NAVIGATING POTHOLES WITH GEOMETRY-AWARE SHARPNESS MINIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Sharpness-aware minimization (SAM) biases optimization toward flat neighborhoods by evaluating gradients at adversarially perturbed parameters, but standard SAM defines these perturbations in Euclidean geometry. Geometry-aware optimizers instead precondition the descent direction using curvature information. We combine these ideas using LLQR (Dufort-Labbé et al., 2026), which learns a structured, slowly varying inverse metric from layerwise network dynamics. LLQR+SAM uses this metric both to shape the SAM perturbation and to precondition the gradient evaluated at the perturbed parameters. On a two-scale quadratic model, we show that the learned geometry enlarges the SAM exploration scale in directions that are flat under the background geometry but locally sharp, providing a mechanism for escaping narrow sharp wells ("potholes"). Empirically, LLQR+SAM consistently improves over SAM and LLQR across CIFAR-10/100, TinyImageNet, ImageNet, and IWSLT14, with analogous gains when combined with recent SAM variants such as F-SAM. These results support the view that slow learned geometry and fast sharpness-aware perturbations can be complementary.

1 INTRODUCTION

Geometry-aware optimizers and sharpness-aware minimization modify different parts of an optimization step. Methods such as natural gradient (Amari, 1998), K-FAC (Martens & Grosse, 2015) and Shampoo (Gupta et al., 2018) use curvature information to reshape the update direction. SAM (Foret et al., 2021), by contrast, changes where that direction is evaluated: it first perturbs the parameters toward a locally adversarial point, then computes the update gradient there. Standard SAM uses an isotropic Euclidean perturbation, leaving these two uses of geometry largely separate.

We ask whether a single learned geometry can connect them. We build on LLQR (Dufort-Labbé et al., 2026), which learns a structured inverse preconditioner U from layerwise network dynamics,

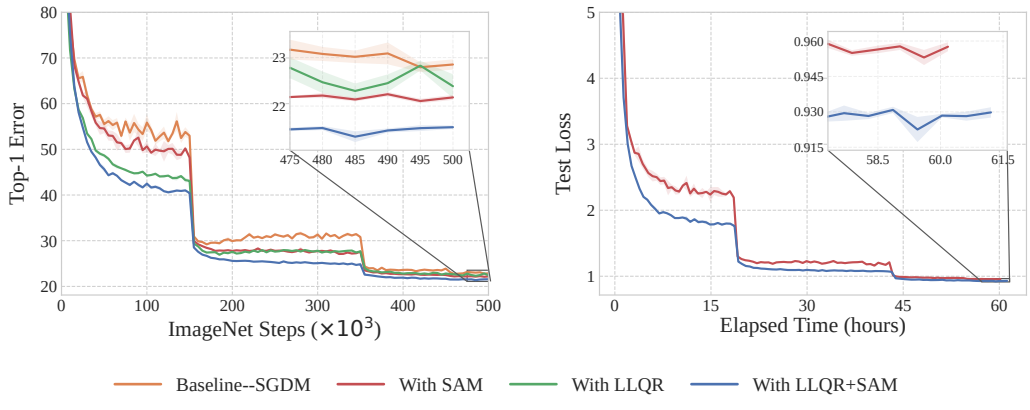

Figure 1: **Interaction between SAM and LLQR on ResNet-50/ImageNet. Left**: top-1 error versus training step. **Right**: test loss versus elapsed time. LLQR+SAM improves over either component alone; matched wall-clock controls are reported in Table 7 and Appendix D.2.

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.

updates it periodically, and smooths it over time. LLQR therefore provides a slowly varying geometry, whereas the SAM perturbation is recomputed from each minibatch. Our approach, LLQR+SAM, uses U both to shape this fast local perturbation and to precondition the resulting gradient.

To understand this coupling, we analyze a local two-scale model consisting of a smooth background curvature and an additional sharp component representing a narrow well, or "pothole." The LLQR metric rescales the SAM exploration radius according to the background curvature: directions that are flat in the background geometry can be explored farther without uniformly increasing the SAM radius. This yields a simple mechanism for escaping localized sharp features.

Empirically, LLQR+SAM improves over both components across CIFAR-10/100, TinyImageNet, and ImageNet, and achieves the highest mean BLEU among the methods tested on IWSLT14. The effect also persists when LLQR is combined with recent SAM variants and under severe label corruption.

Contributions. We make three main contributions. We (i) introduce LLQR+SAM, which uses one slowly learned metric for both the SAM perturbation and outer preconditioning; (ii) characterize its local dynamics in a two-scale sharp-well model; (iii) demonstrate consistent gains across vision and sequence-modeling benchmarks, SAM variants, and label-noise settings.

2 Related Work

Sharpness-aware optimization. SAM (Foret et al., 2021) optimizes the loss evaluated near a locally adversarial parameter perturbation, motivated by the connection between flatness and generalization (Hochreiter & Schmidhuber, 1997; Keskar et al., 2017; Neyshabur et al., 2015), while recognizing that sharpness itself is parameterization dependent (Dinh et al., 2017). Subsequent variants modify the perturbation or the sharpness-aware objective: ASAM (Kwon et al., 2021) introduces scale adaptivity, GSAM (Zhuang et al., 2022) additionally minimizes a surrogate gap related to local sharpness through an orthogonal gradient component, while F-SAM (Li et al., 2024) removes the estimated full-gradient component from the perturbation to emphasize batch-specific stochastic gradient noise.

Geometry-aware SAM. Vanilla SAM defines its neighborhood using the Euclidean norm. FisherSAM (Kim et al., 2022) instead uses a Fisher-induced ellipsoid, Riemannian SAM (Yun & Yang, 2023) formulates SAM under a general Riemannian metric, and preSAM (Zhang et al., 2025) provides a broader preconditioned sharpness-aware framework. These methods establish that the geometry of the adversarial neighborhood matters. LLQR+SAM differs in that the metric is learned from the optimization dynamics and reused for both the perturbation and the outer update.

Geometry-aware optimization. Natural gradient (Amari, 1998), K-FAC (Martens & Grosse, 2015), and Shampoo (Gupta et al., 2018) use structured geometry to precondition the optimization trajectory. More recently, Zhong et al. (2026) study geometry in both stages of SAM using a layerwise spectral-norm perturbation together with conventional optimizers or Muon. LLQR+SAM is closest in spirit to this coupling, but learns a curvature-dependent metric that evolves slowly during training and uses the same metric in both stages. This slow–fast coupling motivates our analysis in Section 4.

3 Method

LLQR+SAM uses a slowly learned optimization geometry in two places: to define the SAM perturbation and to precondition the gradient evaluated at the perturbed parameters. We first introduce the metric, then derive the corresponding sharpness-aware update.

Metric and notation. Let $L : \mathbb{R}^d \to \mathbb{R}$ denote the training loss and $g(\theta) := \nabla L(\theta)$. At iteration t, let $P_t \succ 0$ denote the current optimization metric and $U_t := P_t^{-1}$ its inverse. The associated dual norm on gradients is

$$\|g\|_U := (g^\top U g)^{1/2}.$$

Algorithm 1 LLQR+SAM

1: **Input:** initial parameters θ_0, base optimizer opt with initialized state, SAM radius ρ, LLQR metric state, damping and update schedule.
2: **for** $t = 0, 1, 2, \ldots$ **do**
3: Draw minibatch b_t and compute $g_t = \nabla_\theta L_{b_t}(\theta_t)$.
4: Update the LLQR state if scheduled and obtain the current inverse metric $U_t \approx P_t^{-1}$.
5: Set the metric-aware perturbation $\epsilon_t = \rho\, U_t g_t / (g_t^\top U_t g_t)^{1/2}$.
6: Form the probe parameters $\theta_t^+ = \theta_t + \epsilon_t$.
7: Compute the sharpness-aware gradient $\tilde{g}_t = \nabla_\theta L_{b_t}(\theta_t^+)$.
8: Precondition the probe gradient: $p_t = U_t \tilde{g}_t$.
9: Update the unperturbed parameters with the base optimizer: $\boldsymbol{\theta}_{t+1} = \mathrm{opt}(\boldsymbol{\theta}_t, p_t)$.
10: **end for**
11: **Output:** trained parameters θ_T.

LLQR geometry. LLQR (Dufort-Labbé et al., 2026) learns a structured inverse preconditioner U_t from the layerwise network dynamics and a chosen local quadratic geometry, such as KL/Fisher or Newton/Bregman; Appendix B summarizes the construction. Rather than forming and inverting a global curvature matrix, LLQR optimizes a structured approximation to its inverse action, represented in practice by diagonal or Kronecker-factored blocks. The metric is refreshed only every n_{LLQR}optimizer steps and smoothed with an exponential moving average. Between refreshes, U_t is held fixed and can be applied to a gradient through inexpensive structured matrix-vector products. Thus U_t evolves on a slower timescale than the per-minibatch optimization dynamics.

From Euclidean SAM to metric-aware SAM. Standard SAM (Foret et al., 2021) evaluates a gradient at the worst-case first-order perturbation in a Euclidean ball. With plain SGD as the base optimizer, the update is

$$\theta_t^+ = \theta_t + \rho\, \frac{g(\theta_t)}{\|g(\theta_t)\|_2}, \qquad \theta_{t+1} = \theta_t - \eta\, g(\theta_t^+).$$

LLQR+SAM replaces the Euclidean perturbation geometry by the learned metric $P_t = U_t^{-1}$ and uses the same inverse metric to precondition the outer gradient:

$$\theta_t^+ = \theta_t + \rho\, \frac{U_t\, g(\theta_t)}{\|g(\theta_t)\|_{U_t}}, \qquad \theta_{t+1} = \theta_t - \eta\, U_t\, g(\theta_t^+). \tag{1}$$

The perturbation is therefore the normalized steepest-ascent direction under the metric P, while the descent step applies the same geometry to the gradient evaluated at the perturbed point.

Interpretation. The SGD update in Eq. (1) can be viewed locally as a Riemannian SAM (Yun & Yang, 2023) step in the LLQR metric. With U_t frozen during each perturb-and-recompute step, the update admits a conditional Riemannian-SAM interpretation; Appendix A.1 states the corresponding stationarity transfer under uniform metric assumptions. This also distinguishes LLQR+SAM from Fisher SAM (Kim et al., 2022): Fisher SAM uses a Fisher-scaled perturbation to probe sharpness, but typically applies the resulting gradient with the base optimizer. LLQR+SAM uses the learned inverse metric U_t both to define the perturbation neighborhood and to precondition the outer gradient before passing it to the chosen base optimizer. This couples the fast sharpness-aware probe to the slowly learned optimization geometry; Section 4 isolates the resulting dynamics in a two-scale quadratic model.

Algorithm 1 passes the LLQR-preconditioned sharpness-aware gradient $U_t\tilde{g}_t$ to the practitioner's chosen base optimizer opt, including SGD, SGD with momentum (SGDM), or AdamW. Optimizer state and hyperparameters other than η are implicit in opt, which applies its momentum, adaptive scaling, and weight decay when used. The base optimizer updates the unperturbed parameters $\boldsymbol{\theta}_t$ and advances its state once per iteration, after the probe gradient is computed.

Relative to its base optimizer, SAM adds one additional forward/backward pass per step, while LLQR adds structured preconditioner applications and periodic metric refreshes. The resulting overhead depends on model scale, preconditioner structure, and refresh cadence. Figure 1 shows the ImageNet wall-clock comparison; broader cost and scaling measurements are reported in Sec-

tion 5.5 and Appendix D.5. An anonymized codebase with reproduction instructions is provided as supplementary material.

4 Local Analysis of a Sharp Well

We isolate the interaction between slow geometry and localized sharpness in a minimal quadratic model. The analysis is illustrative: it describes the local dynamics inside a sharp well, not a global model of a neural-network loss.

Two-scale model. Within a local basin, let

$$L(\theta) = \tfrac{1}{2}\,\theta^\top H \theta, \qquad H = \bar{H} + H_\epsilon, \qquad H \succ 0, \tag{2}$$

where $\bar{H} \succ 0$ represents the smooth background curvature and $H_\epsilon \succeq 0$ the additional curvature of the localized sharp feature. We idealize the slowly learned LLQR metric as

$$U = \bar{H}^{-1}.$$

For exposition, assume that $\bar{H}$ and H_ϵ share an eigenbasis; the non-commuting case is handled by whitening with $\bar{H}^{-1/2}$, see Appendix A.2. Along eigendirection v_i, define

$$\lambda_i = \bar{\lambda}_i + \lambda_{\epsilon,i}, \qquad \mu_i = \frac{\lambda_i}{\bar{\lambda}_i}.$$

Thus $\mu_i \approx 1$ where the localized feature adds little curvature, whereas $\mu_i \gg 1$ in strongly sharpened directions.

