OPT2025: 17th Annual Workshop on Optimization for Machine Learning

# A Unified Noise-Curvature View of Loss of Trainability

**Gunbir Singh Baveja** GBAVEJA@STUDENT.UBC.CA
**Mark Schmidt**[†] SCHMIDTM@CS.UBC.CA
*University of British Columbia, Canada CIFAR AI Chair (Amii)*[†]

## Abstract

Loss of trainability (LoT) in continual learning occurs when gradient steps no longer yield improvement as tasks evolve, so accuracy stalls or degrades despite adequate capacity and supervision. We analyze LoT incurred with Adam through an optimization lens and find that single indicators such as Hessian rank, sharpness level, weight or gradient norms, gradient-to-parameter ratios, and unit-sign entropy are not reliable predictors. Instead we introduce two complementary criteria: a batch-size-aware gradient-noise bound and a curvature volatility-controlled bound that combine into a per-layer predictive threshold that anticipates trainability behavior. Using this threshold, we build a simple per-layer scheduler that keeps each layer's effective step below a safe limit, stabilizing training and improving accuracy across concatenated ReLU (CReLU), Wasserstein regularization, and L2 weight decay, with learned learning-rate trajectories that mirror canonical decay.

## 1. Introduction

Continual learning (CL) asks models to keep making progress under non-stationary data streams, where optimization must remain effective as parameters evolve. A central failure mode is the *loss of trainability (LoT)*: over a sequence of tasks, the mapping from gradient steps to improvement degrades, so training slows or stalls even when capacity and supervision are adequate (Lyle et al. 2023; Dohare et al. 2024). This is distinct from catastrophic forgetting, which concerns retention rather than progress on new data. LoT has been observed across architectures and benchmarks (e.g., randomized labels, input permutations, class-incremental regimes), suggesting it is a general property of SGD dynamics in non-stationary settings rather than a peculiarity of any single model or dataset, and arises even without explicit task boundaries (Lewandowski et al. 2024a; Rohani et al. 2025).

Mechanistically, several factors have been implicated—changes in curvature and sharpness (Dohare et al. 2024), decaying Hessian rank (Lewandowski et al. 2024b), reductions in gradient noise (Berariu et al. 2021), or activation pathologies such as saturation and persistent linearization measured by unit-sign entropy (Abbas et al. 2023; Lewandowski et al. 2024a). Yet in practice, these metrics do not behave consistently across optimizers and hyperparameters. This inconsistency matters because continual learning typically precludes *lifetime tuning*: we cannot reset or retune learning rates and regularizers for each task sequence (Ash et al. 2020; Mesbahi et al. 2025). A robust explanation must therefore function under non-trivial hyperparameters, and it must explain counterexamples where single-metric stories break down.

In this work we adopt a two-signal view of trainability. We read a *batch-size–aware* gradient-noise signal together with a *sharpness-volatility* signal based on the normalized (Adam-adjusted) sharpness; together these yield a per-layer *predictive threshold* on the effective step. We then im-

plement a simple per-layer Adam scheduler that keeps each layer's effective step below a volatility-inflated bound, so updates are neither gradient- nor curvature-noise dominated[1].

To evaluate this diagnosis and mitigation in a clean setting, we hold hyperparameters fixed (*no lifetime tuning*) and use configurations that surface complementary failure routes: CReLU (Shang et al. 2016), which preserves bidirectional responses and often improves gradient flow yet can undergo curvature collapse; Wasserstein regularization (Lewandowski et al. 2024b), which curbs drift and stabilizes early tasks but degrades on longer horizons without step-size control; and standard L2 weight decay, the simplest scale regularizer for ReLU networks.

## 2. Related Work

**Diagnosing loss of trainability** A growing body of work characterizes *loss of trainability* (LoT) as a byproduct of SGD optimization dynamics that emerge under non-stationarity. Several studies emphasize geometric collapse: Lewandowski et al. (2024b) argue that reduced curvature directions explain the onset of LoT, connecting Hessian rank degeneration to reduced learning. Ziyin (2024) analyze symmetries in neural network objectives and show that SGD's update rule, with L2 weight decay, makes symmetry saddles attractive and draws networks toward sparse and low-rank solutions. Other accounts highlight sharpness and instability (Dohare et al. 2024), and the unbounded growth of sharp directions (Foret et al. 2021a). Lyle et al. (2024) suggests that unbounded norm growth or decaying gradient norm may be factors that lead to LoT. Beyond geometry and noise, activation usage has been implicated: saturation and persistent linearization reduce the diversity of unit responses, captured by unit-sign entropy (Abbas et al. 2023; Lewandowski et al. 2024a). However, these mechanisms often coexist and vary across setups. In isolation, they are inconsistent and do not fully explain LoT. Moreover, because LoT is generally assessed as a reduction in average task accuracy, perceived loss of trainability might occur without an actual loss of plasticity if Adam functions in a regime where noise-dominated steps prevail across different tasks (see Fig. 3).

