \rangle_{\mathcal{H}}}{\sigma_j} \psi_j. \tag{24}$$

Every u with $Mu = r$ has the form $u = u^{\mathrm{ref}} + z$, where $z \in \ker M$. Since $u^{\mathrm{ref}} \in \mathrm{range}(M^*) = (\ker M)^\perp$,

$$\|u\|_{\mathcal{W}}^2 = \|u^{\mathrm{ref}}\|_{\mathcal{W}}^2 + \|z\|_{\mathcal{W}}^2. \tag{25}$$

This proves that Equation (6) is the unique minimum-norm least-squares solution. It also gives $M^+ = (M^*M)^+ M^* = M^*(MM^*)^+$, and hence the preconditioned-gradient expression.

To prove Proposition 1, write $b = r + b_\perp$, where $b_\perp \perp \mathrm{range}(M)$. For every update u,

$$\|Mu - b\|_{\mathcal{H}}^2 = \|b_\perp\|_{\mathcal{H}}^2 + \|Mu - r\|_{\mathcal{H}}^2. \tag{26}$$

The first term is the unavoidable residual of this local model; the second is the discrepancy between the candidate operator change and the projection. If b, r and Mu are nonzero, orthogonality also yields

$$\cos_{\mathcal{H}}(Mu, b) = \frac{\|r\|_{\mathcal{H}}}{\|b\|_{\mathcal{H}}} \cos_{\mathcal{H}}(Mu, r) = \alpha c_{\mathrm{op}}. \tag{27}$$

The same identity explains which information the instantaneous loss slope aggregates. At a differentiable state, for the same batch, loss and metrics as above, $\nabla_W \mathcal{L} = M^* G$ and $b = -\eta G$ imply

$$\begin{aligned} D\mathcal{L}(W)[u] = \langle G, Mu \rangle_{\mathcal{H}} &= -\frac{\langle b, Mu \rangle_{\mathcal{H}}}{\eta} \\ &= -\frac{\|b\|_{\mathcal{H}} \|Mu\|_{\mathcal{H}}}{\eta} \alpha c_{\mathrm{op}}. \end{aligned} \tag{28}$$

The last expression requires the diagnostic cosines to be defined. It separates the magnitude of the operator change, target reachability and alignment within the reachable space, whereas the loss derivative reports only their product. It is an instantaneous, local identity; by itself it gives no guarantee about a finite training step, particularly one crossing activation boundaries. A kernel component contributes zero to this derivative, but can affect higher-order changes and subsequent states.

Consequently, α is the greatest alignment with b attainable by a nonzero vector in $\mathrm{range}(M)$ when $r \neq 0$. It is a fraction of the target norm, not an upper bound on the norm of an arbitrary induced update: multiplying u by a scalar can make $\|Mu\|$ arbitrarily large. If $b = 0$, $r = 0$, $Mu = 0$, or $u^{\mathrm{ref}} = 0$, any ratio or cosine with a zero denominator is undefined and must be reported as such.

For a weight update, its orthogonal decomposition is

$$u = M^+ Mu + (I - M^+ M)u, \quad M^+ Mu \in \mathrm{range}(M^*), \quad (I - M^+ M)u \in \ker M. \tag{29}$$

Thus a kernel fraction defined by $\|(I - M^+M)u\|_{\mathcal{W}} / \|u\|_{\mathcal{W}}$ is a norm fraction; its square is the fraction of squared norm. A zero operator derivative on the probe batch establishes first-order invariance on that batch. It does not establish invariance for finite steps or for unseen inputs, and the batch kernel need not consist solely of parameter symmetries.

For a candidate u, write $u_R = M^+ Mu$, $u_K = (I - M^+ M)u$ and $u^{\mathrm{ref}} = M^+ b$. If u, u_R and u^{ref} are nonzero, its weight cosine and kernel fraction satisfy

$$c_{\mathrm{w}} = \sqrt{1 - q^2}\, \cos_{\mathcal{W}}(u_R, u^{\mathrm{ref}}), \qquad q = \frac{\|u_K\|_{\mathcal{W}}}{\|u\|_{\mathcal{W}}}. \tag{30}$$

Indeed, $u_K \perp u^{\mathrm{ref}}$, so the numerator of c_{w} is $\langle u_R, u^{\mathrm{ref}} \rangle_{\mathcal{W}}$, while Pythagoras gives $\|u_R\|_{\mathcal{W}} / \|u\|_{\mathcal{W}} = \sqrt{1 - q^2}$. A small weight cosine can therefore reflect a different direction within $\mathrm{range}(M^*)$, motion in $\ker M$, or both. It does not identify kernel motion on its own. If $u_R = 0$ but u and u^{ref} are nonzero, then $q = 1$ and $c_{\mathrm{w}} = 0$ directly; the cosine involving u_R is undefined and Equation (30) is not used.

Table 2: Three update directions with the same first-order loss decrease in the analytic example. Their operator changes and parameter routes differ.

Update	u	Mu	c_{op}	c_{w}	q
Reference	$(1, \frac{1}{2}, 0)$	$(1, 1, 0)$	1	1	0
Rescaled gradient	$(\frac{2}{5}, \frac{4}{5}, 0)$	$(\frac{2}{5}, \frac{8}{5}, 0)$	$5/\sqrt{34}$	$4/5$	0
Reference + kernel	$(1, \frac{1}{2}, \frac{\sqrt{5}}{2})$	$(1, 1, 0)$	1	$1/\sqrt{2}$	$1/\sqrt{2}$

A.4 An example of what the diagnostics distinguish

Consider the local map and target

$$M(u_1, u_2, u_3) = (u_1, 2u_2, 0), \qquad b = (1, 1, 1), \tag{31}$$

with Euclidean inner products. This is an analytic illustration, not a training experiment. The third target component is unreachable, the second reachable parameter direction is amplified twice as strongly as the first, and the third parameter direction lies in the kernel. Direct calculation gives

$$r = \Pi b = (1, 1, 0), \qquad \alpha = \sqrt{\frac{2}{3}}, \qquad u^{\mathrm{ref}} = M^{+} b = (1, \tfrac{1}{2}, 0). \tag{32}$$

Suppose the operator-space loss gradient at the current state is $G = -b$. This occurs, for example, for the quadratic loss $\frac{1}{2}\|P - P_\star\|^2$ at $P_\star = P + b$. The negative parameter gradient is $M^* b = (1, 2, 0)$. Table 2 compares the reference, a positively rescaled gradient direction, and the reference with an added kernel component. Every row satisfies $\langle b, Mu \rangle = 2$. Consequently, an infinitesimal parameter step εu has the same first-order loss change in all three cases,

$$\mathcal{L}(W + \varepsilon u) - \mathcal{L}(W) = -2\varepsilon + O(\varepsilon^2), \tag{33}$$

when the local smoothness assumptions hold. Rescaling the target to εb also rescales its reference to $\varepsilon u^{\mathrm{ref}}$; all the displayed diagnostic scores remain unchanged.

The gradient direction assigns four times as much operator motion to the second coordinate as to the first. Its parameter update has no kernel component, so this redistribution cannot be attributed to locally invisible motion. The third row instead has the same operator change as the reference, with half of its squared parameter norm in the kernel. Its smaller weight cosine comes entirely from that added component, as Equation (30) shows. Finally, all three share the same reachability α: no choice of update can supply the third requested component through this local map. A scalar first-order loss change does not distinguish these three explanations.

The example can be realised by a small factored linear model. Let $f_{a,b,c,d}(x) = abx_1 + cdx_2$ for $x \in \mathbb{R}^3$, with no dependence on x_3, and consider $a = b = 1/\sqrt{2}$, $c = d = \sqrt{2}$. Use the orthonormal increment coordinates

$$\begin{aligned} u_1 &= (\delta a + \delta b)/\sqrt{2}, \quad u_2 = (\delta c + \delta d)/\sqrt{2}, \\ u_3 &= (\delta a - \delta b)/\sqrt{2}, \quad u_4 = (\delta c - \delta d)/\sqrt{2}. \end{aligned} \tag{34}$$

The operator is $P = (ab, cd, 0)$, and its derivative is $(u_1, 2u_2, 0)$. Setting the additional kernel coordinate $u_4 = 0$ yields Equation (31), without changing the Euclidean norm. This construction also illustrates the local qualification: a pure $u_3 = t$ perturbation changes ab by $-t^2/2$ even though its first-order operator change is zero. Kernel motion is not generally finite-step function preservation.

A.5 What is invariant

At fixed weights, batch and optimiser state, both diagnostic cosines are invariant under multiplication of the candidate update by a positive scalar. They are also invariant under positive rescaling of b, which rescales r and u^{ref}. This removes the current scalar step length from the directional comparison. It does not remove the effect of a learning-rate history on the weights, gates or optimiser state, nor does it make a finite-step trajectory independent of step size.

The reference depends on two declared metrics: the Frobenius metric on the sampled operators and the Euclidean metric on the weights. Under a smooth change of weight coordinates $W = \phi(\theta)$ with

invertible Jacobian $D\phi$, the derivative becomes $M\, D\phi$. Its range is unchanged, so the projected operator direction remains the same when the operator metric is held fixed. Its minimum-Euclidean-norm lift generally changes. For example, with a scalar output operator $P = w_1 + w_2$, the minimum-norm lift of a change b is $(b/2, b/2)$. In coordinates $w_1 = c\theta_1$, $w_2 = \theta_2$, minimum Euclidean norm in θ gives the weight displacement $(c^2 b/(c^2+1), b/(c^2+1))$. The two lifts have the same operator effect but different weight directions. Changing input coordinates can also change the Frobenius operator metric. The reference is therefore well defined relative to these choices; it is not a metric-free notion of an unbiased update.

B Deep linear networks

For a deep linear chain, $P = W_L \cdots W_1$ is shared by the batch. Its loss gradient is the aggregate matrix

$$\overline{G} = \frac{1}{B} \sum_{i=1}^{B} g_i x_i^\top. \tag{35}$$

In this subsection the output of the step map is a single matrix, with its ordinary Frobenius inner product:

$$\mathcal{M}(u) = \sum_{\ell} A_\ell u_\ell B_\ell, \qquad (\mathcal{M}^* H)_\ell = A_\ell^\top H B_\ell^\top. \tag{36}$$

The shared target is $b_{\mathrm{lin}} = -\eta \overline{G}$. This reduction has the same minimum-norm least-squares weight update as stacking per-example targets, because

$$\frac{1}{B} \sum_i \|\mathcal{M}(u) + \eta G_i\|_F^2 = \|\mathcal{M}(u) + \eta \overline{G}\|_F^2 + \frac{\eta^2}{B} \sum_i \|G_i - \overline{G}\|_F^2. \tag{37}$$

However, the two target spaces have different reachability denominators. Reachability of the shared target must not be identified with reachability of the full collection of per-example targets.

Define the self-adjoint positive-semidefinite transfer operator

$$\mathcal{T}(H) = \mathcal{M}\mathcal{M}^*(H) = \sum_{\ell} A_\ell A_\ell^\top H B_\ell^\top B_\ell. \tag{38}$$

The exact solution of the *linearised* reference problem is

$$H = \mathcal{T}^+ b_{\mathrm{lin}}, \qquad (u^{\mathrm{ref}})_\ell = A_\ell^\top H B_\ell^\top, \qquad \mathcal{M}(u^{\mathrm{ref}}) = \mathcal{T}\mathcal{T}^+ b_{\mathrm{lin}}. \tag{39}$$

This follows directly from $\mathcal{M}^+ = \mathcal{M}^*(\mathcal{M}\mathcal{M}^*)^+$. It is an exact algebraic characterisation at arbitrary weights; evaluating $\mathcal{T}^+$ can still require a numerical linear solve. It is not a closed-form solution of the finite product constraint $(W_L + u_L) \cdots (W_1 + u_1) = P + b_{\mathrm{lin}}$.

A sufficient condition for $\mathcal{M}$ to be surjective is that, for some layer ℓ, A_ℓ has full row rank and B_ℓ has full column rank. Given any matrix H, the single-layer update $u_\ell = A_\ell^+ H B_\ell^+$ then satisfies $A_\ell u_\ell B_\ell = H$. This condition holds, for example, for a square chain with invertible factors. A parameter count alone does not imply it. If two or more layers are zero, all first derivatives of the product can vanish even when the network has many parameters. In a gated model, two probe inputs with the same gate pattern necessarily have the same operator and the same frozen-gate operator derivative, which already excludes arbitrary independent targets at those inputs.

For a linear chain, the output Jacobian maps u to $(\mathcal{M}(u)x_i)_i$. Its Gram satisfies

$$J_f^* J_f(u) = \mathcal{M}^*\big(\mathcal{M}(u) C_x\big), \qquad C_x = \frac{1}{B} \sum_i x_i x_i^\top. \tag{40}$$

Thus $C_x = cI$, $c > 0$, makes the output Gram proportional to the operator Gram. For squared loss, or an isotropic Gaussian observation model with fixed variance, this connects the reference to Gauss–Newton or natural gradient, up to a scalar step size. For softmax cross-entropy, the Fisher contains an additional output metric depending on class probabilities, so whitening the inputs alone does not establish the same identity.

Finally, at the all-zero DLN parameter point and depth $L \geq 2$, $\mathcal{M} = 0$ and the reference is zero. A nonzero finite factorisation of a target matrix can exist there, but finding it solves a different, nonlinear constraint problem. The least-squares reference does not escape points where its own derivative vanishes.