Local Dynamics. For fixed step size η, the exact LLQR+SAM recursion along v_i is

$$z_{i,t+1} = (1 - \eta\mu_i) z_{i,t} - \eta\rho\,\mu_i \bar{\lambda}_i^{-1/2}\,\mathrm{sign}(z_{i,t}), \qquad z_{i,t} = \langle v_i, \theta_t \rangle.$$

In the stable regime $0 < \eta\mu_i < 1$, the SAM forcing term and the local contraction both scale with μ_i. Their ratio therefore cancels the localized sharpness and gives the characteristic hovering scale

$$|z_{i,t}| \lesssim \rho\,\bar{\lambda}_i^{-1/2}. \tag{3}$$

The exact two-cycle amplitude is given in Appendix A.4. If the local sharp basin has radius r_ϵ in the relevant direction, the model predicts loss of localization when

$$\rho\,\bar{\lambda}_\epsilon^{-1/2} > r_\epsilon. \tag{4}$$

Crucially, this scale is controlled by the *background* curvature rather than by the extra curvature of the sharp feature.

Comparison with SAM. The same calculation for vanilla SAM gives a hovering scale of order ρ in every direction. LLQR+SAM instead rescales that scale by $\bar{\lambda}_i^{-1/2}$, enlarging it in directions that are flat under the background geometry. Matching this effect by increasing the vanilla SAM radius would enlarge the probe in every direction rather than selectively.

Toy illustration. To illustrate this mechanism beyond the local quadratic calculation, we construct an annular landscape with a narrow sharp basin surrounding a broader central minimum (Fig. 2). For matched learning rate η and SAM radius ρ, there are regimes in which Euclidean SAM remains trapped, or crosses the inner ridge without settling in the broad basin, whereas LLQR+SAM reaches the central region. A 21×21 sweep over (η, ρ) shows sustained LLQR+SAM escape over a non-trivial region, whereas SAM does not. Additional controls indicate that continued metric adaptation can sustain escape in some settings, while retaining an already learned metric is sufficient in others. These examples are illustrative rather than evidence of generic nonconvex behavior. A simpler deterministic sharp-well example and a stochastic extension are given in Appendix A.5 and A.7.

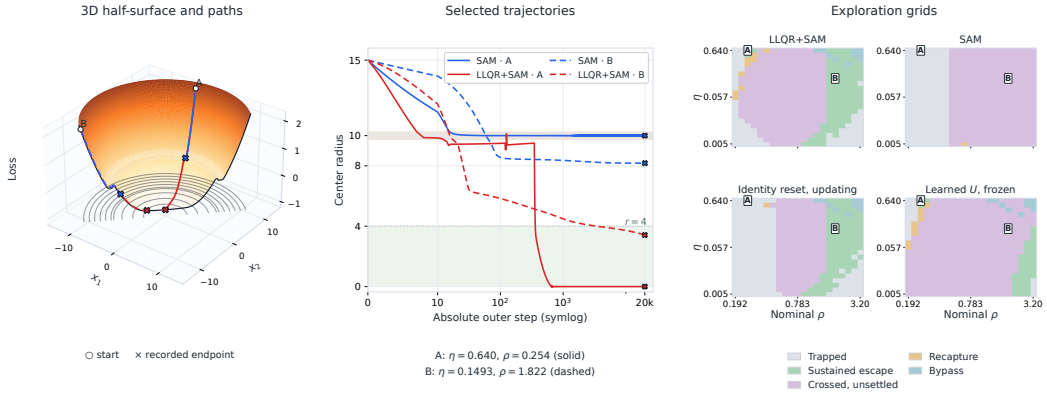

Figure 2: **Toy sharp annular basin escape.** Two illustrative examples (A and B; different outer learning rates (η) and perturbation radii (ρ)) where LLQR+SAM escapes the sharp annular basin but SAM does not. **Left**: radial loss with a narrow annular basin and a broader central minimum. **Middle**: two matched (η, ρ) settings from the parameter sweep; LLQR+SAM reaches the central region, whereas SAM does not within the recorded horizon. **Right**: outcomes over a 21×21 (η, ρ) grid. LLQR+SAM exhibits sustained escape over a nontrivial region where SAM does not; the bottom row shows controls on retention and continued adaptation of the learned metric.

Table 1: **CIFAR-10/100 standard benchmarks.** Entries are best top-1 test accuracy (%, mean ± std, 5 seeds).

	Base geometry		Sharpness baselines		Geometry and sharpness
Architecture	SGDM	LLQR	SAM	FISHERSAM	LLQR+SAM
CIFAR-10					
VGG16-BN	95.10 ± 0.13	95.27 ± 0.20	95.54 ± 0.14	95.45 ± 0.04	$\mathbf{95.92} \pm 0.07$
ResNet-18	96.19 ± 0.16	96.37 ± 0.11	96.65 ± 0.16	96.74 ± 0.14	$\mathbf{96.85} \pm 0.09$
WRN-28-10	96.94 ± 0.09	97.04 ± 0.09	97.43 ± 0.09	97.41 ± 0.04	$\mathbf{97.64} \pm 0.08$
PyramidNet-110	97.18 ± 0.18	97.21 ± 0.09	97.63 ± 0.09	97.57 ± 0.12	$\mathbf{97.86} \pm 0.04$
CIFAR-100					
VGG16-BN	75.65 ± 0.27	76.32 ± 0.36	76.70 ± 0.18	76.70 ± 0.10	$\mathbf{78.14} \pm 0.27$
ResNet-18	78.97 ± 0.49	79.96 ± 0.22	81.13 ± 0.21	80.89 ± 0.31	$\mathbf{81.82} \pm 0.14$
WRN-28-10	82.50 ± 0.18	82.80 ± 0.38	84.83 ± 0.06	84.71 ± 0.13	$\mathbf{85.30} \pm 0.14$
PyramidNet-110	83.96 ± 0.25	84.40 ± 0.47	86.20 ± 0.18	85.92 ± 0.10	$\mathbf{86.67} \pm 0.21$

5 EXPERIMENTS

We test whether LLQR geometry improves sharpness-aware training across architectures, datasets, and optimization settings. All main experiments use NGD-induced LLQR geometry. LLQR and SAM are tuned independently and their settings are reused unchanged when combined, so LLQR+SAM receives no joint retuning. Full protocols are in Appendix C; additional wall-clock, SAM variants, scalability, and exploratory ViT results are in Appendix D.

5.1 VISION BENCHMARKS

CIFAR-10/100. Across four architectures and both datasets, LLQR+SAM improves over both SAM and LLQR in all eight settings (Table 1). The gain over SAM is 0.20–0.38 points on CIFAR-10 and 0.47–1.44 points on CIFAR-100.

TinyImageNet. LLQR+SAM also improves over SAM on both ResNet-18 (62.24 vs. 61.17%) and WRN-28-10 (64.65 vs. 63.91%; Table 2). Notably, LLQR alone underperforms SGDM here, yet its learned geometry still improves SAM.

Table 2: **TinyImageNet and ImageNet benchmarks.** Entries are best top-1 test accuracy (%, mean ± std, 3 seeds).

	Base geometry		Sharpness baseline	Geometry and sharpness
Architecture	SGDM	LLQR	SAM	LLQR+SAM
TinyImageNet-200				
ResNet-18	57.97 ± 0.28	57.02 ± 0.49	61.17 ± 0.27	$\mathbf{62.24} \pm 0.49$
WRN-28-10	61.06 ± 0.37	60.08 ± 0.32	63.91 ± 0.41	$\mathbf{64.65} \pm 0.51$
ImageNet				
ResNet-50	77.60 ± 0.31	78.05 ± 0.12	77.96 ± 0.01	$\mathbf{78.67} \pm 0.08$

Table 3: **CIFAR comparison with SAM variants at 200 epochs.** Entries are best top-1 test accuracy (%, mean ± std, 5 seeds).

	Without LLQR				With LLQR geometry		
Architecture	SAM	F-SAM	ASAM	GSAM	LLQR+SAM	LLQR+F-SAM	LLQR+ASAM
CIFAR-10							
VGG16-BN	95.54 ± 0.14	95.58 ± 0.09	95.07 ± 0.11	94.83 ± 0.23	95.92 ± 0.07	95.81 ± 0.11	$\mathbf{95.95} \pm 0.06$
ResNet-18	96.65 ± 0.16	96.60 ± 0.05	96.77 ± 0.08	96.50 ± 0.10	96.85 ± 0.09	96.88 ± 0.11	$\mathbf{97.05} \pm 0.08$
WRN-28-10	97.43 ± 0.09	97.49 ± 0.08	97.67 ± 0.07	97.38 ± 0.09	97.64 ± 0.08	97.68 ± 0.07	$\mathbf{97.79} \pm 0.11$
PyramidNet-110	97.63 ± 0.09	97.65 ± 0.07	97.86 ± 0.09	97.48 ± 0.08	97.86 ± 0.04	97.89 ± 0.06	$\mathbf{97.96} \pm 0.10$
CIFAR-100							
VGG16-BN	76.70 ± 0.18	76.95 ± 0.32	77.52 ± 0.31	75.52 ± 0.35	78.14 ± 0.27	78.22 ± 0.31	$\mathbf{78.68} \pm 0.22$
ResNet-18	81.13 ± 0.21	81.16 ± 0.12	82.04 ± 0.06	80.78 ± 0.37	81.82 ± 0.14	82.00 ± 0.28	$\mathbf{82.26} \pm 0.35$
WRN-28-10	84.83 ± 0.06	84.86 ± 0.16	85.28 ± 0.24	84.97 ± 0.21	$\mathbf{85.30} \pm 0.14$	85.21 ± 0.20	85.21 ± 0.23
PyramidNet-110	86.20 ± 0.18	86.23 ± 0.08	86.37 ± 0.14	85.98 ± 0.29	86.67 ± 0.21	$\mathbf{86.76} \pm 0.13$	86.53 ± 0.09

Table 4: **Robustness to symmetric label corruption on CIFAR-100**. Entries are best clean-test accuracy (mean ± std, 3 seeds). The corruption rate γ is the fraction of randomly flipped training labels.

γ	SGDM	Adam	LLQR	SAM	LLQR+SAM
0.2	63.41 ± 0.46	41.01 ± 0.63	64.32 ± 0.22	67.15 ± 0.43	$\mathbf{68.64 \pm 0.36}$
0.6	41.21 ± 1.27	9.76 ± 1.46	41.21 ± 1.13	46.81 ± 1.46	$\mathbf{48.54 \pm 0.81}$
0.7	32.52 ± 0.72	4.16 ± 0.52	34.51 ± 0.71	38.96 ± 0.72	$\mathbf{43.27 \pm 0.82}$
0.8	22.97 ± 0.86	2.23 ± 0.12	24.54 ± 0.46	31.19 ± 1.74	$\mathbf{33.06 \pm 1.22}$

ImageNet. On ResNet-50/ImageNet, LLQR+SAM reaches $78.67 \pm 0.08\%$ top-1 accuracy, versus $77.96 \pm 0.01\%$ for SAM and $78.05 \pm 0.12\%$ for LLQR (Table 2). The training trajectories in Fig. 1 (left) show that this advantage persists through much of training.

Composition with recent SAM variants. The gain is not specific to vanilla SAM. LLQR+F-SAM improves F-SAM in all eight CIFAR settings, while LLQR+ASAM improves ASAM in seven of eight CIFAR settings (Table 3). Additional matched-time comparisons are reported in Appendix D.3.

Robustness to Label Noise. Label corruption provides a separate robustness setting in which to test whether learned geometry strengthens the sharpness-aware update. We corrupt CIFAR-100 training labels with symmetric random flips while keeping the test labels clean. LLQR+SAM improves over SAM at every tested corruption rate (Table 4), with gains of 1.49–4.31 percentage points. The largest gain occurs at 70% corruption, where accuracy increases from 38.96% to 43.27%.