**Step-size as noise injection and plasticity.** Step size plays a central role in plasticity: it acts as controllable noise that trades off exploration and stability. Berariu et al. (2021) study a fine-tuning generalization gap and hypothesize that pretraining reduces gradient noise in the tuning stage, leading to less exploration and sharper minima; they show that increasing the step size (amplifying gradient randomness) during tuning substantially reduces the gap (Berariu et al. 2021, Fig. 7). Subsequent analyses likewise emphasize the step size as a key factor for both causing and mitigating loss of plasticity (Lewandowski et al. 2024b). Our results refine this view in continual learning: larger steps are not uniformly beneficial. When gradients become noise-dominated or curvature becomes fragile, *decaying* the step size restores trainability—for example, under L2 and Wasserstein we often reduce the base learning rate before briefly re-warming, whereas CReLU tolerates mild warm-ups only after volatility drops (Sec. 5). This aligns with classical noise-scale theory (Mandt et al. 2017; McCandlish et al. 2018; Smith et al. 2018) and with widely used decaying schedules (Goyal et al. 2017; Loshchilov et al. 2017; Huang et al. 2017; Vaswani et al. 2017; Devlin et al. 2019; Chen et al. 2017).

1. Here, *noise-dominated* $\approx$ high gradient randomness; *curvature-dominated* $\approx$ high sharpness volatility across batches, so nominal steps alternate between descent and instability.

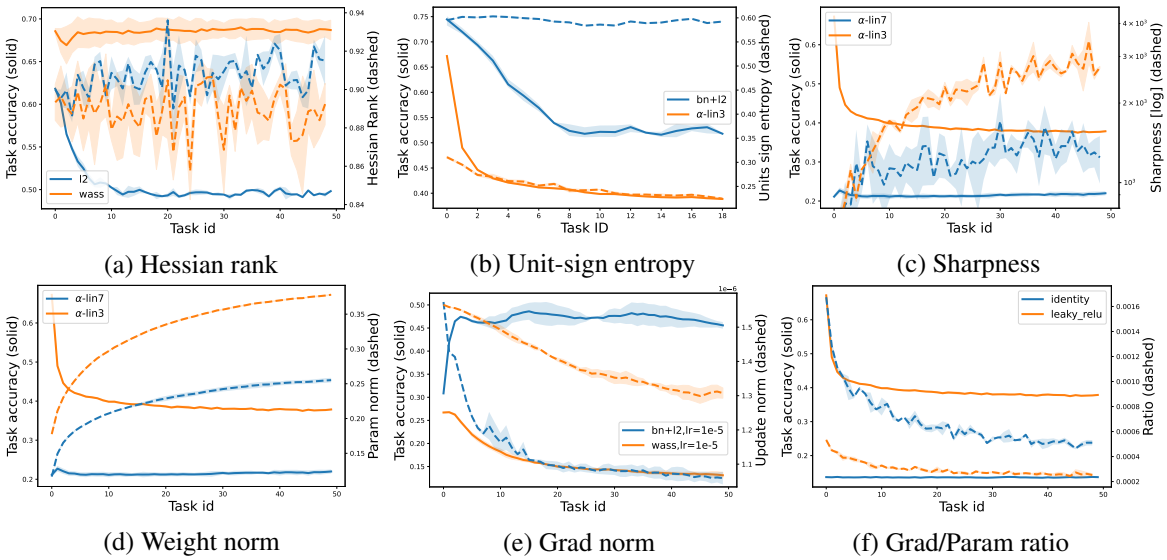


Figure 1: **Counterexamples.** Each panel shows a setting where a single-cause metric appears predictive in one configuration but fails in another.