C Numerical approximation and certification

C.1 Truncation and damping

In exact arithmetic, zero-initialised LSQR searches a nested sequence of Krylov spaces

$$\mathcal{K}_k = \operatorname{span}\{M^*b, (M^*M)M^*b, \dots, (M^*M)^{k-1}M^*b\} \subseteq \operatorname{range}(M^*). \tag{41}$$

Its first nonzero iterate is a positive multiple of M^*b, so it has the gradient-descent direction. Its scalar coefficient is selected by the least-squares problem; a normalised-target, one-iteration training rule is therefore not ordinary gradient descent with a fixed learning rate. At a fixed M and b, the undamped least-squares residual cannot increase with k. If u_k is the exact minimiser on $\mathcal{K}_k$, Mu_k is the orthogonal projection of b onto $M\mathcal{K}_k$, and $\cos(Mu_k, r) = \|Mu_k\|/\|r\|$ when these vectors are nonzero. The resulting monotonicity is a fixed-problem statement. It does not imply monotonic diagnostic values across separately trained runs with different k.

With the convention in Equation (13), damping solves

$$u_\rho = \arg\min_u \left\{ \|Mu - b\|_{\mathcal{H}}^2 + \rho^2 \|u\|_{\mathcal{W}}^2 \right\}, \qquad \rho > 0. \tag{42}$$

The unique solution and its operator effect are

$$\begin{aligned} u_\rho &= (M^*M + \rho^2 I)^{-1} M^* b, \\ Mu_\rho &= \sum_j \frac{\sigma_j^2}{\sigma_j^2 + \rho^2} \langle b, \phi_j \rangle_{\mathcal{H}} \phi_j. \end{aligned} \tag{43}$$

The second expression is a filtered target, not an orthogonal projection. It approaches r as $\rho \downarrow 0$ for a fixed finite-dimensional problem. Its norm is no greater than $\|r\|$, but its direction can differ when singular values have different damping factors. Treating a damped solution as the exact reference requires a check that the quantities being reported are insensitive to the chosen damping.

The implementation uses $\rho = 10^{-7}\widehat{\sigma}$, where $\widehat{\sigma} = \|Mv\|_{\mathcal{H}}/\|v\|_{\mathcal{W}}$ for the vector obtained after one power iteration. For a nonzero v, this ratio is at most the largest singular value; one iteration need not estimate that value accurately. Under our average batch metric, the equivalent Euclidean augmented system stacks $B^{-1/2}M_{\mathrm{raw}}$ and ρI, where M_{raw} maps an increment to the unscaled vector of all operator entries. This scaling must also be used for the target and adjoint. If an implementation instead stacks M_{raw} and $\rho_{\mathrm{raw}} I$, the corresponding coefficient here is $\rho = \rho_{\mathrm{raw}}/\sqrt{B}$.

C.2 Target scaling in the trained control

The trained control normalises the batch of output residuals, rather than the operator gradient. With $g = (g_i)_{i=1}^B$ and $s_g = \max(\|g\|_F, 10^{-12})$, its target is

$$b_i^{\mathrm{Ref}} = -\eta \frac{g_i x_i^\top}{s_g}, \qquad \|b^{\mathrm{Ref}}\|_{\mathcal{H}}^2 = \frac{\eta^2}{B s_g^2} \sum_i \|g_i\|_2^2 \|x_i\|_2^2. \tag{44}$$

The guard gives a zero target when $g = 0$ and suppresses the normalised residual when its norm is below the threshold. Above that threshold, the normalised output-residual batch has unit Frobenius norm, but the requested operator norm still depends on the input norms and how residual mass is distributed across examples. Neither the operator target nor the realised parameter or operator motion has a fixed norm in general. At a fixed nonzero gradient, this normalisation is a common positive rescaling of the target, so it preserves the direction of the exact reference and the scale-invariant diagnostics. Across training, it changes the step sizes and hence the trajectory.

C.3 What residuals establish

The undamped normal-equation residual is $r_N = M^*(Mu - b)$. It measures least-squares stationarity. For the damped problem, the relevant residual is instead $r_\rho = M^*(Mu - b) + \rho^2 u$; the exact damped solution has $r_\rho = 0$ and $r_N = -\rho^2 u_\rho$. Reporting both distinguishes failure to solve the regularised system from the effect of regularisation itself.

Neither the plain residual nor r_N detects an added exact kernel component. With $M = [1\ 0]$, $b = 1$ and $u = (1, t)$, both residuals are zero for every t, although only $t = 0$ gives the minimum-norm solution. Retaining the iterate with smallest normal residual therefore cannot by itself certify minimum norm. Damping penalises such a component, while an explicit small-instance singular-value decomposition can check the minimum-norm property directly.

A small relative normal residual also need not imply a small operator error when the nonzero singular values are poorly conditioned. For example, take $M = \mathrm{diag}(1, \varepsilon)$, $b = (1, 1)$ and $u = (1, 0)$. The relative normal residual is $\varepsilon/\sqrt{1+\varepsilon^2}$, which tends to zero, but $\|Mu - \Pi b\| = 1$. A tolerance on r_N is consequently an operational quality check, not a condition-independent error guarantee for reachability, weight angles or kernel fractions.

An approximate vector $y = Mu$ still supplies rigorous lower bounds on reachability without identifying it with r. In exact arithmetic,

$$\alpha \geq \sqrt{\max\left\{0, 1 - \frac{\|b - y\|_{\mathcal{H}}^2}{\|b\|_{\mathcal{H}}^2}\right\}}, \qquad \alpha \geq \frac{|\langle b, y\rangle_{\mathcal{H}}|}{\|b\|_{\mathcal{H}}\|y\|_{\mathcal{H}}} \tag{45}$$

whenever the displayed denominators are nonzero. The first bound uses optimality of the projection residual; the second uses $y \in \mathrm{range}(M)$ and Cauchy–Schwarz. By contrast, $\|y\|/\|b\|$ is not a lower bound for an arbitrary unconverged numerical iterate. It has that interpretation for an exact restricted least-squares projection, or for the exact damped filter above, under their respective assumptions.

Numerical verification must therefore separate the adjoint identity, least-squares stationarity, minimum-norm behaviour and finite-step fidelity. They are different properties. In particular, calibrating zero kernel fractions for gradient descent at one set of checkpoints does not certify a projection at another optimiser's checkpoints, whose step map can have a different spectrum. Finite-step fidelity additionally requires evaluating the actual difference $(P_i(W + u) - P_i(W))_i$ and comparing it with Mu, because no linear-system residual measures gate changes.

C.4 Numerical checks and their scope

Comparison with an explicit SVD. The small-network checks report agreement between the iterative solution and a dense SVD pseudo-inverse to within $1.3 \cdot 10^{-4}$. They also report one ill-conditioned case with $\sigma_{\min}/\sigma_{\max} = 1.8 \cdot 10^{-5}$, in which the discrepancy is 2.6% in operator space and 65% in parameter space. This example illustrates that a comparatively small operator discrepancy can coexist with a much larger error in its parameter representative. It does not establish a uniform accuracy guarantee for reachability or either cosine: these quantities also depend on the target, projected-target norm and candidate direction.

Configuration: random ReLU, leaky-ReLU and CReLU networks of depth 4, width 10, input dimension 12 and 4 outputs, with 6 Gaussian probe inputs (360 parameters; 600 for CReLU), all in float64. The step map is formed densely and compared with its undamped Moore–Penrose pseudo-inverse (singular values below $10^{-10}\sigma_{\max}$ discarded) against the damped LSQR solve used in the paper (5000 iterations, $\rho = 10^{-7}\widehat{\sigma}_{\max}$). Errors are relative Euclidean norms: $\|u - u^{\mathrm{ref}}\|/\|u^{\mathrm{ref}}\|$ in parameter space and $\|M(u - u^{\mathrm{ref}})\|/\|Mu^{\mathrm{ref}}\|$ in operator space. The ill-conditioned leaky case is a different random draw of the same configuration; its discrepancy is unchanged over pseudo-inverse cutoffs from 10^{-10} to 10^{-5}, so it reflects incomplete convergence along small singular directions rather than damping. The implemented adjoint satisfies $M^*G = \nabla_W \mathcal{L}$ for the batch-mean loss to relative error $\leq 4 \cdot 10^{-16}$ in all five architectures.

Reachability and unresolved probes. The inner-product bound in Equation (45) does not require convergence of the candidate solve. Applied at the ReLU Adam checkpoint, it gives lower estimates of .761, .839 and .870 at 3000, 10000 and 30000 iterations, respectively (Appendix D.2). These are lower bounds on the exact projection ratio when evaluated with the stated step map and inner product, subject to numerical evaluation error; they are not estimates of the remaining gap to one. In particular, increasing the solve budget can improve a certificate without establishing convergence of the projected direction.

Stateful measurements now use exact replay: each run is re-executed from initialisation with its own seed, batch stream and update rule up to its fitting snapshot, which reproduces the stored

snapshot weights exactly (relative difference 0 in every run) and supplies the true optimiser state. At those states, Adam's $\widehat{c}_{\mathrm{op}}$ is .033/.035/.066 at 3000 iterations and .030/.035/.066 at 30000 (ReLU/CReLU/leaky); the guaranteed reachability bound rises from .799 to .874 on ReLU and is 1.000 on the other two; and 3/5, 5/5, 3/3 seeds pass $\mathrm{ne} < 10^{-2}$ at 30000 iterations. Adam's kernel fraction changes by at most .014 between the two budgets. Probes exceeding the threshold are retained and flagged; the two remaining ReLU seeds have $\mathrm{ne} \approx 2 \cdot 10^{-2}$.

Exact replay checks state reconstruction independently of projection accuracy. Stability across the two budgets is evidence of numerical consistency, but does not give a condition-independent error bound for the kernel fractions. The reference solve uses target b, whereas estimating q requires a separate solve with target Mu.

Local derivatives and finite training steps. The first finite-step check used 64 probe inputs; the table below uses the standard 100-input probe. At the tested trained checkpoints, the control changes at least one gate on $63/64$ ReLU inputs and $64/64$ CReLU inputs. The corresponding counts for SGD are $4/64$ and $16/64$. The control therefore crosses a gate boundary on almost every tested input; the SGD counts are substantially smaller. These counts concern inputs with a changed gate pattern, not the fraction of individual gates that change.

Table 3 replays each rule to its fitting snapshot (two seeds), applies its next step u on the 100-input probe batch, and measures the alignment of Mu and $\Delta P = P(W+u) - P(W)$ with the same estimated projected target $\widehat{r}$. This target is obtained from a 3000-iteration damped solve. The reported means with gate changes have smaller realised than first-order cosines. Reference-solve residuals were not recorded, and neither $\cos(\Delta P, Mu)$ nor $\|\Delta P - Mu\|/\|Mu\|$ is reported for these snapshots. The table therefore establishes a difference in alignment with the estimated reference, while the reference accuracy and direct linearisation error remain unquantified.

To separate the sources of this discrepancy, write $P(V; D(W))$ for the sampled operators evaluated with weights V while retaining the gates at W, on the same probe inputs. Define $\Delta P_{\mathrm{fix}} = P(W+u; D(W)) - P(W; D(W))$. Then

$$\Delta P - Mu = \underbrace{\Delta P - \Delta P_{\mathrm{fix}}}_{\text{effect of gate changes at updated weights}} + \underbrace{\Delta P_{\mathrm{fix}} - Mu}_{\text{higher-order fixed-gate terms}} .$$

With W and its original gates fixed, the second term is $O(\|u\|_{\mathcal{W}}^2)$ as $u \to 0$, since the frozen-gate operator is polynomial in the weights. The first term isolates the effect of replacing the original gates by the updated gates at $W+u$. The reported measurements do not resolve these terms separately. For REF, gates change on 94% of ReLU probe inputs and 96% of CReLU probe inputs. This makes a gate contribution plausible, but the frequency of gate changes does not quantify their effect on the operators or establish their dominance.

The separate random-network check without gate changes tests the local limit: $\|\Delta P - tMu\|/\|tMu\|$ decreases linearly with the step scale t from 10^{-3} to 10^{-7}. Per-seed finite-step values and relative step sizes $\|u\|/\|W\|$ are listed in the code repository. The scaling check supports the first-order expansion in its tested regime; attributing the trained-step discrepancy requires the two components above.

Table 3: Local versus realised operator change at the fitting snapshot, mean of two seeds: cosine with the estimated projected target $\widehat{r}$ of the first-order change Mu and of the realised change ΔP, and the fraction of probe inputs on which at least one gate changes.

	ReLU			CReLU		
rule	$\cos(Mu, \widehat{r})$	$\cos(\Delta P, \widehat{r})$	flipped	$\cos(Mu, \widehat{r})$	$\cos(\Delta P, \widehat{r})$	flipped
SGD	.225	.001	.06	.245	.002	.24
REF	.887	.028	.94	.954	.036	.96
Adam	.017	.017	.00	.029	.000	.37
heavy ball	.180	.048	.07	.131	.065	.01
Muon	.034	.001	.40	.038	.001	.45
SOAP	.082	.002	.10	.015	.000	.37
Shampoo	.282	.001	.06	.397	.001	.07

D EXPERIMENTAL PROTOCOL AND SUPPORTING RESULTS

The experiments diagnose optimiser behaviour through update geometry, gating and the evolution of operators. The exact reference M^+b is distinct from the *reference-following control* (Ref) used in training. Unless specified otherwise, Ref normalises its target and takes $k = 50$ LSQR iterations. The minibatch baseline is called SGD; the linear experiment uses full-batch gradient descent.