5.2 Beyond vision

On IWSLT14 German-to-English with a fairseq Transformer, LLQR+SAM obtains 34.57 ± 0.09 validation BLEU, versus 34.20 ± 0.19 for SAM, 34.51 ± 0.12 for LLQR, and 34.24 ± 0.27 for

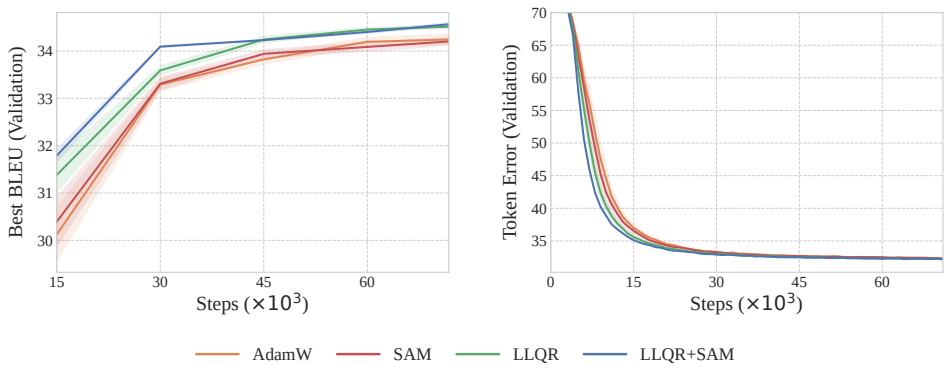

Figure 3: **IWSLT14 German-to-English training dynamics.** Validation BLEU (left) and token error (right) versus training step. LLQR+SAM reaches the high-BLEU, low-token-error regime earlier than the individual LLQR and SAM baselines; final validation BLEU is reported in Table 5.

Table 5: **IWSLT14 German-to-English Transformer benchmark.** Entries are best validation BLEU (mean ± std, 5 seeds).

	Base geometry		Sharpness baseline	Geometry and sharpness
Dataset	AdamW	LLQR	SAM	LLQR+SAM
IWSLT14 De-En	34.24 ± 0.27	34.51 ± 0.12	34.20 ± 0.19	**34.57** ± 0.09

AdamW (Table 5). Although the final BLEU differences are modest, Fig. 3 shows that LLQR+SAM reaches its high-BLEU, low-token-error regime earlier in training.

5.3 Probe geometry and outer preconditioning ablations

LLQR+SAM uses the learned inverse metric in two places: to shape the SAM perturbation and to precondition the gradient evaluated at the perturbed parameters. Table 6 independently ablates these two uses.

On TinyImageNet/ResNet-50, using the LLQR geometry only in the SAM probe provides a modest gain over SAM ($71.63\% \rightarrow 71.80\%$), whereas LLQR outer preconditioning alone raises accuracy to 72.79%. Combining both gives the best result, 73.14%, a further 0.35-point improvement over outer preconditioning alone. These benchmark ablations and the toy analysis address different questions: the former measure aggregate final accuracy, whereas Section 4 isolates regimes in which the probe geometry itself changes sharp-well escape. The relatively larger contribution of outer preconditioning on these benchmarks therefore does not preclude a distinct role for geometry-aware probing in localized sharp regimes.

On CIFAR-100/ResNet-18, the same ablation shows a similar dominance of outer preconditioning: the full construction exceeds the probe-only variant and matches the outer-only mean at 81.82% (Appendix D.4). Thus, outer preconditioning accounts for the larger share of the gain in these ablations, while the geometry-aware probe provides an additional improvement on TinyImageNet and does not reduce mean accuracy on CIFAR-100.

5.4 Wall-clock controls

LLQR introduces additional computation beyond the SAM probe. To test whether the accuracy gains can be explained by this extra compute, we extend the corresponding baselines to comparable or larger wall-clock budgets (Table 7). LLQR+SAM remains ahead of extended SAM in all eight CIFAR settings and on TinyImageNet, and improves ImageNet top-1 accuracy from 77.96% to 78.67% at essentially identical elapsed time ($1.004\times$). On IWSLT14, LLQR+SAM improves

Table 6: **TinyImageNet/ResNet-50 factorial ablation.** Entries are best top-1 test accuracy (%, mean ± std, 5 seeds); gains are percentage points over SGDM (69.56%); for reference LLQR is at 70.65%. The factorial placement ablation independently toggles LLQR geometry in the SAM probe and LLQR preconditioning in the outer update.

LLQR probe	LLQR outer preconditioning	Method	Accuracy (%)	Gain vs. SGDM
No	No	SAM	71.63 ± 0.59	$+2.07$
No	Yes	LLQR+SAM (outer only)	72.79 ± 0.20	$+3.23$
Yes	No	LLQR+SAM (probe only)	71.80 ± 0.33	$+2.24$
Yes	Yes	LLQR+SAM	$\mathbf{73.14} \pm 0.26$	$\mathbf{+3.58}$

Table 7: **SAM versus LLQR+SAM at comparable wall-clock budgets.** Best top-1 accuracy (%) for vision and validation BLEU for IWSLT14 (mean ± std). Reported SAM time ratio is relative to LLQR+SAM; full controls are in Appendix D.2.

Benchmark	Architecture	SAM	LLQR+SAM	SAM time ratio
CIFAR-10	ResNet-18	96.75 ± 0.05	$\mathbf{96.85} \pm 0.09$	$1.02\times$
CIFAR-100	WRN-28-10	84.87 ± 0.13	$\mathbf{85.30} \pm 0.14$	$1.01\times$
TinyImageNet-200	ResNet-18	61.89 ± 0.37	$\mathbf{62.24} \pm 0.49$	$1.04\times$
ImageNet	ResNet-50	77.96 ± 0.01	$\mathbf{78.67} \pm 0.08$	$1.004\times$
IWSLT14 De–En	Transformer	34.21 ± 0.19	$\mathbf{34.57} \pm 0.09$	$1.09\times$

Table 8: **Sensitivity to LLQR update cadence on CIFAR-100/ResNet-18.** Differences in mean top-1 accuracy (percentage points; five seeds per setting). The best tested cadence is 500 optimizer steps.

Update interval	10	50	100	250	500	1000	1500	2500
Δ vs. best	-0.78	-0.36	-0.12	-0.18	0.00	-0.06	-0.14	-0.30
Δ vs. SGDM	$+0.21$	$+0.63$	$+0.87$	$+0.81$	$+0.99$	$+0.93$	$+0.86$	$+0.70$

validation BLEU from 34.21 to 34.57 despite extended SAM receiving a $1.09\times$ larger wall-clock budget. Full controls are reported in Appendix D.2.

5.5 Computational scaling and update cadence

The matched-time controls above address whether additional computation explains the observed gains. We next characterize the computational cost itself. LLQR adds structured preconditioner applications and periodic metric updates. We measure the latter from ViT-Tiny/16 through ViT-Large/16 on ImageNet-scale inputs. Across these four models, a log–log fit of steady preconditioner-update time against trainable parameter count has slope 1.27 for LLQR and 1.28 for LLQR+SAM (Fig. 4). These slopes summarize the measured range and are not asymptotic complexity claims.

Total runtime depends strongly on how often the metric is refreshed. At the least frequent reported settings, LLQR+SAM costs $1.20\times$ SAM per epoch on ViT-B/16 with 1500-step refreshes and $1.63\times$ SAM on ViT-L/16 with 1000-step refreshes; more frequent updates are substantially more expensive. Full runtime and memory measurements are reported in Appendix D.5.

Cadence sensitivity. The computational savings from sparse updates are useful only if the learned geometry remains effective between refreshes. On CIFAR-100/ResNet-18, performance is stable over a broad range of update intervals (Table 8). All intervals from 50 to 2500 steps remain within 0.36 percentage points of the best tested setting at 500 steps while retaining a 0.63–0.99 point gain over SGDM. The 10-step setting performs worse, so more frequent metric updates are not necessarily beneficial.

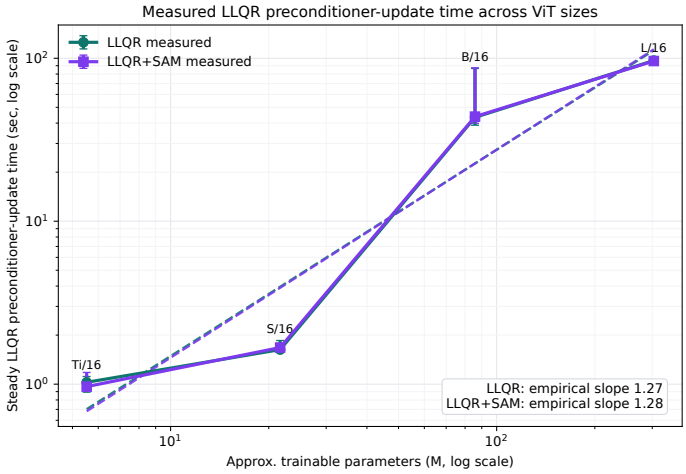

Figure 4: Measured preconditioner-update time across four ViT sizes. A log–log fit gives slopes 1.27 for LLQR and 1.28 for LLQR+SAM.

6 Limitations and Scope

Our analysis isolates a local mechanism rather than providing a global theory of neural-network training. It assumes an idealized two-scale quadratic model and, for the main scalar comparison, a fixed metric $U = \bar{H}^{-1}$ and plain SGD. Deployed LLQR only approximates this geometry, so the analysis does not establish metric alignment in neural networks, global basin selection, or a generalization guarantee. The toy experiments are likewise illustrations on constructed landscapes.

Empirically, we evaluate several vision benchmarks and one sequence-modeling task, but not large-scale language-model training. LLQR+SAM also adds periodic metric fitting, metric application, and structured optimizer state; its cost depends on model scale, implementation, and refresh cadence. Sections 5.4–5.5 characterize this overhead over the tested range. Additional technical assumptions are detailed in Appendix E.

7 Conclusion

LLQR+SAM couples two complementary uses of geometry: a slowly learned metric shapes the fast SAM perturbation and preconditions the gradient evaluated at the perturbed parameters. In a local two-scale quadratic model, this geometry rescales SAM's exploration radius according to the background curvature, selectively enlarging it in directions that are flat under the background geometry but locally sharp. Across CIFAR-10/100, TinyImageNet, ImageNet, and IWSLT14, the combination consistently improves over its individual components; the effect also extends to recent SAM variants (F-SAM, ASAM, and GSAM) and under corrupted-label training.

Taken together, these results suggest that learned optimization geometry can improve not only the descent direction, but also the neighborhood in which sharpness is probed.

AI Use Statement

In this work, we used generative AI tools to refine and assist in validating author-developed theoretical models, conceptual frameworks, and mathematical claims; assist in writing proofs; provide feedback on research methodology and experiments; implement methods; and assist with translation. We did not use generative AI tools to generate synthetic datasets, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, clean or reformat datasets, or interpret results. Qualitative and thematic data analysis are not applicable to this work.

Additionally, we used generative AI tools to create or modify scientific figures; create or edit software code; draft parts of the manuscript under author guidance from existing notes and observations;

summarize and analyse existing literature during the literature review; brainstorm, search for information, and identify relevant literature; and edit the manuscript to improve readability. AI-assisted literature summaries and analyses informed our review of sources and were not used as text for the related-work section.

We have reviewed all AI-assisted work. This review included checking mathematical statements and proofs, testing implementations for correctness, inspecting scientific figures, verifying literature sources, and reviewing drafted, edited, and translated text. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

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

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 248–255. IEEE Computer Society, 2009. doi: 10.1109/CVPR.2009.5206848. URL https://doi.org/10.1109/CVPR.2009.5206848.

Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017. URL http://arxiv.org/abs/1708.04552.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, Proceedings of Machine Learning Research, pp. 1019–1028. PMLR, 2017. URL http://proceedings.mlr.press/v70/dinh17b.html.

Simon Dufort-Labbé, Pierre-Luc Bacon, Razvan Pascanu, Simon Lacoste-Julien, and Aristide Baratin. Layerwise lqr for geometry-aware optimization of deep networks, 2026. URL https://arxiv.org/abs/2605.04230.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=6Tm1mposlrM.

Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, Proceedings of Machine Learning Research, pp. 1837–1845. PMLR, 2018. URL http://proceedings.mlr.press/v80/gupta18a.html.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Comput.*, 9(1):1–42, 1997. doi: 10.1162/neco.1997.9.1.1. URL https://doi.org/10.1162/neco.1997.9.1.1.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=H1oyRlYgg.

Minyoung Kim, Da Li, Shell Xu Hu, and Timothy M. Hospedales. Fisher SAM: information geometry and sharpness aware minimisation. *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, pp. 11148–11161, 2022. URL https://proceedings.mlr.press/v162/kim22f.html.

Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. ASAM: adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, Proceedings of Machine Learning Research, pp. 5905–5914. PMLR, 2021. URL http://proceedings.mlr.press/v139/kwon21b.html.

Bingcong Li and Georgios B. Giannakis. Enhancing sharpness-aware optimization through variance suppression. *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Tao Li, Pan Zhou, Zhengbao He, Xinwen Cheng, and Xiaolin Huang. Friendly sharpness-aware minimization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pp. 5631–5640. IEEE, 2024. doi: 10.1109/CVPR52733.2024.00538. URL https://doi.org/10.1109/CVPR52733.2024.00538.

James Martens and Roger B. Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, JMLR Workshop and Conference Proceedings, pp. 2408–2417. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/martens15.html.

Peng Mi, Li Shen, Tianhe Ren, Yiyi Zhou, Xiaoshuai Sun, Rongrong Ji, and Dacheng Tao. Make sharpness-aware minimization stronger: A sparsified perturbation approach. *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Behnam Neyshabur, Ruslan Salakhutdinov, and Nathan Srebro. Path-sgd: Path-normalized optimization in deep neural networks. *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 2422–2430, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/eaa32c96f620053cf442ad32258076b9-Abstract.html.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pp. 1–9. Association for Computational Linguistics, 2018. doi: 10.18653/V1/W18-6301. URL https://doi.org/10.18653/v1/w18-6301.

Jihun Yun and Eunho Yang. Riemannian SAM: sharpness-aware minimization on riemannian manifolds. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/cf701db0e3b4d0b8681ca6915ac3e87e-Abstract-Conference.html.

Yilang Zhang, Bingcong Li, and Georgios B. Giannakis. Preconditioned sharpness-aware minimization: Unifying analysis and a novel learning algorithm. In *2025 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2025, Hyderabad, India, April 6-11, 2025*, pp. 1–5. IEEE, 2025. doi: 10.1109/ICASSP49660.2025.10889586. URL https://doi.org/10.1109/ICASSP49660.2025.10889586.

Wenzhi Zhong, Edward Milsom, and Michael Murray. Sharpness-aware minimization and muon: Robustness under the spectral norm. *CoRR*, abs/2607.26001, 2026. doi: 10.48550/ARXIV.2607.26001. URL https://doi.org/10.48550/arXiv.2607.26001.

Juntang Zhuang, Boqing Gong, Liangzhe Yuan, Yin Cui, Hartwig Adam, Nicha C. Dvornek, Sekhar Tatikonda, James S. Duncan, and Ting Liu. Surrogate gap minimization improves sharpness-aware training. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=edONMAnhLu-.

A Detailed Analysis for the Two-Scale Pothole Model

The results in this appendix concern the plain-SGD update in Eq. (1), without momentum, adaptive scaling, or weight decay. They do not establish the corresponding guarantees for SGDM or AdamW.

This appendix contains the supporting arguments used in Sections 3 and 4. The first subsection records the fixed-metric Riemannian-SAM transfer for LLQR +SAM. The remaining subsections justify the two-scale pothole calculation: the coordinate recursion, the scalar hovering envelope, the vanilla-SAM comparison, and the stochastic selection picture.

A.1 Riemannian-SAM transfer under the frozen LLQR metric

This subsection records the convergence-transfer statement used in Section 3. The statement is conditional on the LLQR metric being frozen during each perturb-and-recompute step; it gives stationarity in the instantaneous metric, not in a fixed limiting metric unless additional metric convergence is assumed.

Let $P_t = U_t^{-1}$. Assume that U_t is measurable with respect to the optimization history $\mathcal{F}_t$, is fixed during the SAM perturb-and-recompute step, and satisfies the uniform spectral bounds

$$0 < u_{\min} I \preceq U_t \preceq u_{\max} I < \infty.$$

Assume also that the smoothness, retraction-Lipschitz, and stochastic-gradient assumptions of Riemannian SAM (Yun & Yang, 2023) hold uniformly for the compact family of metrics $\{P_t\}$.

Proposition A.1 (Riemannian-SAM transfer for LLQR +SAM)**.** *With the same decaying stepsize and perturbation radius schedule as in Riemannian SAM, for $\tau \sim \mathrm{Unif}\{0, \dots, T-1\}$,*

$$\mathbb{E}\left[\| \operatorname{grad}_{P_\tau} L(\theta_\tau)\|_{P_\tau}^2\right] = \mathbb{E}\left[\nabla L(\theta_\tau)^\top U_\tau \nabla L(\theta_\tau)\right] = O(T^{-1/2}),$$

up to the same stochastic-gradient and minibatch-variance terms as in the Riemannian-SAM theorem. Consequently,

$$\mathbb{E}\|\nabla L(\theta_\tau)\|_2^2 = O(T^{-1/2}).$$

Proof. Condition on $\mathcal{F}_t$. Since U_t is $\mathcal{F}_t$-measurable and fixed during the perturb-and-recompute step, $P_t = U_t^{-1}$ is a fixed Riemannian metric for the current update. For the constant metric

$$\langle \xi, \zeta \rangle_{P_t} = \xi^\top P_t \zeta,$$

the Riemannian gradient is

$$\operatorname{grad}_{P_t} L(\theta) = P_t^{-1} \nabla L(\theta) = U_t \nabla L(\theta),$$

and hence

$$\| \operatorname{grad}_{P_t} L(\theta)\|_{P_t}^2 = \nabla L(\theta)^\top U_t \nabla L(\theta).$$

The $LLQR + SAM$ perturbation

$$\epsilon_t = \rho_t \frac{U_t g_t}{(g_t^\top U_t g_t)^{1/2}}$$

is therefore the normalized Riemannian steepest-ascent direction under the frozen metric P_t. The update

$$\theta_{t+1} = \theta_t - \alpha_t U_t \nabla L_{b_t}(\theta_t + \epsilon_t)$$

is the corresponding Riemannian steepest-descent step on the probed loss. Conditioned on $\mathcal{F}_t$, this is one fixed-metric Riemannian-SAM step.

The spectral bounds on U_t place all metrics $P_t = U_t^{-1}$ in a compact subset of the positive-definite cone. Hence the norm-equivalence, smoothness, retraction-Lipschitz, and stochastic-gradient constants in the Riemannian-SAM one-step descent inequality can be chosen uniformly over t. Applying that inequality conditionally on $\mathcal{F}_t$, then taking expectation and summing over $t = 0, \dots, T-1$, gives

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\| \operatorname{grad}_{P_t} L(\theta_t)\|_{P_t}^2 = O(T^{-1/2}),$$

up to the same stochastic-gradient and minibatch-variance terms. Equivalently, for $\tau \sim \mathrm{Unif}\{0, \dots, T-1\}$,

$$\mathbb{E}\| \operatorname{grad}_{P_\tau} L(\theta_\tau)\|_{P_\tau}^2 = O(T^{-1/2}).$$

Finally, since $U_t \succeq u_{\min} I$,

$$\nabla L(\theta_t)^\top U_t \nabla L(\theta_t) \geq u_{\min} \|\nabla L(\theta_t)\|_2^2,$$

so stationarity in the instantaneous $LLQR$ metric implies Euclidean first-order stationarity:

$$\mathbb{E}\|\nabla L(\theta_\tau)\|_2^2 \leq u_{\min}^{-1} \mathbb{E}\left[\nabla L(\theta_\tau)^\top U_\tau \nabla L(\theta_\tau)\right] = O(T^{-1/2}).$$

If $U_t \to U_f$ and $\|\nabla L(\theta_t)\|_2 \leq G$, then

$$\left|\nabla L(\theta_t)^\top U_t \nabla L(\theta_t) - \nabla L(\theta_t)^\top U_f \nabla L(\theta_t)\right| \leq G^2 \|U_t - U_f\|_{\mathrm{op}}.$$

Averaging gives

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[\nabla L(\theta_t)^\top U_f \nabla L(\theta_t)\right] \leq O(T^{-1/2}) + G^2 \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\|U_t - U_f\|_{\mathrm{op}}.$$

Thus convergence in the limiting metric follows whenever this average discrepancy vanishes, and the original rate is preserved when the discrepancy is $O(T^{-1/2})$. □

A.2 Whitened form and non-commuting case

The commuting hypothesis used in the main text is not essential. Define

$$y_t := \bar{H}^{1/2} e_t, \qquad A := \bar{H}^{-1/2} H \bar{H}^{-1/2} = I + \bar{H}^{-1/2} H_\epsilon \bar{H}^{-1/2}.$$

Since $A \succ 0$ is symmetric, it admits an orthonormal eigendecomposition. Its eigenvalues are the perceived sharpnesses μ_i. In the commuting case, these reduce to

$$\mu_i = \frac{\lambda_i}{\bar{\lambda}_i} = 1 + \frac{\lambda_{\epsilon,i}}{\bar{\lambda}_i}.$$

Using $U = \bar{H}^{-1}$, the exact LLQR+SAM recursion becomes, in whitened coordinates,

$$y_{t+1} = (I - \eta A) y_t - \frac{\eta\rho}{\|A y_t\|_2} A^2 y_t. \tag{5}$$

Thus the non-commuting case is identical after diagonalizing the normalized curvature matrix A. The shared-eigenbasis presentation in the main text is only a notational simplification.

A.3 Coordinate recursion

For the quadratic loss $L(\theta) = \frac{1}{2}\theta^\top H \theta$, we have $g(\theta) = H\theta$. The LLQR+SAM perturbation gives

$$g(\theta^+) = H\theta + \rho \frac{HUH\theta}{\|H\theta\|_U},$$

and therefore, with $e_t = \theta_t$,

$$e_{t+1} = (I - \eta UH) e_t - \frac{\eta\rho}{\|He_t\|_U} UHUHe_t. \tag{6}$$

In the commuting case, let v_i be the common eigenvectors of $\bar{H}$, H_ϵ, and H, and write $e_t = \sum_i z_{i,t} v_i$. Since

$$\bar{H} v_i = \bar{\lambda}_i v_i, \qquad H v_i = \lambda_i v_i,$$

we get

$$UHv_i = \bar{H}^{-1} H v_i = \frac{\lambda_i}{\bar{\lambda}_i} v_i = \mu_i v_i, \qquad UHUHv_i = \mu_i^2 v_i.$$

Moreover,

$$\|He_t\|_U^2 = (He_t)^\top U(He_t) = \sum_j \lambda_j^2 \bar{\lambda}_j^{-1} z_{j,t}^2 = \sum_j \mu_j^2 \bar{\lambda}_j z_{j,t}^2.$$

Substituting these identities into (6) gives the full coordinate recursion

$$z_{i,t+1} = (1 - \eta\mu_i) z_{i,t} - \frac{\eta\rho\mu_i^2}{\sqrt{\sum_j \mu_j^2 \bar{\lambda}_j z_{j,t}^2}} z_{i,t}. \tag{7}$$

If one eigendirection v_i dominates, the denominator reduces to $\mu_i\sqrt{\bar{\lambda}_i}|z_{i,t}|$, yielding the scalar recursion

$$z_{i,t+1} = (1 - \eta\mu_i) z_{i,t} - \frac{\eta\rho\mu_i}{\sqrt{\bar{\lambda}_i}} \operatorname{sign}(z_{i,t}). \tag{8}$$

This is the one-dimensional model used in the main text.

A.4 Scalar Hovering Envelope

Consider the scalar map

$$z_{t+1} = a z_t - b \operatorname{sign}(z_t), \qquad a \in (0,1), \quad b > 0. \tag{9}$$

It has no nonzero fixed point. Indeed, if $z > 0$, then

$$z = az - b \quad \Longrightarrow \quad z = -\frac{b}{1-a} < 0,$$

which contradicts $z > 0$; the negative half-line gives the symmetric contradiction. Hence the map cannot localize at a nonzero point and instead oscillates around the origin.

Once the signs alternate, the limiting two-cycle has amplitude $r_\star$ satisfying

$$r_\star = b - a r_\star,$$

so

$$r_\star = \frac{b}{1+a}. \tag{10}$$

For (8),

$$a_i = 1 - \eta\mu_i, \qquad b_i = \frac{\eta\rho\mu_i}{\sqrt{\bar{\lambda}_i}},$$

and therefore the exact asymptotic two-cycle amplitude is

$$\limsup_{t\to\infty} |z_{i,t}| = \frac{b_i}{1+a_i} = \frac{\eta\rho\mu_i}{(2-\eta\mu_i)\sqrt{\bar{\lambda}_i}}. \tag{11}$$

The main text uses the simpler hovering envelope

$$|z_{i,t}| \lesssim \frac{b_i}{1-a_i} = \frac{\rho}{\sqrt{\bar{\lambda}_i}}. \tag{12}$$

This envelope is the scale relevant for the cancellation argument: the SAM kick b_i and the contraction scale $1 - a_i = \eta\mu_i$ both grow linearly with the perceived sharpness μ_i, leaving a radius controlled only by the average curvature $\bar{\lambda}_i$.