## 3. No Consistent Single Explanation

We believe that no single metric provides a consistent explanation of LoT across architectures, or hyperparameters. Here we present experiments where each of the standard measures does not explain the LoT. We see that in Fig. 1 (a) Hessian rank stays high under L2 while accuracy collapses, yet Wasserstein remains stable despite similar rank; (b) *unit-sign entropy (USE)*: for BatchNorm with weight decay, task accuracy decays sharply while USE stays stable, whereas with LeakyReLU($\rho$=0.3) accuracy also decays but USE dips only slightly; (c) *sharpness*: LeakyReLU ($\rho$=0.7) vs. LeakyReLU ($\rho$=0.3) both see sharpness rise, yet trainability differs; (d) *weight norm*: the same pair shows similar growth, with differing trainability behavior; (e) *gradient norm*: decay appears in both BatchNorm with weight decay and Wasserstein, yet the configuration with faster decay can remain trainable; (f) *grad/param ratio*: it falls for deep linear and LeakyReLU networks, but only the latter loses trainability. These counterexamples motivate our unified view.

## 4. Two Signals that Unify & Predict LoT

Not all drops in task accuracy indicate the same underlying failure. We observe two distinct routes to trainability loss: (i) phases in which updates are *noise-dominated* so progress stalls despite adequate curvature, and (ii) phases in which curvature collapses (Ziyin 2024; Lewandowski et al. 2024b) or becomes *noisy*, so nominally similar steps flip between descent and instability.

**Critical step from gradient noise** Assume the loss is locally $L$-smooth along the Adam-preconditioned update direction. Let the minibatch gradient be $\widehat{g}_t = g_t + \xi_t$ with $\mathbb{E}[\xi_t] = 0$ and $\mathbb{E}\|\xi_t\|^2 = \sigma^2_{t,\mathrm{ps}}/B$. A quadratic expansion of $\ell$ along the update direction with step $\alpha_t$ gives $\ell(w_{t+1}) \lesssim \ell(w_t) -$

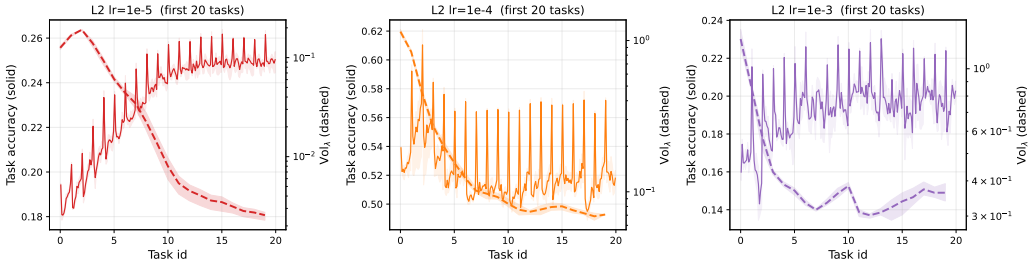


Figure 2: $\mathrm{Vol}_{\bar{\lambda}}$ growth precedes accuracy collapse for L2 with different learning rates.

$\alpha_t\langle g_t, \widehat{g}_t\rangle + \frac{1}{2}\alpha_t^2\lambda_t\|\widehat{g}_t\|^2$. Taking expectations yields

$$\mathbb{E}[\ell(w_{t+1}) - \ell(w_t)] \;\lesssim\; -\alpha_t\|g_t\|^2 + \tfrac{1}{2}\alpha_t^2\lambda_t\Big(\|g_t\|^2 + \tfrac{\sigma_{t,\mathrm{ps}}^2}{B}\Big),$$

where $\lambda_t \le L$. Thus a sufficient condition for expected descent is

$$\alpha_t \;<\; \frac{2}{L}\cdot\frac{\|g_t\|^2}{\|g_t\|^2 + \sigma_{t,\mathrm{ps}}^2/B} \;\le\; \frac{2}{L}\frac{B\|g_t\|^2}{\sigma_{t,\mathrm{ps}}^2}.$$

Defining the batch-size–aware critical step $\alpha_g^\star(t) := B\|g_t\|^2/\sigma_{t,\mathrm{ps}}^2$, we obtain the compact criterion

$$\alpha_t \;\le\; 2\,\alpha_g^\star(t)/L.$$

Empirically, $\alpha_g^\star$ hovers near 1 early (see $\alpha_g^\star$ in Fig. 3) and becomes more volatile late in tasks. As the effective step $\alpha_t$ drifts upward, the fraction of steps where $\alpha_t > \alpha_g^\star(t)$ grows. This noise-dominated phase is evident in L2 where accuracy deteriorates although the curvature signal remains consistent. (middle of Fig. 3).