Data and training. MNIST-1D (Greydanus & Kobak, 2024) has input dimension 40 and 10 classes. We use `mnist1d` version 0.0.2.post1 with its default generator arguments: 5000 examples and train fraction 0.8. The run seed also seeds data generation. A seeded permutation of the 4000 generator-training examples gives 3000 training and 1000 validation examples; the generator's 1000 test examples form the test set. One scalar mean and standard deviation, computed on the training split, standardise the inputs. Thus variation across seeds includes both data and training variation. Table 4 gives the SHA-1 prefixes of the train/validation/test splits.

Table 4: The SHA-1 prefixes for the MNIST-1D training, validation and test splits generated for each of the five principal data seeds.

seed	training	validation	test
0	`37eadb33`	`fda6a0cc`	`8fc4980d`
1	`6991848b`	`96598fbb`	`69e7ffa9`
2	`b1461b3b`	`863f9459`	`b803f7af`
3	`a0bf97a1`	`40bb77ff`	`d492e4cf`
4	`cb8a2114`	`ebcc9e0b`	`397eb7c0`

The principal networks have depth 8, width 128, no biases and ReLU, CReLU or leaky-ReLU activations; the leaky slope is 0.1. Width counts pre-activation units. Thus $W_1 \in \mathbb{R}^{128\times 40}$, $W_2, \dots, W_7 \in \mathbb{R}^{128\times 128}$ and $W_8 \in \mathbb{R}^{10\times 128}$ for ReLU and leaky ReLU. CReLU doubles each hidden post-activation representation, giving shapes 128×256 for $W_2, \dots, W_7$ and 10×256 for W_8. Weights are sampled from $\mathcal{N}(0, 2/\text{fan-in})$. Training minimises cross-entropy for 12000 updates in float32, using batches of 100 sampled with replacement from a generator seeded by the run seed. Numerical probes use float64. Principal ReLU and CReLU comparisons use five seeds; leaky ReLU uses three. Uncertainties are standard deviations across seeds unless stated otherwise.

Targets and optimiser settings. The training target is

$$b_i = -\eta \, \frac{g_i}{\max(\|g\|_F, 10^{-12})} \, x_i^\top,$$

where g stacks the whole batch of output residuals. The normaliser is not a norm of the operator target $G_i = g_i x_i^\top$. Diagnostics use the unnormalised target. Training LSQR starts from zero and takes exactly k iterations. Normalising the target fixes neither the weight-step norm nor the realised operator-step norm, which also depend on the inputs, current step map, reachability and truncation.

The SGD learning-rate grid is $\{0.003, 0.01, 0.03, 0.1\}$, with 0.1 omitted for CReLU. Adam uses $\{10^{-3}, 3\cdot 10^{-3}, 10^{-2}, 3\cdot 10^{-2}\}$, $\beta = (0.9, 0.999)$ and $\epsilon = 10^{-8}$. Ref uses $\eta \in \{0.1, 0.3, 0.5, 1\}$ at $k = 50$. There are no schedules, weight decay or clipping. Target size η is selected in the same way as a learning rate. The principal settings are SGD 0.1, Adam 0.003 and Ref $\eta = 1$ for ReLU and leaky ReLU; CReLU uses SGD 0.03, Adam 0.003 and Ref $\eta = 1$.

Measurement phase and selection. Cheap probes occur every 250 updates. The fitting anchor is the first probe with training accuracy at least 0.999. The convergence-phase score averages the first eight probes beginning at that anchor, or all remaining probes if fewer than eight remain. A run that never reaches the threshold has no phase score. Hyperparameters are selected by validation accuracy, averaging first within this window and then over seeds that fit; test accuracy is reported after selection.

Alignment probes occur every 2000 updates; alignment is read at the first such probe at or after the anchor, rather than averaged over eight cheap probes. Statistics requiring stored weights use the first saved snapshot at or after the anchor. Checkpoints are saved every 4000 updates. Endpoint

values are either final-checkpoint measurements or last-quarter means, as stated for each comparison. All principal seeds reach the threshold. Mean anchor steps in SGD/Adam/Ref order are $1050/7250/1600$ for ReLU, $1000/3750/1750$ for CReLU and $1000/5500/1167$ for leaky ReLU. SGD and Ref remain at training accuracy at least 0.998 in the window. Adam can fall back to 0.971 on ReLU and 0.804 on one CReLU seed. The anchor therefore marks a threshold crossing, not sustained interpolation. Snapshot steps are per seed: 4000 for every SGD and REF seed; for Adam $(12000, 12000, 8000, 8000, 8000)$ on ReLU, $(8000, 4000, 4000, 4000, 8000)$ on CReLU and $(8000, 4000, 8000)$ on leaky ReLU, whose means are the 9600, 5600 and 6667 quoted elsewhere. Appendix D.4 also compares trajectories at matched training loss.

Numerical probes and structural statistics. The principal alignment probe uses 100 fixed training inputs, with the same indices across runs of a seed. It solves

$$\min_u \ \|Mu - b\|^2 + \rho^2\|u\|^2, \qquad \rho = 10^{-7}\widehat{\sigma}_{\max},$$

using the augmented system $[M; \rho I]$ with the metric convention in Appendix C. The scale $\widehat{\sigma}_{\max}$ is $\|Mv\|/\|v\|$ after one power step, a lower bound on the largest singular value rather than an exact value. It stops after 3000 iterations or when the damped normal residual falls below 10^{-10} of its initial value, and returns the iterate u_ρ with the smallest such residual. The recorded undamped residual is

$$\mathrm{ne} = \frac{\|M^*(Mu_\rho - b)\|}{\|M^*b\|}.$$

Probes with $\mathrm{ne} > 10^{-2}$ are retained and flagged. This residual measures proximity to the undamped normal equations; it is not the optimality residual for the damped objective. A small residual alone does not certify minimum norm in an ill-conditioned problem. The implementation also records the norm ratios $\|Mu\|/\|r\|$ and $\|u\|/\|u^{\mathrm{ref}}\|$; damped residuals, absolute solution norms and stopping indices were not recorded. Appendix C gives the small-instance comparison with dense SVD.

The null fraction is $q = \|u - M^+Mu\|/\|u\|$, a norm fraction, with projector solves using relative damping 10^{-10}. Stateful q is measured at the exactly replayed state (Appendix D.3). Gate density is the fraction of input–unit pairs in the positive-pre-activation state. Gate diversity averages the fraction of differing gate states between inputs i and $i + 32$ over 32 pairs of a 64-input cheap probe; this is distinct from the 100-input alignment probe and from the per-unit switching statistic below. Gate churn is the fraction of states changed since the previous probe. A ReLU unit is dead on a probe set if it is off on every probe input; a constant nonpositive gate for leaky ReLU or CReLU need not be dead. For a matrix D of vectorised operators and row mean $\bar{D}$, the shared component is $\|\mathbf{1}\bar{D}\|_F^2/\|D\|_F^2$.

D.1 LINEAR OPERATOR TRAJECTORIES

The linear experiment uses a bias-free width-32 chain from two inputs to one output, so its operator lies in $\mathbb{R}^{1\times 2}$. The 128 examples have feature scales $(1, 0.35)$ and targets $y = Xw$ for one sampled w. Every rule uses the full batch and mean squared loss. Runs use float64, 300 steps, depths 2 and 32, and Xavier or orthogonal initialisation. The eight rules are gradient descent, Adam, the reference, Muon, Shampoo, SOAP, K-FAC and heavy ball. Learning rates span 10^{-5} to 0.1; the reference has the separate target-size grid $\{0.03, 0.1, 0.3, 1\}$ and uses an unnormalised target with a converged damped solve. Grids by rule: gradient descent $\{1, 3\} \times 10^{-5\cdots-2}$ and 0.1; Adam, Muon and Shampoo $\{1, 3\} \times 10^{-5\cdots-2}$; SOAP, K-FAC and heavy ball $\{1, 3\} \times 10^{-5\cdots-3}$ and 10^{-2}; coefficients as in Appendix D.17. The reference solve uses 400 LSQR iterations with $\rho = 10^{-7}\widehat{\sigma}_{\max}$ and stops early when the damped normal residual falls below 10^{-10} of its initial value.

For each arm in Figure 2, the displayed run has the shortest total operator-path length among runs whose final loss is within twice that arm's best final loss. All eight arms have final operator error $\|P - P^\star\| \le 0.003$ in every panel; the reference error rounds to 0.0000. The selected Adam learning rate changes from 10^{-3} to $3 \cdot 10^{-5}$ between shallow and deep Xavier networks, whereas the reference uses $\eta = 0.3$ in all four configurations. Thus the comparison can distinguish routes to nearby endpoints without equating those routes.

Table 5 reports the fraction of finite steps with negative realised alignment $\cos(\Delta P, -\partial\mathcal{L}/\partial P)$ over five seeds in a separate width-8 depth series; brackets give the per-seed range.

Operator-space trajectories, linear chain, width 32. Dot = initialisation, star = OLS solution; axes scale per panel.

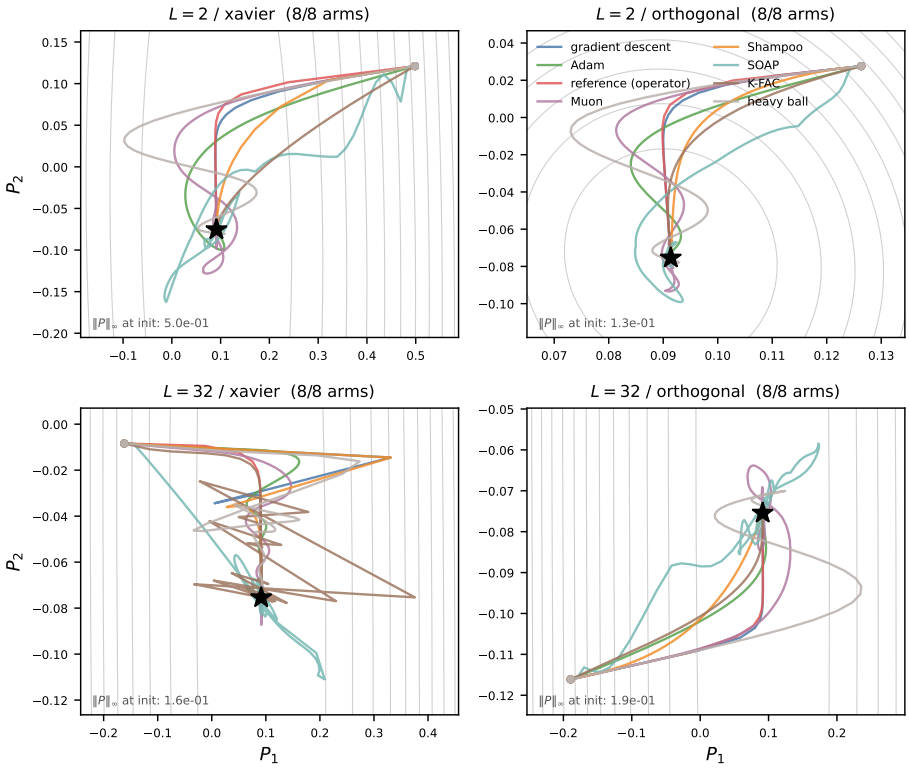

Figure 2: Linear operator trajectories for depths 2 and 32 with Xavier or orthogonal initialisation. Each arm shows its shortest operator path among runs finishing within twice that arm's best final loss. All endpoints lie within 0.003 of the target operator. Different routes can therefore reach nearby endpoints. Settings are given in the text.

Table 5: Fraction of finite steps with negative realised operator alignment in the separate width-8 linear depth series. Entries average five seeds, except gradient descent at depths 8 and 32, where two seeds diverge and are excluded ; brackets give the per-seed range. These are finite-step measurements, rather than frozen-derivative cosines.

depth	GD	Adam	reference
$L = 2$	0.3% $[0, 2]$	9.3% $[0, 22]$	0.0%
$L = 8$	1.0% $[0, 3]$	33.0% $[28, 37]$	0.0%
$L = 32$	0.0% $[0, 0]$	26.9% $[15, 37]$	0.1%

Adam has more negatively aligned finite steps at depths 8 and 32 than gradient descent or the reference. These rates include momentum, higher-order product terms and numerical effects; they do not by themselves rank optimisation quality. He initialisation is excluded from the trajectory panels because the deep linear product becomes large, with infinity norm 1.1×10^4 at depth 32. A separate depth-6, width-8 linear step map has nullity 83%. Both observations are specific to their configurations.

D.2 Reachability and solver sensitivity

Principal reachability estimates round to 1.000 for SGD and Ref in all three architectures. Adam's estimates are 0.770 on ReLU, 1.000 on CReLU and 0.993 on leaky ReLU, but many of its reference

solves fail the numerical threshold. Table 6 reports the effect of increasing the solver budget at one ReLU Adam checkpoint.

Table 6: Solver-budget sensitivity at one ReLU Adam checkpoint. The cosine row evaluates SGD at those same weights; the reachability bounds do not certify the projected direction or minimum-norm lift.