A.5 Pothole Escape Criterion

The quadratic model is local, so it does not by itself describe the global motion after leaving a well. It gives a localization criterion. If a pothole has basin radius r_ϵ in the relevant eigendirection, then the local model predicts loss of localization when the hovering envelope exceeds that radius:

$$\frac{\rho}{\sqrt{\bar{\lambda}_\epsilon}} > r_\epsilon. \tag{13}$$

The key point is that this criterion depends on the average curvature $\bar{\lambda}_\epsilon$, not on the localized pothole sharpness λ_ϵ. A sharper localized component increases the perceived sharpness μ_ϵ, but the SAM kick and contraction scale increase together, producing the cancellation in (12).

In a wide basin, small $\bar{\lambda}_i$ is paired with a large basin radius, so the same envelope can remain inside the basin. In a pothole, the basin radius is small while the average direction remains flat, so the same envelope can exceed the local basin radius.

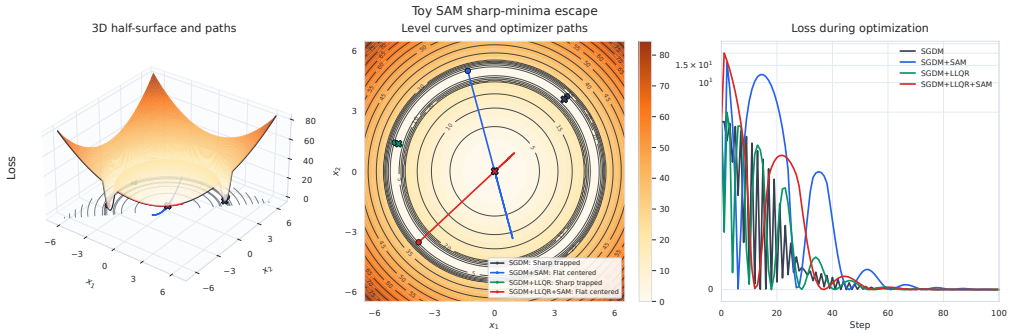

Figure 5: **Deterministic sharp-well escape.** All methods start inside the narrow sharp basin with the same learning rate; SAM and LLQR+SAM use the same perturbation radius. SGDM and LLQR remain localized, whereas both SAM variants escape toward the broader basin.

Deterministic toy illustration. Figure 5 illustrates this local escape picture on a two-dimensional constructed loss containing a narrow sharp basin and a broader minimum. All methods start inside the sharp basin with the same learning rate, and the two SAM variants use the same perturbation radius. SGDM and LLQR remain localized in the sharp basin, whereas SAM and LLQR+SAM escape toward the broad basin. In this example, LLQR+SAM reaches the flat region along a shorter trajectory. The experiment is intended only as an illustration of the non-localization mechanism above; it is not a general comparison of escape times.

A.6 Vanilla SAM Comparison

For vanilla SAM, $U = I$. In an eigendirection of H, the scalar recursion is

$$z_{t+1} = (1 - \eta\lambda_i)z_t - \eta\rho\lambda_i \operatorname{sign}(z_t). \tag{14}$$

The same envelope calculation gives

$$\frac{\eta\rho\lambda_i}{\eta\lambda_i} = \rho. \tag{15}$$

Thus vanilla SAM has a Euclidean hovering envelope of order ρ in every direction, whereas LLQR+SAM has the direction-dependent envelope $\rho/\sqrt{\bar{\lambda}_i}$.

Corollary A.2 (Pothole-escape amplification)**.** *Under the commuting hypothesis, the LLQR+SAM hovering envelope around a pothole minimum is*

$$\frac{\rho}{\sqrt{\bar{\lambda}_\epsilon}} = \frac{1}{\sqrt{\bar{\lambda}_\epsilon}} \times \big(\textit{vanilla-SAM hovering envelope}\big).$$

The amplification depends only on the average curvature $\bar{\lambda}_\epsilon$ in the pothole direction, not on the localized pothole sharpness λ_ϵ.

This amplification cannot be reproduced selectively by retuning vanilla SAM. Choosing

$$\rho^{\mathrm{SAM}} = \frac{\rho}{\sqrt{\bar{\lambda}_\epsilon}}$$

would match the pothole-direction envelope, but it would enlarge the probe in all directions, including non-pothole directions where this extra scale is not needed. The contraction behavior also differs: vanilla SAM contracts at rate $1 - \eta\lambda_i$, so its stable step size is constrained by the largest curvature of H. LLQR+SAM contracts at rate $1 - \eta\mu_i$, which removes the conditioning of the average geometry $\bar{H}$ from this constraint.

A.7 Stochastic Selection Between Wells

The deterministic analysis in Sections A.4- A.5 gives a local non-localization criterion: when the SAM hovering scale becomes comparable to the radius of a sharp well, the iterate cannot remain

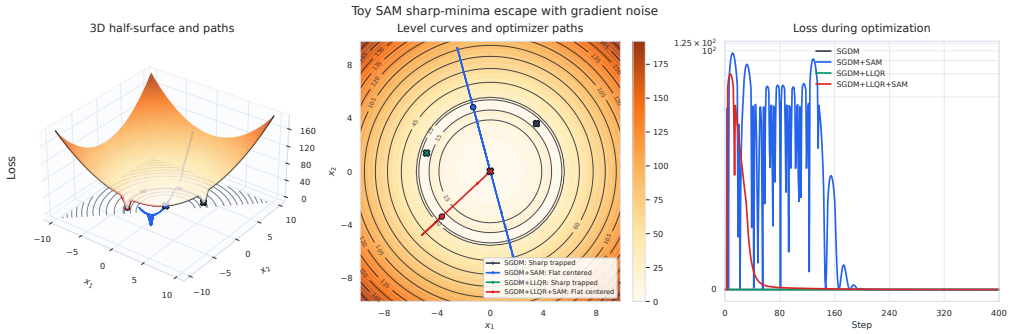

Figure 6: **Gradient-noise escape from the sharp minimum.** All variants start at the bottom of the sharp well and receive a shared deterministic Gaussian perturbation schedule in the update gradient. The non-SAM variants remain near the sharp well, while SAM variants are ejected. At variance 10^{-9}, both SAM variants reach the flat basin, but LLQR +SAM has substantially shorter path length than Euclidean SAM, consistent with the stochastic selection picture in Appendix A.7.

tightly localized near its bottom. Figure 5 illustrates this mechanism deterministically. With stochastic gradients, the same effect suggests a selection mechanism between basins. Noise may repeatedly inject the iterate into a sharp well, but sharp wells from which the SAM dynamics rapidly escape have short residence times; wider basins can support substantially longer visits because their radius is large relative to the local hovering scale.

The following regenerative model formalizes this residence-time interpretation. It is deliberately idealized and is not intended as a global model of neural-network training.

Proposition A.3 (Regenerative selection between wells)**.** *Consider a regenerative idealization with wells indexed by m. At each cycle, a well J_n is sampled from a distribution ν with positive mass on each well; the process starts near the well center, evolves under local gradient-noise dynamics with noise scale σ, and exits when $\|e_t\| \geq R$. Suppose flat wells satisfy*

$$\mathbb{E}[\tau_R^{(m)}] \geq c_m^{\mathrm{flat}} R^2 \sigma^{-2},$$

whereas sharp unstable wells satisfy

$$\mathbb{E}[\tau_R^{(m)}] \leq C_m^{\mathrm{sharp}} \big(1 + \log(R/\sigma)\big).$$

If at least one flat well is present, then the long-run occupation mass of sharp wells satisfies

$$\sum_{m \in \mathcal{M}_{\mathrm{sharp}}} \mu_m(\sigma) = O\left(\frac{\sigma^2 \log(R/\sigma)}{R^2}\right) \xrightarrow[\sigma \to 0]{} 0.$$

Proof. By the renewal-reward formula,

$$\mu_m(\sigma) = \frac{\nu_m \mathbb{E}[\tau_R^{(m)}]}{\sum_\ell \nu_\ell \mathbb{E}[\tau_R^{(\ell)}]}.$$

The total sharp-well numerator is $O(1+\log(R/\sigma))$, while the denominator is at least the contribution of one flat well, namely $\Omega(R^2\sigma^{-2})$. Dividing gives the stated bound. □

For a pothole direction, the relevant dimensionless escape ratio is

$$\frac{\rho}{r_\epsilon \sqrt{\bar{\lambda}_\epsilon}}.$$

Corollary A.2 increases this ratio for LLQR +SAM relative to vanilla SAM, predicting shorter visits to sharp wells and faster selection of the wide flat basin.

The next calculation supports the path-length behavior observed in the noisy toy experiment. It states that, in a stable scalar noisy mode, increasing the effective metric denominator damps both stationary variance and one-step motion.

Proposition A.4 (Metric damping under gradient noise)**.** *Consider the scalar noisy recursion*

$$z_{t+1}^B = q_i^B z_t^B - \eta (d_i^B)^{-1} \varepsilon_{i,t}, \qquad q_i^B := 1 - \eta \lambda_i (d_i^B)^{-1}, \qquad \mathrm{Var}(\varepsilon_{i,t}) = \tau_i^2,$$

where B denotes a metric choice and $d_i^B > \eta\lambda_i/2$. Then

$$\mathrm{Var}_{\pi_B}(z_i) = \frac{\eta \tau_i^2}{\lambda_i (2 d_i^B - \eta \lambda_i)}, \qquad \mathbb{E}_{\pi_B}\big[(z_{i,t+1}^B - z_{i,t}^B)^2\big] = \frac{2\eta^2 \tau_i^2}{d_i^B (2 d_i^B - \eta \lambda_i)}.$$

Consequently, if $d_i^P \geq d_i^Q > \eta\lambda_i/2$, then metric P has no larger stationary variance or one-step motion than metric Q, with strict improvement when $\tau_i^2 > 0$ and $d_i^P > d_i^Q$.

Proof. The AR(1) variance equation gives

$$V_i^B = (q_i^B)^2 V_i^B + \frac{\eta^2 \tau_i^2}{(d_i^B)^2}.$$

Since

$$1 - (q_i^B)^2 = \frac{\eta \lambda_i (2 d_i^B - \eta \lambda_i)}{(d_i^B)^2},$$

we obtain

$$V_i^B = \frac{\eta \tau_i^2}{\lambda_i (2 d_i^B - \eta \lambda_i)}.$$

Moreover,

$$z_{i,t+1}^B - z_{i,t}^B = -\eta \lambda_i (d_i^B)^{-1} z_{i,t}^B - \eta (d_i^B)^{-1} \varepsilon_{i,t}.$$

At stationarity, $z_{i,t}^B$ is independent of the fresh noise, so the cross term vanishes and

$$\mathbb{E}_{\pi_B}\big[(z_{i,t+1}^B - z_{i,t}^B)^2\big] = \frac{\eta^2}{(d_i^B)^2}\left(\lambda_i^2 V_i^B + \tau_i^2\right) = \frac{2\eta^2 \tau_i^2}{d_i^B (2 d_i^B - \eta \lambda_i)}.$$

Both denominators are increasing for $d_i^B > \eta\lambda_i/2$, which proves the comparison. □

Figure 6 provides a stochastic counterpart to the deterministic illustration in Fig. 5. Under a shared gradient-noise schedule, both SAM variants are ejected from the sharp basin, while the non-SAM methods remain nearby. In the displayed setting, LLQR+SAM reaches the broad basin with a markedly shorter path than Euclidean SAM, consistent with the residence-time and metric-damping pictures above.

B Layerwise LQR: Learning a Reusable Geometry

We briefly summarize the Layerwise LQR (LLQR) methodology of Dufort-Labbé et al. (2026), which provides the geometry used throughout this work. The motivation is to obtain the benefits of second-order or natural-gradient descent without explicitly forming, storing, or inverting a global curvature matrix. At an iterate θ^k, a broad class of geometry-aware descent directions can be written as the solution of the local quadratic problem

$$\Delta\theta^\star = \arg\min_{\Delta\theta} \left\{ \nabla L(\theta^k)^\top \Delta\theta + \frac{1}{2} \Delta\theta^\top H(\theta^k) \Delta\theta \right\}, \tag{16}$$

where H may be a regularized Hessian, Gauss–Newton matrix, Fisher matrix, or more general divergence-induced metric. Directly solving $H\Delta\theta = -\nabla L(\theta^k)$ is infeasible for modern networks, and standard scalable approximations typically impose block structure on H before solving the problem. LLQR instead first rewrites the dense quadratic problem as a layerwise optimal-control problem, and only later imposes structure on the learned inverse update.