**Critical step from sharpness volatility** Let $\ell : \mathbb{R}^d \to \mathbb{R}$ be the training loss, $H_t := \nabla^2\ell(w_t)$, and $\lambda_t := \lambda_{\max}(H_t)$. With Adam-like preconditioning we update $w_{t+1} = w_t - \alpha_t\,\widehat{g}_t^{(\mathrm{eff})}$, with $\widehat{g}_t^{(\mathrm{eff})} := D_t\,\widehat{g}_t$, and $D_t := \mathrm{diag}\Big(\frac{1}{\sqrt{\widehat{v}_t}+\epsilon}\Big)$, where $\widehat{g}_t$ is the first-moment estimate and $\widehat{v}_t$ the second-moment estimate. We track the *normalized* sharpness $\bar{\lambda}_t := \alpha_{\mathrm{agg},t}\lambda_t$, with $\alpha_{\mathrm{agg},t} = \eta_t/(\mathrm{RMS}(\sqrt{\widehat{v}_t}) + \epsilon)$ (see Sec. B.1). Because only one $\bar{\lambda}_t$ is observed per step, we estimate temporal statistics over a rolling window $W$ and define

$$\mu_t := \mathrm{EMA}_W[\bar{\lambda}], \qquad \sigma_t^2 := \mathrm{Var}_W[\bar{\lambda}], \qquad \mathrm{Vol}_{\bar{\lambda}}(t) := \frac{\sigma_t^2}{\mu_t + \varepsilon}\,. \tag{1}$$

Using the local model $\bar{\lambda}_s = \mu_t + \xi_s$ with $\mathbb{E}[\xi_s] = 0$ and $\mathrm{Var}[\xi_s] = \sigma_t^2$, a quadratic expansion of $\ell$ along the preconditioned direction gives $\Delta\ell_s \;\approx\; -\alpha_t\,\|D_t^{1/2}\widehat{g}_s\|^2 \;+\; \frac{1}{2}\alpha_t^2\,\bar{\lambda}_s\,\|D_t^{1/2}\widehat{g}_s\|^2$, so instability occurs when $\alpha_t\bar{\lambda}_s$ is too large. Controlling the probability of such events via a simple moment bound yields the sufficient condition and recovers the volatility-controlled critical step

$$\alpha_t \;\lesssim\; \frac{c_\delta\,\mu_t}{\mu_t^2 + \sigma_t^2} \;=\; \frac{1}{\kappa}\cdot\frac{1}{\mathrm{Vol}_{\bar{\lambda}}(t)}\,.$$

*Interpretation.* The bound shows that larger $\mu_t$ and smaller $\sigma_t^2$ make curvature along the preconditioned direction more *predictable*, permitting larger safe steps. Conversely, when $\alpha_{\mathrm{vol}}^\star(t)$ *decays* ($\sigma_t^2 \gg \mu_t$), the curvature signal weakens; persistent exceedance $\alpha_t > \alpha_{\mathrm{vol}}^\star(t)$ indicates curvature-noise–dominated updates and precedes loss of trainability (cf. Fig. 2).

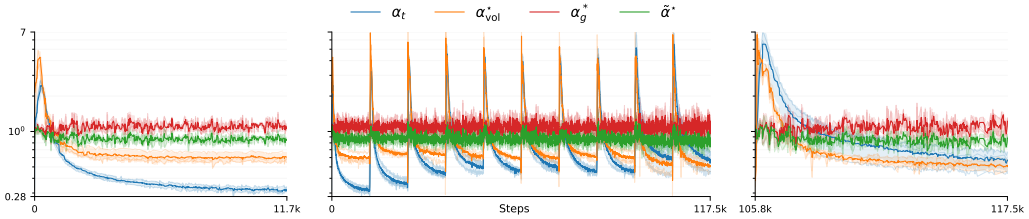


Figure 3: **Effective step vs. sharpness-aware safe bound.** We plot the first layer's effective step $\alpha_{t,l_1}$ with $\alpha^{\star}_{g,l_1}$, $\tilde{\alpha}^{\star}_{l_1}$, and $\alpha^{\star}_{\text{vol},l_1}$ for L2 ($\lambda = 10^{-3}$). Task 1 (left), Tasks $1-10$ (middle), and Task 10 (right).