LSQR iterations	3000	10000	30000
ne	$8.4 \cdot 10^{-2}$	$3.7 \cdot 10^{-2}$	$1.5 \cdot 10^{-2}$
estimated $\|Mu_\rho\|/\|b\|$	0.761	0.839	0.870
SGD's c_{op} at the same weights	0.133	0.121	0.116
certified lower bound on α	0.761	0.839	0.870

For any nonzero $y = Mu$, the range condition gives the bound $\alpha \geq |\langle b, y\rangle|/(\|b\|\|y\|)$, independently of solver convergence. The last row therefore supplies the lower bound 0.870 at this checkpoint even though the residual remains above 10^{-2} at 30000 iterations. This lower bound does not certify the estimated projection direction or the minimum-norm lift. The cosine row evaluates the gradient-descent direction at Adam's weights, not Adam's stateful step.

All principal SGD and Ref seeds pass $\mathrm{ne} < 10^{-2}$. For Adam, only $0/5$ ReLU, $1/5$ CReLU and $0/3$ leaky-ReLU seeds pass at 3000 iterations; with exact state replay and 30000 iterations, $3/5$, $5/5$ and $3/3$ pass, and the estimated alignment remains small (Appendix C). The CReLU bound rounds to 1.000; this is evidence of nearly complete reachability at the displayed precision, rather than a proof of exact surjectivity.

D.3 Directional alignment and stateful updates

Each optimiser is measured on its own trajectory. The comparison therefore describes the update rule together with the weights, gates and optimiser state it reaches. It does not isolate the update rule at common weights. All probes are retained; an asterisk marks an Adam cell containing reference solves with residual above 10^{-2}. Table 7 reports the corresponding geometric estimates.

Table 7: Estimated update geometry at each rule's first 100-input alignment probe at or after its fitting anchor. Displayed deviations are seed standard deviations; asterisks flag cells containing failed reference solves. Adam's replayed 30000-iteration values and all stateful kernel fractions are given in the text below.

architecture	arm	$\widehat{c}_{\mathrm{op}}$	$\widehat{c}_{\mathrm{w}}$	$\widehat{\alpha}$	ne
ReLU	SGD	0.228 ± 0.021	0.122 ± 0.018	1.000	$3.4 \cdot 10^{-4}$
	Adam*	0.032 ± 0.016	-0.002 ± 0.007	0.770 ± 0.102	$7.7 \cdot 10^{-2*}$
	Ref	0.861 ± 0.042	0.425 ± 0.094	1.000	$4.4 \cdot 10^{-4}$
CReLU	SGD	0.253 ± 0.021	0.272 ± 0.020	1.000	$8.9 \cdot 10^{-11}$
	Adam*	0.034 ± 0.012	0.066 ± 0.022	1.000	$3.7 \cdot 10^{-2*}$
	Ref	0.931 ± 0.025	0.732 ± 0.035	1.000	$1.4 \cdot 10^{-8}$
leaky	SGD	0.204 ± 0.009	0.132 ± 0.018	1.000	$2.0 \cdot 10^{-4}$
	Adam*	0.066 ± 0.012	0.012 ± 0.004	0.993	$2.2 \cdot 10^{-2*}$
	Ref	0.860 ± 0.016	0.473 ± 0.017	1.000	$1.3 \cdot 10^{-4}$

At the original 3000-iteration budget, all five ReLU Adam seeds, four of five CReLU Adam seeds and all three leaky-ReLU Adam seeds fail the threshold. The single-checkpoint budget test in Appendix D.2 evaluates SGD at Adam's weights and does not certify Adam's stateful cosines. The replayed Adam measurements below use the larger budget. For SGD on the probed batch and the exact reference, membership in $\mathrm{range}(M^*)$ gives $q = 0$ algebraically; the SGD and Ref projector checks round to 0.0000.

Stateful kernel fractions are measured at the exact optimiser state, recovered by replaying each run from initialisation to its fitting snapshot with its own seed, batch stream and rule; the replayed

weights match the stored snapshots exactly. With 30000-iteration solves and all seeds, Adam has $\widehat{q} = 0.742 \pm 0.031$ (ReLU), 0.557 ± 0.038 (CReLU) and 0.616 ± 0.012 (leaky), $\widehat{c}_{\mathrm{op}} = 0.030 \pm 0.014$, 0.035 ± 0.012 and 0.066 ± 0.012, and guaranteed reachability at least 0.874 ± 0.098, 1.000 and 1.000; the normal residuals are 0.014 ± 0.007, $(1.2 \pm 0.2) \cdot 10^{-10}$ and $(2.3 \pm 1.6) \cdot 10^{-4}$, against 0.073 ± 0.018, 0.040 ± 0.027 and 0.020 ± 0.012 at 3000 iterations. SGD and REF have $q = 0$ algebraically and measure 0.0000.

Ref's operator alignment varies with measurement phase. Convergence-phase, second-half and last-quarter means are $0.861/0.855/0.842$ on ReLU, $0.931/0.776/0.710$ on CReLU and $0.860/0.815/0.778$ on leaky ReLU. These changes concern the truncated, normalised training arm and the states it visits; they do not contradict the exact identity $MM^{+}b = \Pi b$.

D.4 PREDICTIVE PERFORMANCE AT MATCHED PHASE AND MATCHED LOSS

Table 1 reports the validation-selected phase comparison. Test accuracies in SGD/Adam/Ref order are $0.531/0.634/0.565$ on ReLU, $0.504/0.612/0.592$ on CReLU and $0.535/0.654/0.567$ on leaky ReLU. Seed deviations are respectively $0.010/0.013/0.013$, $0.016/0.071/0.011$ and $0.011/0.018/0.010$. Ref uses $\eta = 1, k = 50$ throughout. Its endpoint accuracies are 0.570, 0.587 and 0.583, respectively. The loss and accuracy measurements use the same examples, probes and seeds (Appendix E).

Table 8 reports test accuracy at the first crossing of each training-loss threshold, using linear interpolation between probes and then averaging over seeds. Per-entry seed deviations lie in the range 0.007–0.023, except for CReLU Adam at losses at most 10^{-2}, where one oscillating seed gives deviations near 0.07.

Table 8: Test accuracy at the first crossing of each training-loss threshold, linearly interpolated between probes and averaged over seeds. No CReLU SGD seed reaches 10^{-4}, so that entry is undefined.

ReLU, at training loss	10^{-1}	10^{-2}	10^{-3}	10^{-4}
SGD	0.518	0.529	0.531	0.532
Adam	0.566	0.628	0.637	**0.641**
Ref	0.528	0.548	0.560	0.575
CReLU, at training loss	10^{-1}	10^{-2}	10^{-3}	10^{-4}
SGD	0.495	0.503	0.508	—
Adam	0.641	0.620	0.609	**0.613**
Ref	0.554	0.566	0.587	0.592
leaky, at training loss	10^{-1}	10^{-2}	10^{-3}	10^{-4}
SGD	0.524	0.540	0.535	0.533
Adam	0.594	0.637	0.652	**0.652**
Ref	0.522	0.557	0.562	0.585

Every seed reaches every listed threshold except CReLU SGD at 10^{-4}, which none of its five seeds reaches. Adam has the highest test accuracy at all reported thresholds. Ref generally lies between SGD and Adam, with a small exception on leaky ReLU at loss 10^{-1}: SGD scores 0.524 and Ref 0.522. At the principal phase, the Ref–SGD gaps are 3.4, 8.8 and 3.2 percentage points for ReLU, CReLU and leaky ReLU. These comparisons show that the accuracy ordering is not solely due to evaluating methods at different training losses; they remain conditional on this task, architecture and selection protocol.

D.5 NONPARAMETRIC AND LINEAR BASELINES

Across five data seeds, Euclidean k-nearest neighbours on the inputs gives test accuracy 0.559 at $k = 1$, 0.562 at $k = 5$, 0.573 ± 0.015 at $k = 10$ and 0.473 at $k = 50$. The grid also contains $k = 3$ and 20; validation selects $k = 10$. Multinomial logistic regression, with its L_2 penalty selected on validation from $\{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ (coefficient on $\|W\|_F^2$, L-BFGS; selected 10^{-3}), scores 0.333 ± 0.016. Ref's principal scores of 0.565–0.592 are close to the selected nearest-neighbour

score. This comparison does not establish memorisation or a particular representation mechanism. Approximate published MLP and CNN scores of 0.68 and 0.94 (Greydanus & Kobak, 2024), obtained on the original MNIST-1D release, provide context only, since they are not matched to these generated splits.

D.6 Truncation sweep

The main sweep varies $k \in \{1, 2, 5, 20, 50, 200\}$ at $\eta = 1$ on ReLU and CReLU, using three seeds per setting and five at $k = 50$. CReLU also includes $\eta \in \{0.3, 3, 10\}$ at $k \leq 50$, with one or two seeds; the principal $k = 50$ grid additionally supplies $\eta = 0.1$ and 0.5. All reference probes pass the numerical threshold. At $\eta = 1$:

Table 9: Reference-following truncation sweep at $\eta = 1$: three seeds per setting, five at $k = 50$. Phase quantities are unavailable when no seed fits; only one of three ReLU seeds fits at $k = 2$. Its phase score uses the single fitting seed.

k (ReLU)	1	2	5	20	50	200
test accuracy	—	0.465	0.486	0.564	**0.565**	0.528
train accuracy	0.879	0.993	1.000	1.000	1.000	1.000
c_{op}	—	0.135	0.365	0.800	0.861	0.948
operator PR	—	5.69	7.20	7.27	6.55	7.13
k (CReLU)	1	2	5	20	50	200
test accuracy	0.478	0.487	0.528	**0.602**	0.592	0.558
train accuracy	1.000	1.000	1.000	1.000	1.000	1.000
c_{op}	0.140	0.237	0.562	0.864	0.931	0.992
operator PR	5.99	6.12	7.47	7.22	7.16	7.71

Increasing the truncation budget improves alignment across these settings, but the highest alignment does not give the highest test accuracy. On CReLU, all settings fit: accuracy is 0.602 at $k = 20$, 0.592 at $k = 50$ and 0.558 at $k = 200$, while alignment increases from 0.864 to 0.931 and 0.992. This separates alignment from predictive performance within the trained reference-following family. ReLU $k = 1$ has no fitting seed, and only one of three seeds fits at $k = 2$; those cases do not support the same generalisation comparison.

At a fixed state, the $k = 1$ LSQR direction is parallel to the gradient, but its scalar step and target normalisation need not reproduce SGD training. The phase participation ratios in Table 9 range from 5.69 to 7.71; the endpoint range is 1.50–7.76. The interpretation therefore depends on reading time as well as truncation. In the CReLU $k = 50$ cells, read at the first alignment probe after fitting, alignment is 0.989, 0.984, 0.985, 0.931, 0.702 and 0.493 for $\eta = 0.1, 0.3, 0.5, 1, 3, 10$ with $5, 5, 5, 5, 2, 1$ seeds. Per-cell test deviations are $\pm.009$, $.004$, $.004$, $.009$, $.011$, $.005$ on CReLU ($k = 1, \ldots, 200$) and $\pm.001$, $.009$, $.013$, $.009$ on ReLU ($k = 5, 20, 50, 200$). The ReLU $k = 2$ phase score 0.465 comes from its single fitting seed; the training-accuracy row gives the mean over seeds of the maximum training accuracy (0.985, 0.996, 1.000 at $k = 2$).

D.7 Input dependence of the gates

Gate states are evaluated on 256 training inputs at the first saved snapshot at or after the fitting anchor. The fractions in Table 10 pool all hidden units. A switching gate takes both states on this finite set; the other categories indicate constant positive or constant nonpositive pre-activation. For leaky ReLU and CReLU, the latter category does not imply a dead unit.

Table 10: Gate-state fractions across 256 training inputs at the first saved snapshot at or after fitting, pooling all hidden units. Switching denotes variation across inputs within that snapshot; constant nonpositive gates need not be dead for leaky ReLU or CReLU.

architecture	arm	switching	always on	always off	test acc.
ReLU	SGD	0.996	0.000	0.004	0.531
	Adam	0.915	0.000	0.085	0.634
	Ref	0.715	0.143	0.143	0.565
leaky	SGD	0.999	0.000	0.001	0.535
	Adam	0.995	0.000	0.005	0.654
	Ref	0.794	0.107	0.099	0.567
CReLU	SGD	0.999	0.000	0.000	0.504
	Adam	0.997	0.002	0.001	0.612
	Ref	0.711	0.142	0.147	0.592

Ref has fewer switching gates than both baselines in all three architectures. Adam also has a substantial constant-gate fraction on ReLU, so the baselines should not both be described as keeping every gate switching. The leaky-ReLU comparison shows that this difference can occur when activation slopes never vanish. It does not establish why the test accuracies differ. At 256 inputs, Ref's switching-fraction deviations are 0.056 for ReLU, 0.010 for leaky ReLU and 0.017 for CReLU; the baseline deviations are at most 0.015.