Consider a depth-N network

$$x_{i+1} = f_i(x_i, \theta_i), \qquad i = 0, \ldots, N-1.$$

Algorithm 2 LLQR: relaxed periodic preconditioner

Require: model f, loss L, data loader $\mathcal{D}$, outer optimizer $\mathcal{O}_{\text{out}}$, step size η
Require: recompute period n, inner steps T, structure $\mathcal{S}$, inner optimizer $\mathcal{O}_{\text{in}}$, inner step size α, EMA parameter β
Initialize θ^0 and $U \leftarrow I$ projected to $\mathcal{S}$
for $k = 0, 1, 2, \ldots$ **do**
  Sample minibatch $(x^k, y^k) \sim \mathcal{D}$ and compute $g_k \leftarrow \nabla_\theta L(\theta^k)$
  if $k \bmod n = 0$ **then**
    Linearize the network: $(A_i, B_i) \leftarrow (\nabla_x, \nabla_\theta) f_i(x_i^k, \theta_i^k)$
    Form the LQR blocks (Q_i, R_i, M_i) for the chosen divergence-induced metric
    Set $U_0 \leftarrow U$
    for $t = 0$ to $T - 1$ **do**
      $U_{t+1} \leftarrow \mathcal{O}_{\text{in}}(U_t, \nabla_{U_t} J, \alpha)$, where J is the relaxed objective (18)
    end for
    $U \leftarrow \beta U + (1 - \beta) U_T$
  end if
  Precondition the gradient: $\widetilde{g}_k \leftarrow U g_k$
  Update parameters with the base optimizer: $\theta^{k+1} \leftarrow \mathcal{O}_{\text{out}}(\theta^k, \widetilde{g}_k, \eta)$
end for

Linearizing the forward pass at (x_i^k, θ_i^k) gives the perturbation dynamics

$$\delta x_{i+1} = A_i \delta x_i + B_i \delta \theta_i, \qquad \delta x_0 = 0, \tag{17}$$

with $A_i = \partial f_i / \partial x_i$ and $B_i = \partial f_i / \partial \theta_i$. For divergence-induced quadratic models, the global quadratic form in (16) can then be decomposed into layerwise costs

$$\delta x_N^\top Q_N \delta x_N + \sum_{i=0}^{N-1} \begin{bmatrix} \delta x_i \\ \delta \theta_i \end{bmatrix}^\top \begin{bmatrix} Q_i & M_i^\top \\ M_i & R_i \end{bmatrix} \begin{bmatrix} \delta x_i \\ \delta \theta_i \end{bmatrix},$$

subject to the linearized dynamics (17). The resulting problem is a finite-horizon Linear Quadratic Regulator: the network defines the dynamics, while the chosen descent geometry defines the quadratic costs. Its exact Riccati solution recovers the corresponding second-order update, including Newton, Gauss–Newton, and natural-gradient variants. In large networks, however, the exact Riccati recursion remains too expensive, so LLQR uses it as a reference objective rather than as the deployed optimizer.

The scalable relaxation parameterizes the update by a structured inverse preconditioner

$$\delta \theta_i = -U_i \nabla_{\theta_i} L(\theta^k), \qquad U = \operatorname{diag}(U_0, \ldots, U_{N-1}),$$

where each U_i may be diagonal, Kronecker-factored, E-KFAC-like, or another structured block. Substituting this restricted update into the LQR objective yields a direct objective over U:

$$\begin{aligned} \min_U \quad & \nabla_{x_N} \ell(x_N)^\top \delta x_N + \frac{1}{2} \delta x_N^\top Q_N \delta x_N + \sum_{i=0}^{N-1} \left[\frac{1}{2} \delta x_i^\top Q_i \delta x_i + \frac{1}{2} \delta \theta_i^\top R_i \delta \theta_i + \delta \theta_i^\top M_i \delta x_i \right] \\ \text{s.t.} \quad & \delta x_{i+1} = A_i \delta x_i - B_i U_i \nabla_{\theta_i} L(\theta^k), \qquad \delta x_0 = 0. \end{aligned} \tag{18}$$

Thus, structure is imposed on the reusable inverse action U, not on the curvature model before the layerwise objective is derived. This distinction is important for LLQR +SAM: the learned geometry still comes from a layer-coupled objective that encodes the dense quadratic model through the forward dynamics, while the deployed update only requires applying U to gradients.

In the LLQR +SAM setting, U should be viewed as a slowly updated transport geometry. Between preconditioner refreshes, it is held fixed and applied cheaply to the current gradient. This makes the method compatible with standard optimizers and with perturb-and-recompute schemes: SAM determines a local sharpness-aware gradient, while LLQR supplies the anisotropic geometry used to probe and/or transport that gradient.

C Experiments Details

C.1 Standard Supervised Training Benchmarks

Shared choices and tuning. Across the main experiments, LLQR uses NGD-induced divergence with E-KFAC blocks, following Dufort-Labbé et al. (2026). We also test Newton-induced divergence; Appendix D.1 reports the comparison for LLQR and LLQR+SAM in Table 9. We tune LLQR and SAM variants independently, starting from their recommended ranges. LLQR+SAM and LLQR+F-SAM reuse the component settings without joint tuning; the radius choice for LLQR+ASAM is described below. LLQR+SAM introduces no additional regularization term; baseline weight decay and data augmentation are retained.

CIFAR protocol. CIFAR-10/100 training uses a 200-epoch cosine schedule with initial learning rate 0.05, Polyak momentum 0.9, batch size 128, random crops, horizontal flips, normalization, and Cutout (Devries & Taylor, 2017). Weight decay is selected from $\{10^{-4}, 5\times10^{-4}, 10^{-3}\}$ by baseline best performance. SAM uses $\rho = 0.1$ on CIFAR-10 and $\rho = 0.2$ on CIFAR-100 (Foret et al., 2021; Li & Giannakis, 2023; Mi et al., 2022; Li et al., 2024); FisherSAM uses $\rho = 0.1$ and inverse-Fisher regularization $\eta = 0.1$ (Kim et al., 2022). LLQR retains the outer SGDM recipe and uses inner batch size 128, preconditioner learning rate 10^{-3}, 50 momentum-solver steps, and an update interval of 500 optimizer steps (Dufort-Labbé et al., 2026). NGD- and Newton-induced geometries use EMA decays 0.95 and 0.9, respectively. Table 1 reports best top-1 test accuracy over five independent seeds (mean $\pm$ std). All runs use NVIDIA L40S GPUs.

TinyImageNet protocol. The TinyImageNet-200 CNN experiments use the same training protocol and optimizer hyperparameters as the CIFAR benchmarks. Table 2 compares ResNet-18 and WRN-28-10 with best top-1 test accuracy (mean $\pm$ std over three seeds). Table 6 reports current estimates of the same metric for the ResNet-50 factorial ablation under a planned five-seed protocol. The LLQR variants in the ResNet-50 ablation use NGD-induced geometry. All TinyImageNet experiments were run on NVIDIA L40S GPUs.

SAM-variant protocol. Table 3 follows the CIFAR protocol above. F-SAM (Li et al., 2024) uses the recommended exponential moving average of gradient accumulation: $\lambda = 0.9$ for WRN-28-10 and PyramidNet-110, and $\lambda = 0.6$ for the other architectures. ASAM uses $\rho = 4$, selected by tuning on CIFAR-100/ResNet-18, and scaling offset $\eta = 0.01$. For GSAM, tuning selected $(\rho_{\max}, \rho_{\min}) = (0.5, 0.1)$; we use $\alpha = 0.01$, non-adaptive perturbations, and `minimize_fp=true`.

For LLQR+ASAM, the learned inverse metric U_t interacts with ASAM's parameter-dependent diagonal scaling D_{θ_t}. With minibatch gradient g_t, we use the combined perturbation

$$\epsilon_t = \rho \frac{D_{\theta_t}^2 U_t g_t}{\|D_{\theta_t} U_t g_t\|_2}.$$

Introducing U_t changes the effect of the normalization, so we retuned ρ over the same candidate set used for ASAM and selected $\rho = 2$. The perturbed gradient is then transported by U_t as in LLQR+SAM.

ImageNet protocol. For ResNet-50/ImageNet (Deng et al., 2009), we use $\rho = 0.075$ for SAM and $\lambda = 0.95$ for F-SAM, following Li et al. (2024). All ImageNet experiments were run on NVIDIA A100 GPUs.

IWSLT14 German-to-English protocol. We use the fairseq Transformer recipe for IWSLT14 German-to-English (Ott et al., 2018). The SAM radius is tuned over $[0.0005, 0.2]$, with $\rho = 0.005$ giving the best validation result; perturbations start after learning-rate warmup. The LLQR preconditioner retains the vision hyperparameters except for EMA decay 0.925. Table 5 reports best validation BLEU (mean $\pm$ std, five seeds). All runs use NVIDIA L40S GPUs.

Wall-clock controls. The comparisons in Appendix D.2 retain the dataset recipes above. Extended CIFAR and TinyImageNet runs adjust the cosine schedule to the longer training horizon. For ImageNet, we truncate LLQR+SAM to 490k steps and compare with SAM at 500k steps. Each table gives the training horizons and relative times. Performance uncertainties are standard deviations over five seeds for CIFAR and IWSLT14, and three seeds for ImageNet and TinyImageNet.

C.2 Sensitivity and resource controls

Memory control. The LLQR metric update runs between parameter updates, with its working batch configured to fit the available GPU memory. We use chunking only for PyramidNet, accumulating the metric objective over smaller portions of the same batch to trade runtime for peak memory.

D Additional Experimental Results

D.1 NGD- and Newton-induced divergences

We also evaluate Newton-induced divergence on CIFAR-10/100 and TinyImageNet-200. Table 9 compares the NGD- and Newton-induced variants of LLQR and LLQR+SAM using E-KFAC blocks and the protocols in Appendix C.1. The metric EMA decays are 0.95 for NGD and 0.9 for Newton, so this comparison includes the divergence-specific EMA choice.

Table 9: NGD- and Newton-induced divergence comparison. Entries are best top-1 test accuracy (%, mean ± std), over five seeds for CIFAR-10/100 and three seeds for TinyImageNet-200. Blue$^{\circ}$ denotes NGD-induced divergence; orange$^{\triangle}$ denotes Newton-induced divergence. Bold marks the highest mean in each row.

	LLQR		LLQR+SAM	
Architecture	NGD$^{\circ}$	Newton$^{\triangle}$	NGD$^{\circ}$	Newton$^{\triangle}$
CIFAR-10				
VGG16-BN	95.27 ± 0.20	95.41 ± 0.11	$\mathbf{95.92} \pm 0.07$	95.73 ± 0.08
ResNet-18	96.37 ± 0.11	96.37 ± 0.11	$\mathbf{96.85} \pm 0.09$	96.78 ± 0.03
WRN-28-10	97.04 ± 0.09	97.10 ± 0.02	$\mathbf{97.64} \pm 0.08$	97.52 ± 0.07
PyramidNet-110	97.21 ± 0.09	97.26 ± 0.09	$\mathbf{97.86} \pm 0.04$	97.66 ± 0.08
CIFAR-100				
VGG16-BN	76.32 ± 0.36	76.31 ± 0.11	$\mathbf{78.14} \pm 0.27$	77.67 ± 0.27
ResNet-18	79.96 ± 0.22	79.83 ± 0.18	$\mathbf{81.82} \pm 0.14$	81.65 ± 0.10
WRN-28-10	82.80 ± 0.38	82.72 ± 0.25	$\mathbf{85.30} \pm 0.14$	85.05 ± 0.12
PyramidNet-110	84.40 ± 0.47	84.00 ± 0.16	$\mathbf{86.67} \pm 0.21$	86.53 ± 0.05
TinyImageNet-200				
ResNet-18	57.02 ± 0.49	57.62 ± 0.47	$\mathbf{62.24} \pm 0.49$	61.98 ± 0.22
WRN-28-10	60.08 ± 0.32	60.37 ± 0.33	$\mathbf{64.65} \pm 0.51$	64.59 ± 0.25

NGD-induced LLQR+SAM has the highest reported mean in all ten settings. For standalone LLQR, the ordering between divergences varies across architectures and datasets. These comparisons concern reported means and do not establish statistical significance. Unless specified otherwise, we default to NGD-induces geometry across all our experiments.