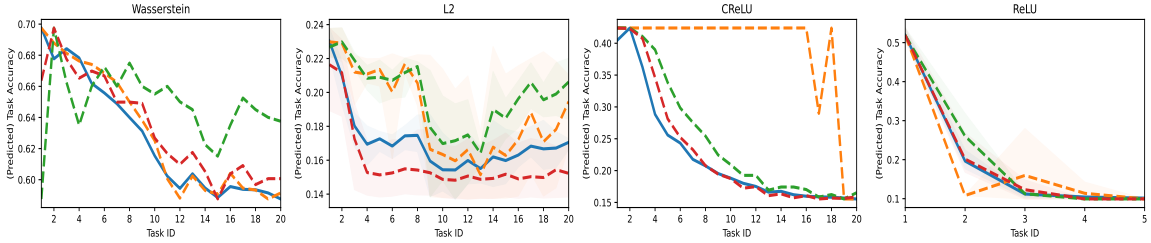


Figure 4: **Predicting per-task accuracy from threshold crossings.** Blue: actual task accuracy. Green: $\alpha^{\star}_{g}$. Orange: $\alpha^{\star}_{\text{vol}}$. Red: combined bound $\tilde{\alpha}^{\star}$.

**Effective step drift** Even with L2 weight decay, the layerwise effective step $\alpha_t$ drifts upward across tasks (Fig. 3). Weight-norm growth shifts gradient statistics and Adam-style preconditioning converts this shift into larger $\alpha_t$. Compared to a ReLU baseline (App. A), L2 moderates but does not halt the drift. As $\alpha_t$ approaches or crosses conservative thresholds $\alpha^{\star}_{g}(t)$ and $\alpha^{\star}_{\text{vol}}(t)$, an increasing share of updates becomes noise- or volatility-dominated even when curvature remains benign. In principle, loss of trainability can be mitigated by reducing the effective step size ($\alpha_t$) or, equivalently, by increasing the batch size ($B$). Appendix A.2 connects this mechanism to further isolated explanations.

## 5. A Combined Predictive Bound and an Adaptive Per-Layer Scheduler

We now combine the two critical effective steps via a volatility-inflated noise proxy, thereby tightening the bound precisely when curvature becomes fragile by inflating the effective noise under volatile sharpness. We define the volatility-inflated per-layer ($\ell$) noise proxy and the corresponding critical effective step as:

$$\tilde{\sigma}_t^{2(\ell)} := \sigma_{t,\text{ps}}^{2(\ell)} + \beta \left\|\widehat{g}_t^{(\ell)}\right\|^2 \text{Vol}_{\lambda}^{(\ell)}(t), \qquad \beta \in [0,1], \tag{2}$$

$$\widetilde{\alpha}_t^{\star(\ell)} := \frac{B \left\|\widehat{g}_t^{(\ell)}\right\|^2}{\tilde{\sigma}_t^{2(\ell)}}. \tag{3}$$

**Simple adaptive per-layer scheduler** From Fig. 4 (see details in App. C.2) we see that this combined bound predicts trainability behaviour most consistently across the considered baselines. The scheduler therefore tracks $\widetilde{\alpha}_t^{\star(\ell)}$ and sets a per-layer safe upper limit $\alpha_t^{\text{safe}(\ell)} := \widetilde{\alpha}_t^{\star(\ell)}(t)$ for the

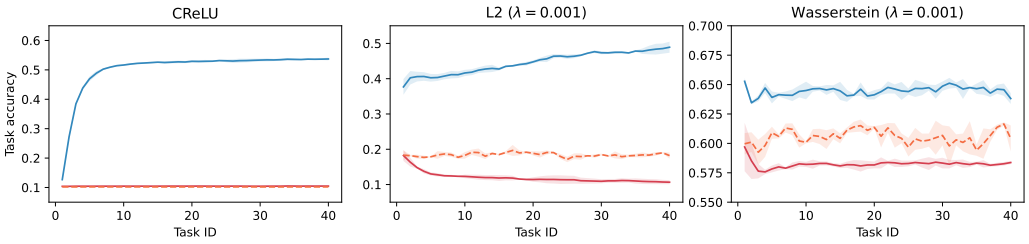


Figure 5: **Controller outcomes across methods.** Red: vanilla. Orange: reset. Blue: scheduled.

effective step size. We modulate per-layer effective steps in Adam every $K$ steps: if a layer's current $\alpha_t^{(\ell)}$ exceeds $\alpha_t^{\text{safe}(\ell)}$, we cool its base LR; if it is conservatively below *early* in training, we warm it. Full pseudocode and hyperparameters appear in App. C (Alg. 1).