Probe size changes the classification of a gate as switching. At 256, 1024 and all 3000 training inputs, CReLU Ref has fractions 0.711 ± 0.017, 0.752 ± 0.015 and 0.783 ± 0.017; ReLU Ref gives $0.715/0.758/0.791$ and leaky Ref $0.794/0.829/0.856$. SGD remains at least 0.996. From the selected snapshot to the endpoint, Ref changes from 0.715 to 0.723 on ReLU, 0.711 to 0.720 on CReLU and 0.794 to 0.758 on leaky ReLU. Switching is a per-unit statistic across inputs, whereas the paired-input gate-diversity statistic in Appendix D is a different summary.

D.8 Operator participation ratio and diversity

For singular values σ_j, the participation ratio (PR) is $(\sum_j \sigma_j)^2 / \sum_j \sigma_j^2$. Pointwise operator PR averages this quantity over 512 training inputs. Zero operators are excluded by definition; none occurs in the cells or snapshots. Operator diversity is the PR of the matrix obtained by centring the rows of D, where $D \in \mathbb{R}^{512\times 400}$ stacks $\operatorname{vec} P(x_i)$. The shared component is $\|\mathbf{1}\bar{D}\|_F^2/\|D\|_F^2$. Diversity has rank ceiling $\min(511, 400) = 400$ and depends on the input set; it is not an intrinsic network dimension.

The first measurement in Table 11 uses the saved snapshot after the fitting anchor, rather than the anchor itself. This is step 4000 for SGD and Ref. Adam's per-seed snapshot steps are listed in Appendix D. The final measurement uses the last checkpoint.

Table 11: Full-logit operator PR, across-input diversity and shared component on 512 training inputs. The selected snapshot follows the fitting anchor; the final measurement uses the last checkpoint. The percentage column gives diversity relative to initialisation.

		at the selected snapshot				at the final checkpoint		
architecture	arm	PR	diversity	% init.	shared	PR	diversity	shared
ReLU	initialisation	—	241.2	100%	—	—	—	—
	SGD	5.72	$\mathbf{207.3 \pm 2.3}$	**86%**	0.068	5.75	206.3	0.069
	Adam	1.98	161.7 ± 11.3	67%	0.014	1.98	161.4	0.012
	Ref	**6.38**	191.5 ± 33.5	79%	0.027	3.29	109.8	0.025
CReLU	initialisation	—	266.3	100%	—	—	—	—
	SGD	6.58	$\mathbf{212.9 \pm 4.4}$	**80%**	0.117	6.53	210.5	0.114
	Adam	2.91	96.7 ± 19.1	36%	0.010	3.03	104.9	0.010
	Ref	**6.80**	199.6 ± 34.3	75%	0.018	1.79	45.4	0.023
leaky	initialisation	—	232.9	100%	—	—	—	—
	SGD	6.01	197.2 ± 4.9	85%	0.091	6.03	195.8	0.093
	Adam	3.89	192.3 ± 4.9	83%	0.027	3.91	193.3	0.027
	Ref	**6.85**	$\mathbf{203.6 \pm 18.3}$	**87%**	0.027	2.70	78.3	0.013

At the selected snapshot, Ref has the highest pointwise PR in all three architectures, but the highest diversity only on leaky ReLU. Its raw operator statistics then change substantially, whereas SGD and Adam change less between their selected and final snapshots.

These are comparisons from each rule's selected snapshot to its endpoint. They do not use a common starting step: two ReLU Adam seeds already have their selected snapshot at step 12000. Over the common 4000–12000 interval, Adam's PR changes from 2.79 to 1.98 and diversity from 190.6 to 161.4 (Tables 12 and 13).

The ReLU PR trajectory in Table 12 uses the 64-input cheap probe and five seeds, rather than the 512-input snapshot statistic. It is therefore a distinct measurement series.

Table 12: ReLU pointwise operator PR from the 64-input cheap probe, averaged over five seeds, with Ref training loss at the listed updates. This series is distinct from the 512-input snapshot measurements.

step	0	1000	2000	4000	8000	12000
SGD	7.02	5.60	5.69	5.70	5.71	5.72
Adam	7.02	3.86	3.42	2.79	2.02	1.98
Ref	7.02	**6.93**	**6.62**	**6.37**	5.31	3.28
Ref, train loss	3.1	$1.7 \cdot 10^{-2}$	$2.1 \cdot 10^{-3}$	$6.7 \cdot 10^{-7}$	$1.0 \cdot 10^{-7}$	$2.7 \cdot 10^{-6}$

The baseline diversity trajectories are given in Table 13:

Table 13: ReLU baseline diversity trajectories and mean fitting anchors. Ref's snapshot values are in Table 11.

ReLU, diversity at step	0	4000	8000	12000	mean anchor
SGD	239.8	207.3	206.5	206.4	1050
Adam	241.6	190.6	161.7	161.4	7250

Across the broader sweep, PR declines after fitting in 54 of the 55 fitting width-128, $k = 50$ Ref runs (the exception is the single CReLU $\eta = 10$ run, which fits only at update 7750), with median lag 3250 steps; five of six $k = 200$ runs do not show this decline. A decline is counted when the 64-input cheap-probe PR first falls one unit below its value at the fitting anchor; the lag is the number of updates from the anchor to that probe.

A separate single-run ReLU control at $\eta = 0.5$ compares an unnormalised target with the normalised training arm at step 12000. It reaches PR 6.27 and loss 6.4×10^{-4} for the raw target, versus PR

1.59 and loss 8×10^{-11} for the normalised target. The unequal losses prevent a conclusion about normalisation at matched optimisation progress.

Removing the common-logit direction. For $Q = I - \mathbf{1}\mathbf{1}^\top/10$, the class-centred operator $QP(x)$ removes the component that shifts every logit equally. Such shifts leave softmax probabilities unchanged. Table 14 applies the same statistics to $P(x)$ and $QP(x)$ on the same 512 inputs and snapshots. Subscript c denotes class centring; the diversity calculation still includes centring across inputs.

Table 14: Raw and class-centred operator statistics on the same 512 inputs and snapshots. Subscript c denotes $QP(x)$, which removes the common-logit direction; both diversity calculations additionally centre across inputs.

		selected snapshot				final checkpoint			
arch.	arm	PR	PR_c	div	div_c	PR	PR_c	div	div_c
ReLU	SGD	5.72	5.58	207.3	201.8	5.75	5.61	206.3	201.1
	Adam	1.98	2.03	161.7	158.4	1.98	2.03	161.4	159.2
	Ref	6.38	6.47	191.5	**221.0**	3.29	5.35	109.8	**218.8**
CReLU	SGD	6.58	6.34	212.9	205.9	6.53	6.33	210.5	204.2
	Adam	2.91	4.77	96.7	186.8	3.03	4.71	104.9	189.9
	Ref	6.80	6.65	199.6	**220.2**	1.79	5.70	45.4	**220.1**
leaky	SGD	6.01	5.85	197.2	191.5	6.03	5.87	195.8	190.4
	Adam	3.89	3.78	192.3	180.5	3.91	3.80	193.3	181.7
	Ref	6.85	6.57	203.6	**219.6**	2.70	5.58	78.3	**216.7**

Class centring changes the diagnosis. For ReLU Ref, raw PR falls from 6.38 to 3.29, while class-centred PR falls from 6.47 to 5.35. Raw diversity falls from 191.5 to 109.8, whereas class-centred diversity changes from 221.0 to 218.8. CReLU and leaky Ref show the same qualitative distinction: class-centred diversities change from 220.2 to 220.1 and 219.6 to 216.7. Thus the pronounced raw diversity decline does not establish comparable simplification of the operator governing class differences. A direct decomposition of the operator displacement between the two snapshots, $\Delta = P_{\text{end}} - P_{\text{fit}}$ on the same 512 inputs, gives the common-logit share $\|(I - Q)\Delta\|_F^2/\|\Delta\|_F^2$: for REF 0.86 ± 0.08 (ReLU), 0.98 ± 0.02 (CReLU) and 0.92 ± 0.05 (leaky); for SGD 0.01 in all three. Seed deviations of the class-centred statistics are: REF PR_c $6.47 \pm 0.19 \to 5.35 \pm 0.47$, $6.65 \pm 0.09 \to 5.70 \pm 0.10$, $6.57 \pm 0.18 \to 5.58 \pm 0.17$ and diversity_c $221.0 \pm 2.4 \to 218.8 \pm 3.2$, $220.2 \pm 1.1 \to 220.1 \pm 2.1$, $219.6 \pm 1.2 \to 216.7 \pm 3.5$ (ReLU/CReLU/leaky); SGD and Adam change by less than one standard deviation.

The displacement decomposition concerns the difference between the two snapshots. It does not measure cumulative path length or the sum of per-step squared displacements. Class-centred diversity remaining nearly constant also does not mean that the class-centred operators themselves remain unchanged.

D.9 LAYERWISE ACTIVATION VARIANCE

PCA is fitted separately for each of five seeds on 400 test inputs, using centred, unscaled post-activation representations at layers $1, 2, 4, 6, 8$. Layer 8 is the 10-dimensional logit output. CReLU uses post-activations consistently with the other architectures. Table 15 averages the percentage of variance explained by the first two principal components across seeds. Selected-snapshot and endpoint measurements are distinguished below. Each seed uses its own snapshot (Appendix D): step 4000 for every SGD and Ref seed, and for Adam steps $(12000, 12000, 8000, 8000, 8000)$ on ReLU and $(8000, 4000, 4000, 4000, 8000)$ on CReLU. The Adam times 9600 and 5600 are the means of these per-seed snapshot steps. The endpoint is step 12000 for every seed.

Table 15: Percentage of variance explained by the first two principal components of centred, unscaled post-activation representations. PCA uses 400 test inputs per seed; values average five seeds and distinguish the selected snapshot from the endpoint, with seed standard deviations in percentage points. Layer 8 contains raw logits.

		at the selected snapshot					endpoint	
architecture	arm	layer 1	layer 2	layer 4	layer 6	layer 8	layer 6	layer 8
ReLU	SGD	23±2	21±1	26±2	45±1	80±1	45±2	80±1
	Adam	21±1	17±1	41±2	**72±3**	**83±5**	74±6	83±6
	Ref	23±1	24±2	37±5	63±5	74±19	75±5	93±7
CReLU	SGD	25±1	22±1	25±1	42±2	86±1	43±2	85±1
	Adam	16±0	15±1	38±7	59±11	**95±2**	58±10	94±2
	Ref	20±1	21±1	31±2	60±2	69±16	77±3	99±0

Ref's first-two-component share at the ReLU output changes from 74% to 93%; on CReLU it changes from 69% to 99%. At the snapshot this share varies strongly across seeds (±19 and ±16 points); endpoint deviations are ±7 and ±0 points. Thus output variance becomes more concentrated in the first two components on average. SGD and Adam change by at most two points between their selected snapshots and endpoints. At ReLU layer 4, Adam has a larger share than SGD (41% versus 26%). CReLU Adam and Ref are close at layer 6 (59% versus 60%). These are variance-concentration observations on the stated representations, not evidence of class separation or exact intrinsic dimension. Output PCA here uses raw logits, so the distinction between logits and class-centred logits in Appendix D.8 also matters for its interpretation.

D.10 Partition density on data planes

For each of three trained seeds, a seeded draw supplies eight triples of training points from distinct classes. Each triple defines an affine plane. Coordinates put x_0 at $(0, 0)$ and x_1 at $(1, 0)$, with the second axis along the component of $x_2 - x_0$ orthogonal to $x_1 - x_0$; both axes use units of $\|x_1 - x_0\|$. A 240×240 grid covers $[-0.6, 1.6]^2$. A grid point is an edge if its full gate pattern differs from that of a horizontal or vertical neighbour. Density is the fraction of edge points in a distance bin, where distance is to the nearest of the three anchors. Concentration is mean density within 0.25 minus mean density beyond 0.75.

Optimisers are compared on the same planes. Table 16 preserves the plane-level descriptive means and standard deviations, but also gives the three training-seed contrasts obtained by averaging the eight planes of each seed. Planes sharing a network are not independent training replicates.

Table 16: Pairwise contrasts in partition-edge concentration near training anchors. Eight planes are nested within each of three trained seeds; seed contrasts and descriptive plane-level mean and standard deviation are reported separately.

architecture	pair	three seed contrasts	plane mean ± s.d.	same sign
ReLU	Ref − SGD	$(0.084, 0.074, 0.130)$	0.0960 ± 0.0406	100%
	Ref − Adam	$(0.061, 0.080, 0.084)$	0.0751 ± 0.0531	96%
	Adam − SGD	$(0.023, -0.006, 0.046)$	0.0209 ± 0.0485	71%
CReLU	Ref − SGD	$(0.089, 0.084, 0.095)$	0.0892 ± 0.0417	96%
	Ref − Adam	$(0.118, 0.087, 0.115)$	0.1067 ± 0.0433	100%
	Adam − SGD	$(-0.029, -0.003, -0.020)$	-0.0175 ± 0.0320	67%

All twelve seed-level Ref–SGD and Ref–Adam contrasts are positive. Adam–SGD is mixed on ReLU and negative on all three CReLU seeds. These are descriptive observations from three trained seeds, without a significance claim. The comparison concerns where sampled partition edges occur; it does not establish a cause of generalisation.

D.11 WIDTH SWEEP

CReLU widths $16, 32, 64, 128$ are compared across each optimiser's grid, with Ref fixed at $k = 50$. Hyperparameters are reselected on validation. The endpoint comparison in Table 17 allows cells that do not reach the fitting threshold; parentheses give the selected cell's highest training accuracy.