D.2 Matched and extended wall-clock comparisons

We compare the LLQR combinations with longer baseline runs on CIFAR, TinyImageNet, and IWSLT14, and truncate LLQR+SAM on ImageNet to match the SAM budget. The protocols are given in Appendix C.1; the tables report training horizons and relative times, including the remaining budget differences.

CIFAR controls. LLQR+SAM exceeds the extended SAM mean in all eight settings, by 0.07–0.13 percentage points on CIFAR-10 and 0.39–0.90 on CIFAR-100 (Table 10). It also exceeds the extended plain LLQR mean in each setting.

ImageNet, TinyImageNet, and IWSLT14. On ImageNet, LLQR+SAM improves mean accuracy over SAM by 0.71 points at nearly the same elapsed time (Table 11). On TinyImageNet, the NGD- and Newton-induced combinations exceed the extended SAM mean by 0.35 and 0.09 points, respectively (Table 12). The IWSLT14 gains are more modest: +0.05 BLEU over extended LLQR at the same mean wall time, and +0.36 over extended SAM, which receives 9% more time (Table 13).

Table 10: CIFAR matched and extended wall-clock controls. Entries are best top-1 test accuracy (%, mean ± std, five seeds). Times are relative to the 200-epoch LLQR+SAM run for the same dataset and architecture. LLQR uses NGD-induced geometry.

Architecture	Metric	SAM	LLQR	LLQR+SAM
CIFAR-10				
VGG16-BN	Accuracy	95.79 ± 0.08	95.50 ± 0.07	$\mathbf{95.92} \pm 0.07$
	Epochs	292	277	200
	Rel. time	$1.02\times$	$1.08\times$	$1.00\times$
ResNet-18	Accuracy	96.75 ± 0.05	96.49 ± 0.15	$\mathbf{96.85} \pm 0.09$
	Epochs	267	282	200
	Rel. time	$1.02\times$	$1.03\times$	$1.00\times$
WRN-28-10	Accuracy	97.55 ± 0.02	97.20 ± 0.04	$\mathbf{97.64} \pm 0.08$
	Epochs	256	328	200
	Rel. time	$1.03\times$	$1.00\times$	$1.00\times$
PyramidNet-110	Accuracy	97.79 ± 0.07	97.30 ± 0.08	$\mathbf{97.86} \pm 0.04$
	Epochs	272	292	200
	Rel. time	$1.01\times$	$1.00\times$	$1.00\times$
CIFAR-100				
VGG16-BN	Accuracy	77.24 ± 0.32	76.34 ± 0.18	$\mathbf{78.14} \pm 0.27$
	Epochs	267	277	200
	Rel. time	$0.99\times$	$1.13\times$	$1.00\times$
ResNet-18	Accuracy	81.20 ± 0.15	79.52 ± 0.26	$\mathbf{81.82} \pm 0.14$
	Epochs	251	282	200
	Rel. time	$0.98\times$	$1.06\times$	$1.00\times$
WRN-28-10	Accuracy	84.87 ± 0.13	82.22 ± 0.87	$\mathbf{85.30} \pm 0.14$
	Epochs	251	333	200
	Rel. time	$1.01\times$	$1.01\times$	$1.00\times$
PyramidNet-110	Accuracy	86.28 ± 0.23	83.95 ± 0.20	$\mathbf{86.67} \pm 0.21$
	Epochs	272	287	200
	Rel. time	$1.01\times$	$0.99\times$	$1.00\times$

Table 11: ResNet-50/ImageNet wall-clock comparison. Entries are best top-1 accuracy (%, mean ± std, 3 seeds). Relative times use the SAM run as reference.

Method	Steps	Accuracy (%)	Rel. time
SAM	500k	77.96 ± 0.01	$1.000\times$
LLQR+SAM	490k	$\mathbf{78.67} \pm 0.08$	$0.996\times$

Table 12: ResNet-18/TinyImageNet-200 wall-clock controls for Tables 2 and 9. Entries are best top-1 accuracy on the same evaluation split (%, mean ± std, 3 seeds). Relative times use the 200-epoch LLQR+SAM run as reference.

Method	Epochs	Accuracy (%)	Rel. time
SAM	234	61.89 ± 0.37	$1.03\times$
LLQR+SAM	200	$\mathbf{62.24} \pm 0.49$	$1.00\times$

Table 13: IWSLT14 German-to-English wall-clock comparison. Entries are best validation BLEU (mean ± std, 5 seeds). Relative times use the LLQR+SAM run as reference.

Method	Epochs	BLEU	Rel. time
SAM	152	34.21 ± 0.19	$1.09\times$
LLQR	193	34.52 ± 0.11	$1.00\times$
LLQR+SAM	133	$\mathbf{34.57} \pm 0.09$	$1.00\times$

D.3 SAM VARIANTS AND COMPOSITION.

Tables 3, 14 and 15 extend the comparison to F-SAM, FisherSAM, ASAM, GSAM, and the LLQR combinations. For the matched-wall-clock runs, LLQR+ASAM has the highest mean on five of eight settings. LLQR+F-SAM leads on PyramidNet-110/CIFAR-100, and LLQR+SAM on WRN-28-10/CIFAR-100; extended ASAM leads on ResNet-18/CIFAR-100. An LLQR-combined method thus has the highest mean in seven of eight settings.

Relative to the corresponding extended variant, LLQR+F-SAM improves the mean in all eight settings ($+0.07$ to $+0.81$ points). LLQR+ASAM improves it in seven, including $+0.58$ points on both VGG16-BN benchmarks, but remains 0.15 points below ASAM on ResNet-18/CIFAR-100. The effect of combining LLQR with ASAM therefore depends on the architecture and dataset.

Table 14: CIFAR-10 SAM variants at comparable wall-clock budgets. Entries are best top-1 test accuracy (%, mean $\pm$ std, 5 seeds). Relative times use the 200-epoch LLQR+SAM reference for each architecture in Table 10.

Architecture	Metric	F-SAM	FisherSAM	ASAM	GSAM	LLQR+SAM	LLQR+F-SAM	LLQR+ASAM
VGG16-BN	Accuracy	95.73 ± 0.06	95.74 ± 0.08	95.37 ± 0.06	95.40 ± 0.12	95.92 ± 0.07	95.81 ± 0.11	$\mathbf{95.95} \pm 0.06$
	Epochs	292	292	292	292	200	200	200
	Rel. time	$1.07\times$	$1.01\times$	$1.03\times$	$1.06\times$	$1.00\times$	$1.07\times$	$1.03\times$
ResNet-18	Accuracy	96.78 ± 0.11	96.81 ± 0.07	96.90 ± 0.13	96.69 ± 0.04	96.85 ± 0.09	96.88 ± 0.11	$\mathbf{97.05} \pm 0.08$
	Epochs	267	267	267	267	200	200	200
	Rel. time	$1.08\times$	$1.02\times$	$1.03\times$	$1.06\times$	$1.00\times$	$1.04\times$	$1.02\times$
WRN-28-10	Accuracy	97.55 ± 0.05	97.54 ± 0.12	97.74 ± 0.12	97.49 ± 0.08	97.64 ± 0.08	97.68 ± 0.07	$\mathbf{97.79} \pm 0.11$
	Epochs	256	256	256	256	200	200	200
	Rel. time	$1.07\times$	$1.03\times$	$1.03\times$	$1.04\times$	$1.00\times$	$1.07\times$	$1.04\times$
PyramidNet-110	Accuracy	97.81 ± 0.06	97.81 ± 0.06	97.90 ± 0.12	97.60 ± 0.09	97.86 ± 0.04	97.89 ± 0.06	$\mathbf{97.96} \pm 0.10$
	Epochs	272	272	272	272	200	200	200
	Rel. time	$1.04\times$	$1.02\times$	$1.02\times$	$1.02\times$	$1.00\times$	$1.02\times$	$1.03\times$

Table 15: CIFAR-100 SAM variants at comparable wall-clock budgets. Entries are best top-1 test accuracy (%, mean $\pm$ std, 5 seeds). Relative times use the 200-epoch LLQR+SAM reference for each architecture in Table 10.

Architecture	Metric	F-SAM	FisherSAM	ASAM	GSAM	LLQR+SAM	LLQR+F-SAM	LLQR+ASAM
VGG16-BN	Accuracy	77.41 ± 0.25	77.36 ± 0.28	78.10 ± 0.26	76.29 ± 0.27	78.14 ± 0.27	78.22 ± 0.31	$\mathbf{78.68} \pm 0.22$
	Epochs	267	267	267	267	200	200	200
	Rel. time	$1.04\times$	$0.98\times$	$0.99\times$	$1.03\times$	$1.00\times$	$1.05\times$	$1.01\times$
ResNet-18	Accuracy	81.27 ± 0.31	81.05 ± 0.15	$\mathbf{82.41} \pm 0.10$	80.99 ± 0.30	81.82 ± 0.14	82.00 ± 0.28	82.26 ± 0.35
	Epochs	251	251	251	251	200	200	200
	Rel. time	$1.01\times$	$0.99\times$	$0.99\times$	$1.03\times$	$1.00\times$	$1.06\times$	$1.00\times$
WRN-28-10	Accuracy	85.14 ± 0.16	84.73 ± 0.25	85.20 ± 0.22	85.12 ± 0.22	$\mathbf{85.30} \pm 0.14$	85.21 ± 0.20	85.21 ± 0.23
	Epochs	251	251	251	251	200	200	200
	Rel. time	$1.03\times$	$1.00\times$	$1.01\times$	$1.02\times$	$1.00\times$	$1.05\times$	$1.03\times$
PyramidNet-110	Accuracy	86.43 ± 0.26	86.22 ± 0.23	86.49 ± 0.16	86.16 ± 0.31	86.67 ± 0.21	$\mathbf{86.76} \pm 0.13$	86.53 ± 0.09
	Epochs	272	272	272	272	200	200	200
	Rel. time	$1.02\times$	$1.03\times$	$1.02\times$	$1.02\times$	$1.00\times$	$1.03\times$	$1.02\times$

D.4 GEOMETRY-AWARE PROBE AND TRANSPORT ABLATIONS

On CIFAR-100/ResNet-18, the factorial ablation separates geometry-aware probing from LLQR outer preconditioning (Table 16). LLQR uses no sharpness probe. LLQR+SAM (probe only) uses the learned preconditioner only in the perturbation step; LLQR+SAM (outer only) uses a Euclidean SAM probe and preconditions only the outer gradient. Full LLQR+SAM uses the learned metric in both steps. All rows follow the CIFAR protocol in Appendix C.1.

D.5 SCALABILITY

Our LLQR implementation attaches a blockwise preconditioner to network layers. The storage footprint is therefore determined by the chosen parameterization of each block rather than by LLQR as a framework. Richer blocks can encode more geometry but require more optimizer state, while

Table 16: CIFAR-100/ResNet-18 factorial ablation. Entries are best top-1 test accuracy (%, mean ± std, five seeds). Gains are percentage points over SGDM (78.97%). For reference, LLQR (no SAM probe) is at 79.96%.

LLQR probe	LLQR outer preconditioning	Method	Accuracy (%) ↑	Gain vs. SGDM
No	No	SAM	81.13 ± 0.21	$+2.16$
No	Yes	LLQR+SAM (outer only)	$\mathbf{81.82} \pm 0.11$	$\mathbf{+2.85}$
Yes	No	LLQR+SAM (probe only)	81.23 ± 0.14	$+2.26$
Yes	Yes	LLQR+SAM	$\mathbf{81.82} \pm 0.14$	$\mathbf{+2.85}$

Table 17: Cross-model analytical fp32 storage for e-KFAC and diag-KFAC preconditioners under ImageNet-scale ViT geometry. Model counts include heads and classification layers; #precond reports the number of stored preconditioner scalars for each parameterization.