## 6. Experiments

Figure 5 compares vanilla, reset-at-task, and our per-layer scheduler across methods. On CReLU, vanilla and reset hover near random accuracy ($\sim 0.1$), while our method stabilizes training around $0.5$ with steady gains. Under L2, vanilla decays to $\sim 0.1$; our scheduler not only restores stability but continues improving, reaching about $0.5$ by $\sim 40$ tasks. With Wasserstein regularizer, longer horizons exhibit mild decay, yet our controller maintains and improves performance to about $0.65$.

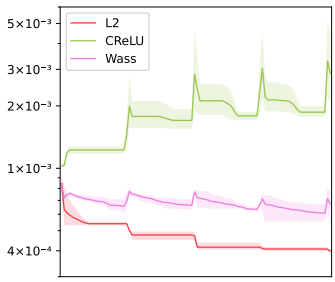


Figure 6: Scheduled learning rate behavior.

Figure 6 shows the learned learning-rate trajectories naturally decay, mirroring standard schedules and yielding improved performance. For L2, the LR cools sharply at the start and settles to a low plateau far below the initial $10^{-3}$; for Wasserstein, a brief warm-up is followed by per-task cool-downs. For CReLU, the LR warms moderately, then plateaus with within-task cool-downs, matching the drop in sharpness volatility and the rise in accuracy.

## 7. Conclusion

We frame loss of trainability in continual learning as an optimization problem shaped jointly by gradient noise and curvature volatility. Single-metric explanations can appear predictive in one setting yet fail in another. Our joint view connects classical noise-scale analyses from stationary training with recent accounts of curvature fragility in CL, motivating a practical controller. The per-layer scheduler reduces noise- or volatility-dominated updates without resets or per-task tuning and composes effectively with L2, Wasserstein, and CReLU. Though tested on a limited set of setups, results suggest that LoT reflects the interaction of noise and curvature, and that lightly-tuning the learning rate is a promising path for robust trainability. Beyond CL, these signals may also guide single-run training regimes (e.g., large-scale LLM pretraining) where only one hyperparameter pass is feasible.

## References


Abbas, Zaheer et al. (2023). "Loss of plasticity in continual deep reinforcement learning". In: *Conference on Lifelong Learning Agents*, pp. 620–636.

Arora, Sanjeev, Zhiyuan Li, and Abhishek Panigrahi (2022). "Understanding Gradient Descent on the Edge of Stability in Deep Learning". In: *Proceedings of the 39th International Conference on Machine Learning*. Vol. 162. Proceedings of Machine Learning Research. PMLR, pp. 948–1024. URL: https://proceedings.mlr.press/v162/arora22a.html.

Ash, Jordan and Ryan P. Adams (2020). "On Warm-Starting Neural Network Training". In: *NeurIPS*.

Berariu, Tudor et al. (2021). *A study on the plasticity of neural networks*. arXiv: 2106.00042 [cs.LG]. URL: https://arxiv.org/abs/2106.00042.

Chen, Liang-Chieh et al. (2017). *Rethinking Atrous Convolution for Semantic Image Segmentation*. arXiv: 1706.05587 [cs.CV]. URL: https://arxiv.org/abs/1706.05587.

Devlin, Jacob et al. (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of NAACL-HLT*. arXiv:1810.04805. URL: https://arxiv.org/abs/1810.04805.

Dohare, Shibhansh et al. (2024). "Loss of plasticity in deep continual learning". In: *Nature* 632.8026, pp. 768–774.

Foret, Pierre et al. (2021a). "Sharpness-Aware Minimization for Efficiently Improving Generalization". In: *ICLR*.

— (2021b). "Sharpness-Aware Minimization for Efficiently Improving Generalization". In: *International Conference on Learning Representations*. arXiv:2010.01412. URL: https://arxiv.org/abs/2010.01412.

Ghorbani, Behrooz, Shankar Krishnan, and Ying Xiao (2019). "An Investigation into Neural Net Optimization via Hessian Eigenvalue Density". In: *Proceedings of the 36th International Conference on Machine Learning*. Vol. 97. PMLR, pp. 2232–2241. URL: https://proceedings.mlr.press/v97/ghorbani19b.html.