Table 17: Endpoint test accuracy across CReLU widths after validation selection, allowing nonfitting cells. Parentheses give the selected cell's highest training accuracy; fitting-phase selections are reported separately in the text.

width	SGD	Adam	Ref
16	0.374 (0.889)	0.476 (0.910)	0.420 (1.000)
32	0.515 (1.000)	0.478 (1.000)	0.494 (1.000)
64	0.475 (1.000)	0.635 (1.000)	0.562 (1.000)
128	0.511 (1.000)	0.613 (1.000)	0.589 (1.000)

For widths $32, 64, 128$, phase accuracies selected among fitting cells, in SGD/Adam/Ref order, are $0.519/0.484/0.495$, $0.472/0.640/0.566$ and $0.504/0.612/0.592$. At width 16, Ref scores 0.429 with two of three seeds fitting. Adam also fits at learning rate 0.003, with all three seeds fitting and phase accuracy 0.382; this differs from the endpoint-selected Adam cell. Only SGD has no fitting cell at width 16. Two width-16 Adam seeds have anchors at 11750 and 12000; the latter phase score uses only the final probe under the stated incomplete-window convention.

Selected cells use three seeds at width 16, one to three at widths 32 and 64, and five at width 128, except the endpoint-selected Ref cell, which uses two. These limited counts constrain comparisons. Selected cells (endpoint / phase): width 16: SGD 0.03 ($0/3$ fit) / none; Adam 10^{-2} ($0/3$) / $3 \cdot 10^{-3}$ ($3/3$); REF $\eta = 1$ ($2/3$) / same. Width 32: SGD 0.1 ($1/1$); Adam $3 \cdot 10^{-3}$ ($3/3$); REF $\eta = 1$ ($2/2$), both selections. Width 64: SGD 0.03 ($2/2$); Adam $3 \cdot 10^{-3}$ ($1/2$, phase uses the fitting seed); REF $\eta = 1$ ($2/2$). Width 128: SGD 0.03, Adam $3 \cdot 10^{-3}$ ($5/5$ each); REF $\eta = 3$ ($2/2$) / $\eta = 1$ ($5/5$). Ref exceeds Adam at width 32 under both endpoint (0.494 versus 0.478) and fitting-phase (0.495 versus 0.484) selection. Its endpoint accuracy increases with width in this sweep, so the results do not establish an advantage for smaller networks.

D.12 CHANGING THE OPTIMISER DURING TRAINING

This intervention tests how an initial period of Adam training affects a subsequent reference-following trajectory. On ReLU and CReLU, the handover occurs after $s \in \{0, 250, 1000, 3000, 6000, 12000\}$ updates, within a total budget of 12000 updates and three seeds. Adam uses learning rate $3 \cdot 10^{-3}$; REF uses $\eta = 0.5$ and $k = 50$. Test accuracy is the last-quarter mean. The results are given in Table 18:

Table 18: Adam-to-Ref schedules over 12000 updates and three seeds, evaluated by last-quarter mean test accuracy. The endpoints $s = 0$ and $s = 12000$ are pure Ref and Adam. In the reverse rows, s counts REF updates before the handover; Adam's moments and bias-correction counter start at the handover.

handover step s	0	250	1000	3000	6000	12000
ReLU, Adam then REF	0.548	0.554	0.574	0.601	0.606	0.645
ReLU, REF then Adam	0.645	0.641	0.631	0.620	0.560	0.548
CReLU, Adam then REF	0.561	0.581	0.645	0.660	0.609	0.596
CReLU, REF then Adam	0.596	0.658	0.613	0.655	0.637	0.561

At $s = 0$ the run uses only REF; at $s = 12000$ it uses only Adam. These endpoints use this experiment's three seeds, target size and last-quarter estimator, rather than the principal table's selected near-interpolation cells. On CReLU, the mixture with 3000 initial Adam steps has accuracy 0.660 ± 0.008, compared with 0.596 ± 0.070 for pure Adam and 0.561 for pure REF.

With 6000 updates of each rule, Adam first beats REF first on ReLU in all three paired seeds (0.606 versus 0.560; differences 0.034, 0.073, 0.031). On CReLU the order has no consistent effect (0.609

versus 0.637; differences 0.000, -0.096, 0.009). After a REF start, Adam recovers its usual accuracy given 9000 updates (0.620 ReLU, 0.655 CReLU). The CReLU mixture with 3000 Adam updates is not reliably above pure Adam: paired differences are 0.023, 0.173 and -0.003, the large one against an oscillating Adam seed. Order dependence at equal update counts is therefore moderate and specific to ReLU.

D.13 LONGER TRAINING

One ReLU seed per rule is continued from its own 12000-step checkpoint. The checkpoint stores weights and Adam's moments in float32; there is no learning-rate schedule. The continued settings are SGD 0.1, Adam $3 \cdot 10^{-3}$, and REF $\eta = 1$, $k = 50$. These source runs were selected using test accuracy, so the continuation is a descriptive analysis of selected trajectories. SGD and Adam reach 300000 steps, with Adam additionally continued to 10^6. The REF continuation was completed to 300000 steps. Table 19 compares the available snapshots.continuing

Table 19: Extended training of one test-selected ReLU trajectory per rule. Arrows connect the snapshots: all three rules are measured at 300000 updates; REF's switching fraction is from the 275000 snapshot.

Operator PR uses uncentred logits.

Rule	Snapshot interval	Test accuracy	Switching fraction	Uncentred operator PR
SGD	$25000 \to 300000$	$0.529 \to 0.527$	$0.993 \to 0.992$	$5.68 \to 5.72$
Adam	$25000 \to 300000$	$0.639 \to 0.477$	$0.925 \to 0.614$	$2.14 \to 1.00$
REF	$25000 \to 300000$	$0.526 \to 0.498$	$0.694 \to 0.655$	$1.28 \to 1.02$

SGD changes little over its interval. Adam reaches test accuracy 0.660 near 100000 steps, then falls to 0.477 at 300000 as training loss rises from approximately 10^{-9} to 0.69. At that common checkpoint SGD exceeds Adam, reversing their earlier accuracy ordering. Adam's million-step continuation reaches chance accuracy. At 300000 steps the ordering is SGD 0.527, REF 0.498, Adam 0.477. Its low uncentred participation ratio also does not by itself establish compression of the predictor, given the common-logit effect in Appendix D.8.

A loss spike is a probe whose training loss exceeds ten times the median over the preceding 20 probes. The Adam run contains 104 spikes, with mean gate churn 0.0348 during spikes versus 0.0058 otherwise, and test accuracy 0.620 versus 0.612. This association does not determine whether gate changes help or harm subsequent learning. No delayed accuracy transition (grokking) occurs in this trajectory.

D.14 MOMENTUM APPLIED TO WEIGHT INCREMENTS

This ablation asks how smoothing the reference-following control's parameter increments changes its local direction and predictive outcome. It uses

$$u_t = \beta u_{t-1} + (1 - \beta)\Delta W_t^{\mathrm{Ref}}, \qquad W_{t+1} = W_t + u_t.$$

Because the local step map M_t changes with the weights, this averages parameter increments rather than realised operator displacements. The factor $1 - \beta$ preserves the gain on a constant increment sequence, but does not fix the norm for a changing sequence. The grid covers ReLU and CReLU, $\eta \in \{0.5, 1\}$, $k = 50$, $\beta \in \{0, 0.5, 0.9, 0.99\}$ and three seeds, giving 48 runs. Table 20 reports paired accuracy differences from the same-seed $\beta = 0$ control, measured with the near-interpolation estimator of Appendix D, as mean $\pm$ seed standard deviation.

Table 20: Change in near-interpolation test accuracy when smoothing Ref parameter increments, with $k = 50$. Entries are paired differences from the same-seed $\beta = 0$ control, as mean $\pm$ standard deviation over three seeds.

architecture	η	$\beta = 0.5$	$\beta = 0.9$	$\beta = 0.99$
ReLU	0.5	-0.004 ± 0.005	-0.017 ± 0.007	-0.020 ± 0.010
	1	-0.013 ± 0.009	-0.006 ± 0.009	-0.075 ± 0.027
CReLU	0.5	-0.006 ± 0.008	-0.019 ± 0.010	-0.010 ± 0.010
	1	-0.014 ± 0.011	-0.021 ± 0.016	-0.037 ± 0.007

All twelve mean differences are nonpositive. The largest decreases occur at $\beta = 0.99$ and $\eta = 1$: 0.075 on ReLU and 0.037 on CReLU. Operator alignment falls from 0.80–0.88 at $\beta = 0$ to 0.09–0.30 at $\beta = 0.99$, and ReLU weight alignment falls from 0.34 to 0.03. All cells fit, with final loss at most 10^{-8}. This intervention changes the update direction without reproducing Adam's accuracy advantage in the tested configurations; it does not eliminate momentum as a contributor to Adam's behaviour. With exact state replay at the fitting snapshot (step 4000, $\eta = 1$, three seeds), the estimated kernel fraction of the smoothed step is 0.21 ± 0.06 (ReLU) and 0.32 ± 0.06 (CReLU) at $\beta = 0.5$, and 0.37 ± 0.01 and 0.60 ± 0.01 at $\beta = 0.9$, against 0 for plain REF. At $\beta = 0.9$ the estimates overlap the range reported for heavy ball (0.39 and 0.45), while the accuracy changes remain nonpositive. These are numerical kernel-fraction estimates with the interpretation described in Appendix D.17; the intervention does not isolate the effect of kernel motion on accuracy.

D.15 ADVERSARIAL EVALUATION

This experiment compares operator measurements with accuracy under a specified input perturbation. The attack takes 50 projected L_∞ steps of size $2.5\epsilon/50$, with uniform random starts and three restarts, maximising $\max_{j \neq y} z_j - z_y$. Inputs are standardised and clipped only to the ϵ-ball, without an additional box constraint. The recorded attack checks the final iterate of each restart; it does not retain the worst iterate along the trajectory. Attack strength has not been validated against stronger settings.

Table 21 uses a validation-selected fitting cell at its first saved convergence-phase snapshot, with up to three seeds. Each percentage is mean adversarial accuracy across seeds divided by mean clean accuracy. The margin is the mean of $z_y - \max_{j \neq y} z_j$ over clean test examples. Operator norms use the same snapshot, with endpoint norms in parentheses. These settings and seed counts explain why the clean accuracies need not coincide with Table 1.

Table 21: Attack-specific summaries at the first saved convergence-phase snapshot of validation-selected fitting cells, using up to three seeds. Percentages divide mean adversarial accuracy by mean clean accuracy for the stated final-iterate attack; parenthesised operator norms are endpoint measurements.

architecture	arm	clean	ϵ=0.05	ϵ=0.1	ϵ=0.3	margin	$\|P\|_F$
ReLU	SGD	0.531	58.5%	32.1%	0.6%	1.41	43.8
	Adam	0.642	65.8%	40.7%	3.7%	9.53	80.2
	Ref	0.573	67.7%	43.4%	4.4%	4.78	67.7 (237.4)
CReLU	SGD	0.517	57.0%	30.9%	1.2%	0.67	32.1
	Adam	0.572	46.5%	22.5%	0.3%	6.30	203.1
	Ref	0.600	69.4%	44.8%	5.6%	6.24	58.7 (390.4)
leaky	SGD	0.529	59.0%	32.0%	1.0%	1.34	44.9
	Adam	0.655	65.9%	39.8%	2.4%	6.76	43.3
	Ref	0.595	65.2%	42.2%	3.6%	5.54	63.6 (243.2)

The REF cells have the largest attack-specific robust/clean ratios among these three rules at each displayed radius on ReLU and CReLU. For example, the CReLU ratios at $\epsilon = 0.1$ are 30.9%, 22.5% and 44.8% for SGD, Adam and REF. This is a comparison under the stated attack, not a validated robustness guarantee. Over the fitting cells (46 at the phase, 47 at the endpoint; REF at $k = 50$; robustness at $\epsilon = 0.1$; 90% bootstrap intervals over cells), correlations with robustness are $+0.12$ $[-0.04, +0.38]$ at the phase and -0.69 $[-0.82, -0.53]$ at the endpoint for operator PR,

$+0.36$ $[+0.22, +0.53]$ and $+0.69$ $[+0.55, +0.82]$ for log operator norm, and $+0.56$ $[+0.40, +0.90]$ and $+0.53$ $[+0.37, +0.86]$ for logit margin. The interval for operator PR excludes zero only at the endpoint. The common-logit control in Appendix D.8 shows why this raw structural association should not be interpreted directly as predictor simplification.

An unnormalised logit margin is not a geometric distance to the decision boundary: multiplying all logits by a positive scalar multiplies both the margin and its input gradient while preserving the decision boundary. Common-logit components can also change raw operator norms without affecting softmax predictions. These distinctions must be retained when comparing structural measurements with attack outcomes.

D.16 Depth sweep

The depth sweep asks whether the diagnostic patterns persist when factorisation depth changes. It covers depths $2, 4, 8, 16, 32$, width 128, 12000 updates and three seeds for ReLU and CReLU. Adam's grid extends to 10^{-4} and SGD's to 10^{-3}; Ref uses $\eta \in \{0.3, 1, 3\}$ at $k = 50$. The sweep comprises 390 jobs plus six $k = 200$ jobs: 371 completed and 25 diverged, the latter using large SGD or Adam rates.