Model	Storage	S	#model params	#precond params	precond (MiB/dev.)	total (MiB/dev.)
ViT-Ti/16	e-kfac	1	5,427,080	24,235,082	92.45	113.15
ViT-Ti/16	e-kfac	4	5,427,080	24,235,082	23.11	43.82
ViT-Ti/16	diag-kfac	1	5,427,080	73,426	0.28	20.98
ViT-Ti/16	diag-kfac	4	5,427,080	73,426	0.07	20.77
ViT-B/16	e-kfac	1	86,567,656	387,826,882	1,479.44	1,809.67
ViT-B/16	e-kfac	4	86,567,656	387,826,882	369.86	700.09
ViT-B/16	diag-kfac	1	86,567,656	294,038	1.12	331.35
ViT-B/16	diag-kfac	4	86,567,656	294,038	0.28	330.51
ViT-L/16	e-kfac	1	304,326,632	1,367,114,178	5,215.13	6,376.04
ViT-L/16	e-kfac	4	304,326,632	1,367,114,178	1,303.78	2,464.70
ViT-L/16	diag-kfac	1	304,326,632	771,990	2.94	1,163.86
ViT-L/16	diag-kfac	4	304,326,632	771,990	0.74	1,161.65

simpler parameterizations reduce memory at the cost of expressivity. Table 17 summarizes the resulting fp32 storage across ViT scales for the preconditioner families considered in this scalability study.

For comparison, we also include a hypothetical `diag-kfac` structure, which approximates each Kronecker factor by its diagonal: $\mathrm{Diag}(a)\, X\, \mathrm{Diag}(b)$ for a reshaped kernel $X \in \mathbb{R}^{m \times n}$. This reduces the per-kernel preconditioner storage from dense Kronecker-style scaling to $O(m + n)$ scalars, providing a lightweight option for trading geometric expressivity for memory.

We also provide cadence results in Table 18 and Fig. 7, to be read as wall-clock overhead measurements, not as accuracy or convergence comparisons. As expected, the cost decreases when LLQR updates are performed less frequently, since fewer preconditioner refreshes are executed per epoch. This steady-update scaling trend is further exposed in Fig. 4.

D.6 ViT accuracy and continuation results

Shared training recipes. Within each comparison, LLQR+SAM reuses SAM's learning rate, weight decay, batch size, and learning-rate and radius schedules without additional tuning of these shared hyperparameters. The TinyImageNet/TinyViT placement comparison uses relaxed LLQR (EKFAC), AdamW, 300 epochs, and seeds 41–44. CUB-200 fine-tuning uses pretrained ViT-Large/16 at 384-pixel resolution, 20 epochs, the same four seeds, learning rate 5×10^{-5}, weight decay 0.05, cosine decay without warmup, effective batch size 128, and SAM radius 0.05. Its scoped-EKFAC implementation structures the last transformer block and classifier, with diagonal-Fisher fallback elsewhere; it is a PyTorch surrogate, not exact full-network LLQR.

Table 19 shows a 0.77-point TinyViT gain with LLQR in both locations. The outer-only ablation gains 2.62 ± 0.45 points over SAM (paired seed SD; 95% paired t-interval $[1.89, 3.34]$) and retains a

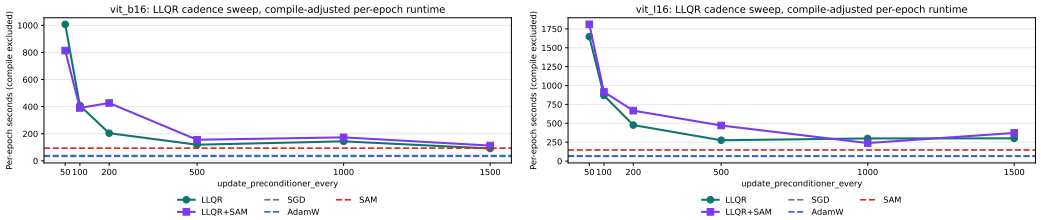

Figure 7: LLQR cadence sweep for ViT-B/16 and ViT-L/16. Each panel shows compile-adjusted seconds per epoch versus the LLQR update interval, with first-order baselines as horizontal references. Note that the ImageNet ResNet-50 runs in Fig. 1 use 1500-step updates.

Table 18: Measured LLQR cadence details for ViT-B/16 and ViT-L/16. Times are compile-adjusted seconds per epoch, excluding the first preconditioner-update overhead. Baseline rows report first-order optimizer runtimes; LLQR rows additionally report ratios relative to SGD and SAM on the same architecture.

Arch	Method	Update interval	Updates	Epoch time	vs. SGD	vs. SAM	Steady update
ViT-B/16	SGD	–	–	33.20	1.00×	0.35×	–
ViT-B/16	AdamW	–	–	38.51	1.16×	0.41×	–
ViT-B/16	SAM	–	–	93.84	2.83×	1.00×	–
ViT-B/16	LLQR	50	32	1007.31	30.34×	10.73×	59.38
ViT-B/16	LLQR	100	16	408.09	12.29×	4.35×	44.25
ViT-B/16	LLQR	200	8	204.11	6.15×	2.18×	39.38
ViT-B/16	LLQR	500	4	118.97	3.58×	1.27×	38.85
ViT-B/16	LLQR	1000	2	143.95	4.34×	1.53×	86.55
ViT-B/16	LLQR	1500	2	91.24	2.75×	0.97×	42.52
ViT-B/16	LLQR+SAM	50	32	813.98	24.51×	8.67×	46.05
ViT-B/16	LLQR+SAM	100	16	390.59	11.76×	4.16×	39.91
ViT-B/16	LLQR+SAM	200	8	426.81	12.85×	4.55×	85.33
ViT-B/16	LLQR+SAM	500	4	155.50	4.68×	1.66×	40.58
ViT-B/16	LLQR+SAM	1000	2	173.25	5.22×	1.85×	87.41
ViT-B/16	LLQR+SAM	1500	2	112.65	3.39×	1.20×	41.63
ViT-L/16	SGD	–	–	62.59	1.00×	0.43×	–
ViT-L/16	AdamW	–	–	67.84	1.08×	0.46×	–
ViT-L/16	SAM	–	–	146.54	2.34×	1.00×	–
ViT-L/16	LLQR	50	32	1647.13	26.32×	11.24×	97.46
ViT-L/16	LLQR	100	16	868.47	13.88×	5.93×	97.34
ViT-L/16	LLQR	200	8	476.77	7.62×	3.25×	97.52
ViT-L/16	LLQR	500	4	275.63	4.40×	1.88×	95.18
ViT-L/16	LLQR	1000	2	299.48	4.79×	2.04×	96.46
ViT-L/16	LLQR	1500	2	301.14	4.81×	2.06×	95.91
ViT-L/16	LLQR+SAM	50	32	1809.84	28.92×	12.35×	95.78
ViT-L/16	LLQR+SAM	100	16	914.70	14.62×	6.24×	96.36
ViT-L/16	LLQR+SAM	200	8	667.77	10.67×	4.56×	96.20
ViT-L/16	LLQR+SAM	500	4	470.89	7.52×	3.21×	96.57
ViT-L/16	LLQR+SAM	1000	2	238.29	3.81×	1.63×	96.11
ViT-L/16	LLQR+SAM	1500	2	372.23	5.95×	2.54×	95.50

2.62-point final-epoch advantage. Its runtime is 12.54 hours versus 6.75 for SAM, so this is an equal-epoch comparison. On CUB-200, scoped EKFAC LLQR+SAM improves mean best validation accuracy by 0.188 points over SAM and 0.271 over the same-backend LLQR baseline. These small differences are descriptive; official-test means are 89.61% for LLQR+SAM and 89.74% for SAM.

Continuation from the same source checkpoint. For CUB-200 ViT-Base and ViT-Large, we branch SAM and a diagonal-Fisher LLQR+SAM surrogate from each recorded AdamW, SAM, and LLQR+SAM source checkpoint. Both arms reset optimizer and scheduler state once and use 100 epochs with the same cubic schedule $\rho_e = (e/99)^3$, $e = 0, \ldots, 99$. All comparisons use seed 41 and the 1,200-image selection-validation split. Sources retain their original validation-selection history.

Table 19: **ViT accuracy with shared SAM hyperparameters.** Best validation top-1, mean ± sample SD over four matched seeds. Gains are percentage points relative to SAM within each panel. Outer-only retains SAM's isotropic perturbation.

Method	Best validation (%)	Gain over SAM (pp)
TinyImageNet / TinyViT, relaxed LLQR (EKFAC)		
SAM	48.28 ± 0.62	—
LLQR+SAM (both locations)	49.05 ± 0.24	$+0.77$
LLQR+SAM (outer-only)	$\mathbf{50.90} \pm 0.39$	$\mathbf{+2.62}$
CUB-200 / pretrained ViT-Large/16		
AdamW	89.271 ± 0.550	-0.167
SAM	89.438 ± 0.172	—
Scoped EKFAC LLQR	89.354 ± 0.300	-0.083
Scoped EKFAC LLQR+SAM	$\mathbf{89.625} \pm 0.388$	$\mathbf{+0.188}$

Table 20: **CUB-200 source-matched continuation.** Final validation top-1 after 100 epochs. All six source/model pairs are shown. Differences use unrounded values; their mean is descriptive across sources at one seed. LLQR+SAM uses the diagonal-Fisher surrogate.

Model	Source	SAM (%)	LLQR+SAM (%)	Difference (pp)
ViT-Base	AdamW	88.75	88.50	-0.25
ViT-Base	SAM	88.08	**88.75**	$\mathbf{+0.67}$
ViT-Base	LLQR+SAM	87.92	**88.08**	$\mathbf{+0.17}$
ViT-Large	AdamW	88.83	**88.92**	$\mathbf{+0.08}$
ViT-Large	SAM	89.58	**89.83**	$\mathbf{+0.25}$
ViT-Large	LLQR+SAM	89.17	**89.75**	$\mathbf{+0.58}$
Mean paired difference				$\mathbf{+0.25}$

LLQR+SAM finishes above SAM in five of six pairs (Table 20). From the ViT-Base SAM source, best accuracy is 88.92% versus 88.17% for SAM ($+0.75$ points). ViT-Large from its LLQR+SAM source reaches 90.17%, 0.58 points above that source. These are exploratory validation results; source-relative gains include additional training and do not isolate the effect of radius expansion.

Additional continuation evidence. In the earlier discrete CUB continuation, ViT-Large from the SAM source reaches 90.00% with LLQR+SAM versus 89.08% with SAM at $\rho = 0.5$. Separately, EKFAC TinyViT LLQR+SAM exceeds SAM in all 24 matched source/radius cells, with endpoint advantages of 6.51, 7.45, and 0.79 points from AdamW, SAM, and LLQR+SAM sources. These TinyViT trajectories remain below their source accuracies. Adaptive stopping gives unequal work, and LLQR+SAM takes 2.9–4.5 times the summed stage runtime; this supports relative resistance to degradation under that recipe, not equal-time superiority.

E Limitations and Scope

Scope of the quadratic mechanism. Section 4 analyzes an idealized two-scale quadratic loss with $H = \bar{H} + H_\epsilon$, background curvature $\bar{H} \succ 0$, localized curvature $H_\epsilon \succeq 0$, and a fixed prescribed metric $U = \bar{H}^{-1}$. The quadratic recursion is exact under these assumptions for the plain-SGD update in Eq. (1); the scalar comparison with vanilla SAM additionally uses a shared eigenbasis and motion along a single eigendirection. Whitening (Appendix A.2) handles noncommuting curvature within the same fixed quadratic model. This illustrates how preconditioning changes SAM's response to localized sharpness relative to the surrounding geometry, without establishing this decomposition or learned metric alignment in neural-network training.

The local model does not describe motion beyond the quadratic neighborhood; the hovering-envelope comparison is not a general escape or convergence guarantee. Appendix A.7 also idealizes stochastic residence times in wells. The separate stationarity statement in Appendix A.1 assumes a positive-definite metric frozen within each perturb-and-recompute step, uniform spectral and regu-

larity bounds, stochastic-gradient conditions, and prescribed decaying schedules. These assumptions are not established for the deployed training recipes; stationarity alone does not imply convergence to a preferred basin or improved generalization.

Empirical coverage. Our experiments cover standard supervised vision benchmarks and one sequence-modeling task, but not large language model pre-training or fine-tuning, nor every architecture family. Benchmark gains do not by themselves validate the proposed curvature decomposition or identify their cause. The noisy toy experiments illustrate the mechanism only in their constructed landscapes.

Computation and storage. LLQR+SAM inherits SAM's additional gradient evaluation and adds periodic metric fitting, metric application, and structured optimizer state. Block structure trades geometric expressivity for memory and computation; refresh cadence trades fitting cost for metric freshness, and chunking can trade peak working memory for execution overhead. Practical cost therefore depends on model scale, implementation, and the memory budget. The cadence measurements in Appendix D.5 quantify runtime overhead, not accuracy or convergence at each cadence.