Table 22 reports endpoint test accuracy for the validation-selected cell, allowing selection of nonfitting cells. Parentheses give the maximum training accuracy for such cells. Consequently it addresses performance within the tested horizon, rather than a comparison restricted to fitting runs.

Table 22: Endpoint test accuracy across depths for validation-selected cells in 12000-update runs. Selection includes nonfitting cells, whose maximum training accuracy is parenthesised. The additional $k = 200$ Ref result is available only for depth-32 ReLU.

		$L = 2$	$L = 4$	$L = 8$	$L = 16$	$L = 32$
ReLU	SGD	0.538 (.872)	0.558	0.527	0.579	0.505
	Adam	**0.545**	**0.607** (.997)	**0.645**	**0.635** (.997)	0.472 (.992)
	Ref (k=50)	0.502	0.573	0.592	0.567	0.441
	Ref (k=200)	—	—	—	—	**0.520**
CReLU	SGD	0.533 (.940)	0.572	0.517	0.532	0.479
	Adam	**0.543** (.914)	**0.676** (.996)	0.596	**0.645**	**0.528**
	Ref (k=50)	0.516	0.578	**0.597**	0.558	0.491

For the complementary comparison, Table 23 restricts selection to fitting cells and uses the near-interpolation estimator. A dagger marks a selected cell for which only two of three seeds fit; its phase value averages the fitting seeds.

Table 23: Near-interpolation test accuracy across depths after restricting selection to fitting cells. Values average fitting seeds; a dagger marks a cell with only two of three seeds fitting. This selection and estimator differ from the endpoint comparison.

	Rule	$L = 2$	$L = 4$	$L = 8$	$L = 16$	$L = 32$
ReLU	SGD	.526	.557	.527	.575	.507
	Adam	.546	.589	.642	.552	.322
	Ref ($k = 50$)	.508	.586	.593	.557	$.445^{\dagger}$
	Ref ($k = 200$)	—	—	—	—	.504
CReLU	SGD	.518	.572	.511	.531	.477
	Adam	.528	.623	.591	.645	$.531^{\dagger}$
	Ref ($k = 50$)	.520	.587	.597	.556	.474

The estimated operator alignments in Table 24 use the same selected cells and phase as Table 23.

Table 24: Estimated operator alignment for the fitting-phase selections in Table 23. Ref target size is selected at each depth; the $k = 50$ and $k = 200$ rows describe separately trained cells, not fixed-state solver convergence.

$\widehat{c}_{\mathrm{op}}$		$L = 2$	$L = 4$	$L = 8$	$L = 16$	$L = 32$
ReLU	SGD	0.408	0.195	0.213	0.136	0.068
	Adam	0.230	0.122	0.035	0.029	0.076
	Ref (k=50)	1.000	0.835	0.670	0.656	0.694
	Ref (k=200)	—	—	—	—	0.627
CReLU	SGD	0.579	0.290	0.245	0.286	0.155
	Adam	0.337	0.139	0.040	0.023	0.045
	Ref (k=50)	1.000	0.978	0.698	0.901	0.632

SGD's estimated alignment declines overall with depth on ReLU, although it rises from depth 4 to 8; on CReLU it rises from depth 8 to 16. The selected REF target size also changes across depths, so its row combines depth and hyperparameter selection with a fixed solve budget.

At the phase, REF-minus-SGD accuracy differences are $-0.018, +0.029, +0.066, -0.018, -0.062$ on ReLU for $k = 50$, and $+0.002, +0.015, +0.086, +0.025, -0.003$ on CReLU. At ReLU depth 32, $k = 200$ reduces the difference to -0.003. Thus the separation seen at depth 8 is not uniform across depths. Moreover, the selected depth-32 $k = 200$ ReLU cell has higher accuracy but lower estimated alignment than its $k = 50$ counterpart. These trained cells visit different states; their comparison is not a fixed-state solver-convergence test.

A separate check at one fixed depth-32 checkpoint increases the LSQR budget from 50 to 200, 1000 and 3000, with alignments 0.71, 0.88, 0.978 and 0.9999. It shows that a small solve budget can affect the measured direction. It does not explain the trained depth sweep by alignment alone. Per-cell settings, seed counts, fitting counts, seed deviations and maximum training accuracies are listed in the code repository.

D.17 ADDITIONAL OPTIMISERS

The expanded comparison examines whether similar predictive performance can accompany different local update profiles. Heavy ball, Muon, Shampoo, SOAP and K-FAC are evaluated at depth 8, width 128, on ReLU and CReLU. The comparator sweeps use three seeds. Heavy ball uses momentum 0.9 and rates $\{0.003, 0.01, 0.03\}$. Muon uses Nesterov momentum 0.95, five Newton–Schulz iterations and a rate range $3 \cdot 10^{-4}$–0.1. Shampoo accumulates unweighted second moments ($\beta_2 = 1$), uses $\epsilon = 10^{-12}$ and grafts the update to the SGD step norm, with rates $3 \cdot 10^{-3}$–0.1. SOAP uses $\beta = (0.95, 0.95)$, refreshes its preconditioner every ten steps and uses rates 10^{-4}–$3 \cdot 10^{-2}$. Optimiser buffers are stored and restored with checkpoints; they are not reset within a run.

Unit tests show that Adam and heavy ball agree with their PyTorch counterparts to 10^{-16}, and check Muon, Shampoo and SOAP against their published updates. These checks concern update implementation, rather than convergence of the diagnostic projection. Tables 25 and 26 give the selected fitting-cell summaries, with $k = 50$, $\eta = 1$ REF cells. Adam's geometric values carry the solver caveats of Appendix D.2.

Table 25: ReLU comparator summaries for selected fitting cells at depth 8 and width 128. Geometric quantities are estimates.

ReLU	heavy ball	Adam	K-FAC	Ref	SGD	Shampoo	Muon	SOAP
test accuracy	**0.642**	0.634	0.562	0.565	0.531	0.513	0.475	0.458
lr / η	0.03	0.003	0.03	1	0.1	0.1	0.001	0.001
$\widehat{c}_{\mathrm{op}}$	0.150	0.032	0.251	0.861	0.228	0.273	0.039	0.082
$\widehat{c}_{\mathrm{w}}$	0.013	−0.002	0.112	0.425	0.122	0.169	0.023	0.017
$\widehat{\alpha}$	0.996	0.770	1.000	1.000	1.000	1.000	1.000	1.000
operator PR	3.70	**2.05**	6.25	6.55	5.68	6.39	6.99	7.18
gate density	0.316	**0.221**	0.482	0.558	0.488	0.524	0.550	0.559
gate diversity	0.410	0.290	0.407	0.236	0.425	0.388	0.345	0.341
gate churn	0.010	0.021	0.020	0.042	0.013	0.010	0.035	0.022
dead units	0.007	0.120	0.012	0.160	0.009	0.015	0.037	0.024

Table 26: CReLU comparator summaries for selected fitting cells at depth 8 and width 128. Geometric estimates retain their numerical qualifications.

CReLU	Adam	Ref	heavy ball	K-FAC	SGD	Muon	SOAP	Shampoo
test accuracy	**0.612**	0.592	0.587	0.594	0.504	0.495	0.491	0.540
lr / η	0.003	1	0.01	0.03	0.03	0.001	0.001	0.1
$\widehat{c}_{\mathrm{op}}$	0.034	**0.931**	0.151	0.361	0.253	0.072	0.017	0.396
$\widehat{c}_{\mathrm{w}}$	0.066	**0.732**	0.163	0.294	0.272	0.047	0.002	0.348
$\widehat{\alpha}$	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
operator PR	**2.94**	7.16	6.01	7.36	6.61	7.56	7.58	7.02
gate density	0.498	0.498	0.497	0.500	0.505	0.502	0.503	0.502
gate diversity	0.432	0.253	0.449	0.443	0.388	0.364	0.358	0.392
gate churn	0.037	0.045	0.014	0.010	0.009	0.042	0.029	0.006
dead units	0.004	0.167	0.000	0.001	0.003	0.018	0.020	0.005

K-FAC uses factored Tikhonov damping and a rate range 10^{-3}–$3 \cdot 10^{-2}$ with three seeds. Its factors are $A_k = \mathbb{E}[h_{k-1}h_{k-1}^\top]$ and $G_k = \mathbb{E}[d_k d_k^\top]$ with per-example pre-activation gradients d_k (no batch-mean factor), read from the same contexts as M, averaged with decay 0.95 and inverted at every step with damping 10^{-2} split by $\pi = \sqrt{\operatorname{tr} A / \operatorname{tr} G}$ (clamped to $[10^{-4}, 10^4]$, each factor's damping floored at 10^{-6} of its mean eigenvalue). All 24 runs fit. Accuracy rises monotonically with the rate ($0.450 \to 0.562$ on ReLU, $0.487 \to 0.594$ on CReLU), so the best cell is at the edge of the grid and larger rates may do better. In this setting a factor formed from 100 examples in dimension 128 has rank at most 100, making the damping convention consequential.

Some nonfitting cells have higher test accuracy than the selected fitting cells. Muon reaches 0.568 on ReLU and 0.666 on CReLU, with maximum training accuracies 0.997 and 0.995; SOAP reaches 0.524 and 0.608, with maximum training accuracies 0.994 and 0.985. These describe the tested horizon and grids. They do not show that no other setting can both fit and generalise well. Heavy ball's ReLU mean of 0.642 is close to Adam's 0.634; paired uncertainty is needed to interpret that difference. Matrix-update orthogonalisation in Muon does not imply well-conditioned global parameter geometry.

Kernel-fraction estimates are obtained at exactly replayed optimiser states. Each run is re-executed from initialisation to its fitting snapshot, reproducing the stored weights exactly, before the rule's next update u is analysed. For the sampled map M, we compute an approximate minimum-norm lift $\widehat{v} \approx M^+(Mu)$ using the iterative solver, and report $\widehat{q} = \|u - \widehat{v}\|_{\mathcal{W}} / \|u\|_{\mathcal{W}}$. This is a norm fraction; the random-direction statistic in Table 30 uses a squared-norm fraction. The kernel solve uses 3000 iterations and three seeds per cell, except Adam, which uses all principal seeds and 30000 iterations. Replay checks the state used to construct u; the solve with target Mu estimates its effective parameter component separately from the reference solve with target b.

Table 27: Numerical kernel-fraction estimate $\widehat{q}$ at each rule's replayed fitting-snapshot state; mean $\pm$ s.d. over three seeds (Adam: all seeds, 30000 iterations). Seed deviations describe variation across runs; the numerical checks and their scope are given below.

rule	ReLU	CReLU
SGD, REF	0.0000	0.0000
K-FAC	0.074 ± 0.005	0.112 ± 0.002
Shampoo	0.320 ± 0.008	0.409 ± 0.001
heavy ball	0.389 ± 0.032	0.451 ± 0.004
Muon	0.535 ± 0.027	0.673 ± 0.008
Adam	0.742 ± 0.031	0.557 ± 0.038
SOAP	0.746 ± 0.009	0.884 ± 0.020

Only the zero kernel fractions for same-batch SGD and the exact reference follow directly from the geometry. A preconditioner can map a current gradient outside $\mathrm{range}(M^*)$, so nonzero kernel motion need not require momentum. The estimates in Table 27 are consistent with this distinction: heavy ball and Shampoo, which respectively use momentum without preconditioning and second-moment preconditioning without a momentum direction, both have nonzero estimates. K-FAC has smaller estimates at its visited states. The table alone does not establish that its layer-context construction causes this difference, nor does it measure a relationship between q and accuracy.

Numerical checks. Same-batch SGD and the exact reference have $q = 0$ algebraically, and their numerical projector checks round to 0.0000 (Appendix D.3). At Adam's replayed states, increasing the solve budget from 3000 to 30000 iterations changes the kernel fraction by at most $.014$ (Appendix C). This is the reported budget-sensitivity check for Adam. Appendix C separately compares the reference solver with dense SVD on small instances, identifies greater parameter-space sensitivity in an ill-conditioned case, and checks the implemented adjoint. These comparisons assess solver behaviour; they do not supply direct error bounds for every kernel fraction in Table 27. The table reports finite-budget estimates whose seed standard deviations describe variation across runs, not projection error. Unresolved nonzero singular directions can inflate the estimated kernel component. The reported checks therefore support the numerical interpretation within their tested settings, without providing a uniform approximation guarantee.

D.18 ASSOCIATIONS ACROSS AND WITHIN OPTIMISER GROUPS

The exploratory analysis uses an architecture/optimiser/hyperparameter cell, averaged over seeds, as its unit. Both quantities are residualised on architecture–optimiser indicators before computing one pooled correlation. This removes group means; it does not estimate a separate correlation for each optimiser.

Table 28 gives the recomputation on 53 fitting cells in 16 groups (K-FAC re-run; REF keyed by η and $k = 50$; gate diversity as defined in Appendix D; cells without a fitting seed excluded). Intervals resample cells within groups; sensitivity drops either one architecture–optimiser group or one optimiser in both architectures.

Table 28: Correlation with test accuracy over fitting cells: across cells, pooled after removing architecture–optimiser means, 90% bootstrap interval from resampling cells within groups, and range when dropping one group / one optimiser.

diagnostic	across	residualised	90% interval	drop group / optimiser
gate churn	$+0.54$	$+0.56$	$[+0.41, +0.72]$	$+0.50$ to $+0.59$ / $+0.48$ to $+0.61$
gate diversity	$+0.02$	$+0.51$	$[+0.23, +0.72]$	$+0.40$ to $+0.70$ / $+0.43$ to $+0.74$
gate density	-0.52	-0.58	$[-0.77, -0.45]$	-0.70 to -0.46 / -0.71 to -0.47
training loss (log)	-0.53	-0.47	$[-0.81, -0.26]$	-0.60 to -0.42 / -0.74 to -0.38
$\widehat{c}_{\mathrm{op}}$	$+0.39$	$+0.15$	$[-0.05, +0.39]$	$+0.11$ to $+0.45$ / $+0.08$ to $+0.50$
$\widehat{c}_{\mathrm{w}}$	$+0.31$	-0.14	$[-0.29, -0.06]$	-0.17 to -0.11 / -0.16 to -0.08
$\widehat{\alpha}$	-0.16	$+0.05$	$[-0.46, +0.67]$	-0.37 to $+0.54$ / -0.12 to $+0.07$
operator PR	-0.23	-0.08	$[-0.58, +0.38]$	-0.54 to $+0.09$ / -0.44 to $+0.02$
dead units	$+0.31$	-0.09	$[-0.31, +0.15]$	-0.34 to $+0.03$ / -0.50 to -0.04

Five statistics retain their sign under every removal and have intervals excluding zero: gate churn, gate diversity, gate density, training loss and weight alignment. The gate associations are larger in magnitude than the weight-alignment association (-0.14). Operator alignment has an interval including zero, as do reachability, operator PR and dead-unit fraction; reachability includes Adam's unresolved probes. These pooled results describe associations within the tested grids, not a relationship demonstrated separately for every optimiser. Resampling seed-averaged cells does not quantify shared seed or dataset uncertainty, and removing group means does not control learning rate, fitting time or training loss. The nine diagnostics were not specified in advance, so the analysis is exploratory and does not identify a cause of generalisation.

D.19 CIFAR-10 FEASIBILITY CHECK

This check assesses the feasibility of measuring and updating larger local operators. The setup uses 45000 training, 5000 validation and 10000 test images; a seeded permutation defines the training/validation split. Images are flattened to 3072 coordinates, initially in $[0, 1]$, then standardised using training-split statistics. The network has depth 8 and width 128, with batch size 100 and 20000 training steps, approximately 44 epochs. Diagnostic probes use 32 examples, float32 and 1200 LSQR iterations.

The baseline study comprises 11 cells: 9 complete with three seeds each, while ReLU Adam at 0.01 (two seeds) and CReLU SGD at 0.1 (three seeds) diverge, and no hyperparameter selection for this analysis. No cell fits; the highest training accuracy is 0.928. Test accuracy is the last-quarter mean, whereas training loss is the final value. The available summaries are given in Table 29. Two one-seed REF trials were run, at $\eta = 0.3$ and $\eta = 1$; each takes about eight hours on the shared GPU, so the planned multi-seed comparison was stopped.

Table 29: Available CIFAR-10 feasibility summaries for 20000-update runs: last-quarter test accuracy and final training loss. The baseline study uses three seeds; Ref has two separate one-seed trials. No cell reaches the fitting threshold.

arm	hyper	test acc.	train loss	note
SGD	0.01	0.464	0.44	
	0.03	0.473	0.43	
	0.1	0.488	0.58	
Adam	0.001	0.491	0.44	
	0.003	0.501	0.68	
	0.01	0.100	2.30	diverged
Ref	0.3	0.441	1.35	one seed
Ref	1	0.362	3.99	one seed
CReLU SGD	0.01	0.469	0.29	max train acc. 0.928
	0.03	0.480	0.41	
CReLU Adam	0.001	0.505	0.32	
	0.003	0.445	0.86	

The REF trials reach test accuracy 0.441 and 0.362 at training losses 1.35 and 3.99 (maximum training accuracy 0.658 and 0.459), while the baselines attain substantially different losses, so these runs do not establish a comparison at matched fit. They also do not establish a scalable optimisation advantage. A 32-example CIFAR probe has $32 \times 10 \times 3072 = 983040$ target entries, compared with $100 \times 10 \times 40 = 40000$ for the stated MNIST-1D diagnostic. The counts use the actual, different probe batch sizes.

D.20 NOISE ADDED TO THE TRUNCATED UPDATE

This intervention asks how a perturbation of the reference-following increment changes the learning trajectory. It applies

$$\Delta W = \Delta W^{\mathrm{Ref}} + \varepsilon \frac{\|\Delta W^{\mathrm{Ref}}\|}{\|\Pi_{\bullet} g\|} \Pi_{\bullet} g, \qquad g \sim \mathcal{N}(0, I),$$

where $\Pi_\bullet$ is the identity or an estimated projector onto $\ker M$ or $\mathrm{range}(M^*)$. The projected direction is scaled relative to the current parameter increment. An exact kernel perturbation is invisible only to first order on the probe batch, within the fixed-gate model; it need not preserve the function after a finite step or on other inputs.

The numerical kernel/range projector uses 300 LSQR iterations, with operator leakage 1.2% (the norm of the operator change caused by the projected kernel noise, relative to that of the unprojected noise). At $\varepsilon = 0.1$ on ReLU, the induced operator perturbation relative to REF's motion is 0.137 for isotropic noise, 0.017 for estimated kernel noise and 0.324 for range noise. Table 30 reports the estimated fraction of a Gaussian parameter direction's squared norm assigned to the sampled local kernel, across depths and activations.

Table 30: Estimated squared-norm fraction $\|\widehat{\Pi}_{\ker M} g\|^2/\|g\|^2$ of a Gaussian parameter direction assigned to the sampled local kernel by the numerical projection procedure, using 300 LSQR iterations.

Squared-norm fraction	$L = 2$	$L = 8$	$L = 32$
ReLU	0.02	0.63	0.94
CReLU	0.13	0.81	0.96

The estimated squared-norm fraction increases with depth for both activations: from 0.02 at depth two to 0.94 at depth 32 for ReLU, and from 0.13 to 0.96 for CReLU. At depth eight, the fractions are 0.63 and 0.81, respectively. Thus, in these settings, the numerical kernel component contains a larger share of the random direction's squared norm at greater depth. These measurements describe parameter-space geometry relative to the sampled local operator map and depend on the probe and numerical approximation. They do not establish preservation of the network's function under finite perturbations. No kernel-noise training run was performed.

Training uses ReLU, $\eta = 1$, $k = 50$ and two seeds. Seven isotropic settings, $\varepsilon \in \{0, 0.01, 0.03, 0.1, 0.3, 1, 3\}$, include a newly run unperturbed control. Range-noise training was performed only at $\varepsilon = 0.1$. This gives 14 isotropic or control runs and two range runs. The isotropic results are given in Table 31:

Table 31: Isotropic-noise training on ReLU with Ref $\eta = 1$, $k = 50$ and two seeds. Phase and endpoint scores are separate estimators; paired differences use the newly run $\varepsilon = 0$ control. The $\varepsilon = 3$ cell never fits and has no phase score.

ε	0	0.01	0.03	0.1	0.3	1	3
test acc. (convergence)	0.560	0.559	**0.572**	0.565	**0.573**	0.556	—
paired Δ	—	-0.001	$+0.012$	$+0.005$	$+0.013$	-0.004	—
test acc. (endpoint)	0.560	0.568	0.575	**0.577**	0.566	0.559	0.455
paired Δ	—	$+0.008$	$+0.015$	$+\mathbf{0.017}$	$+0.005$	-0.000	-0.104
train loss (conv.)	$5.6 \cdot 10^{-4}$	$2.6 \cdot 10^{-3}$	$5.0 \cdot 10^{-4}$	$8.5 \cdot 10^{-4}$	$1.1 \cdot 10^{-3}$	$7.8 \cdot 10^{-4}$	—
train acc. (conv.)	1.000	0.999	1.000	1.000	1.000	1.000	never
step at convergence	1625	1500	1875	1625	1625	2625	—
gate churn (conv.)	0.0461	0.0478	0.0424	0.0474	0.0483	0.0411	—

The paired convergence-phase differences, in increasing nonzero noise order through $\varepsilon = 1$, are $(0.001, -0.003)$, $(0.014, 0.011)$, $(0.004, 0.006)$, $(0.019, 0.007)$ and $(-0.020, 0.011)$. Seven of the first eight differences are positive. The $\varepsilon = 3$ cell does not fit and has no phase score. The modest apparent gains depend on the measurement phase and have only two-seed support. Gate churn has no monotone trend over the fitting cells, so these observations do not isolate a gate-crossing explanation.

For range noise at $\varepsilon = 0.1$, the phase improvement is 0.005, with seed-level changes 0.008 and 0.003, compared with 0.004 and 0.006 for isotropic noise. The experiment also shows increasing accuracy and gate churn across larger unperturbed target sizes. Such associations do not identify noise as the cause. The noise stream was seeded through Python's per-process string hash, so it cannot be regenerated exactly; the released code uses explicit seeds.

E NUMERICAL CONSISTENCY CHECKS

Cross-entropy and training accuracy. For multiclass cross-entropy on the same examples and with the same nonnegative averaging weights, an incorrectly classified example has true-class probability at most $1/2$. Hence

$$\bar{\ell}_{\mathrm{CE}} \geq (1 - \overline{\mathrm{accuracy}}) \log 2.$$

The inequality also survives averaging across probes and seeds when the weights agree. Training loss is the window mean on the same data as accuracy, and a per-probe check finds no violation. The principal summaries are given in Table 32:

Table 32: Principal summaries with training loss and accuracy evaluated on the same convergence-phase data and windows. Convergence steps are mean threshold-crossing times; test deviations are seed standard deviations. Crossing the threshold does not imply sustained interpolation.

architecture	arm	hyper	conv. step	train loss	train acc.	test acc.
ReLU	SGD	0.1	1050	$1.4 \cdot 10^{-3}$	1.000	0.531 ± 0.010
	Adam	0.003	7250	$9.3 \cdot 10^{-3}$	0.998	$\mathbf{0.634 \pm 0.013}$
	Ref	1	1600	$1.5 \cdot 10^{-3}$	1.000	0.565 ± 0.013
CReLU	SGD	0.03	1000	$1.1 \cdot 10^{-2}$	1.000	0.504 ± 0.016
	Adam	0.003	3750	$3.0 \cdot 10^{-2}$	0.992	$\mathbf{0.612 \pm 0.071}$
	Ref	1	1750	$6.9 \cdot 10^{-4}$	1.000	0.592 ± 0.011
leaky	SGD	0.1	1000	$1.6 \cdot 10^{-3}$	1.000	0.535 ± 0.011
	Adam	0.003	5500	$1.1 \cdot 10^{-4}$	1.000	$\mathbf{0.654 \pm 0.018}$
	Ref	1	1167	$2.2 \cdot 10^{-3}$	1.000	0.567 ± 0.010

For ReLU Adam, the loss $9.3 \cdot 10^{-3}$ exceeds the bound $1.39 \cdot 10^{-3}$ implied by displayed accuracy 0.998. For CReLU Adam, $3.0 \cdot 10^{-2}$ exceeds the corresponding $5.55 \cdot 10^{-3}$ bound at accuracy 0.992. They do not establish equal training loss across rules or sustained interpolation: some Adam seeds fall below the fitting threshold within the averaging window.

Figure provenance. All figures are generated from the validation-selected cells used in the tables. Alignment and gate illustrations use seed 0; the tables aggregate seeds. Scatter plots contain the fitting cells of their named experiment.

F HARDWARE AND COMPUTATIONAL COST

The machine is an NVIDIA RTX 6000 Ada Generation GPU (47.4 GiB, 142 SMs, compute capability 8.9), an Intel Core i9-14900 CPU with 32 threads and 123 GiB host memory, running Linux 6.17, CUDA 13.0, PyTorch 2.11 and Python 3.13. The measured rates on 2048×2048 matrix products are 36.1 TFLOP/s in float32 and 0.99 TFLOP/s in float64. These are observations for this benchmark and workstation, rather than a general precision-cost model.

Recorded elapsed times. The GPU was shared with other jobs throughout, so these times are indicative only.

The complete-run medians in Table 33 include evaluation, diagnostics, checkpointing and I/O. Dashes mark unavailable values.

Table 33: Median complete-run elapsed times, including evaluation, diagnostics, checkpointing and I/O. Dashes mark unavailable values.

Rule	$L = 2$	$L = 4$	$L = 8$	$L = 16$	$L = 32$	Records
SGD	9 s	37 s	60 s	222 s	229 s	134
Adam	10 s	31 s	74 s	279 s	417 s	139
Heavy ball	—	—	59 s	—	—	15
Muon	—	—	75 s	—	—	24
K-FAC	—	—	—	—	—	—
SOAP	—	—	112 s	—	—	29
Shampoo	—	—	429 s	—	—	24
REF ($k = 50$)	111 s	184 s	320 s	938 s	1259 s	87

Float64 probe costs are 4.0, 8.1 and 16.5 seconds at depths $8, 16, 32$. Including the step-zero probe gives seven probes per 12000-step run, or approximately 28 seconds at depth 8. Diagnostics therefore cannot be described as negligible relative to the recorded 60-second SGD run. SOAP refreshes its eigenbasis every ten steps, and its elapsed time includes these refreshes.