Goyal, Priya et al. (2017). *Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour*. arXiv: 1706.02677 [cs.CV]. URL: https://arxiv.org/abs/1706.02677.

Huang, Gao et al. (2017). "Snapshot Ensembles: Train 1, Get M for Free". In: *International Conference on Learning Representations*.

Keskar, Nitish Shirish et al. (2016). "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima". In: *arXiv preprint arXiv:1609.04836*. DOI: 10.48550/arXiv.1609.04836. URL: https://arxiv.org/abs/1609.04836.

Lewandowski, Alex, Dale Schuurmans, and Marlos C. Machado (2024a). "Plastic Learning with Deep Fourier Features". In: *arXiv preprint arXiv:2410.20634*.

Lewandowski, Alex et al. (2024b). "Directions of Curvature as an Explanation for Loss of Plasticity". In: arXiv: 2312.00246 [cs.LG].

Loshchilov, Ilya and Frank Hutter (2017). "SGDR: Stochastic Gradient Descent with Warm Restarts". In: *International Conference on Learning Representations*. arXiv:1608.03983. URL: https://arxiv.org/abs/1608.03983.

Lyle, Clare et al. (2023). "Understanding plasticity in neural networks". In: *arXiv preprint arXiv:2304.02086*.

Lyle, Clare et al. (2024). *Disentangling the Causes of Plasticity Loss in Neural Networks*. arXiv: 2402.18762 [cs.LG]. URL: https://arxiv.org/abs/2402.18762.

Mandt, Stephan, Matthew D. Hoffman, and David M. Blei (2017). "Stochastic Gradient Descent as Approximate Bayesian Inference". In: *Journal of Machine Learning Research* 18.134, pp. 1–35. URL: https://www.jmlr.org/papers/volume18/17-214/17-214.pdf.

McCandlish, Sam et al. (2018). "An Empirical Model of Large-Batch Training". In: *arXiv preprint arXiv:1812.06162*. DOI: 10.48550/arXiv.1812.06162. URL: https://arxiv.org/abs/1812.06162.

Mesbahi, Golnaz et al. (2025). *Position: Lifetime tuning is incompatible with continual reinforcement learning*. arXiv: 2404.02113 [cs.LG]. URL: https://arxiv.org/abs/2404.02113.

Papyan, Vardan (2018). "The Full Spectrum of Deepnet Hessians at Scale: Dynamics with SGD Training and Sample Size". In: *arXiv preprint arXiv:1811.07062*. DOI: 10.48550/arXiv.1811.07062. URL: https://arxiv.org/abs/1811.07062.

Rohani, Seyed Roozbeh Razavi et al. (2025). "Preserving Plasticity in Continual Learning with Adaptive Linearity Injection". In: *arXiv preprint arXiv:2505.09486*.

Sagun, Levent et al. (2017). "Empirical Analysis of the Hessian of Over-Parametrized Neural Networks". In: *arXiv preprint arXiv:1706.04454*. DOI: 10.48550/arXiv.1706.04454. URL: https://arxiv.org/abs/1706.04454.

Santurkar, Shibani et al. (2018). "How Does Batch Normalization Help Optimization?" In: *Advances in Neural Information Processing Systems*. DOI: 10.48550/arXiv.1805.11604. URL: https://arxiv.org/abs/1805.11604.

Shang, Wenling et al. (2016). "Understanding and Improving Convolutional Neural Networks via Concatenated Rectified Linear Units". In: *ICML*.

Smith, Samuel L. et al. (2018). "Don't Decay the Learning Rate, Increase the Batch Size". In: *International Conference on Learning Representations*. arXiv:1711.00489. DOI: 10.48550/arXiv.1711.00489. URL: https://arxiv.org/abs/1711.00489.

Song, Minhak, Kwangjun Ahn, and Chulhee Yun (2025). "Does SGD really happen in tiny subspaces?" In: *International Conference on Learning Representations*. OpenReview ID: v6iLQBoIJw. URL: https://openreview.net/forum?id=v6iLQBoIJw.

Vaswani, Ashish et al. (2017). "Attention Is All You Need". In: *Advances in Neural Information Processing Systems*.

Yoshida, Yuichi and Takeru Miyato (2017). "Spectral Norm Regularization for Improving the Generalizability of Deep Learning". In: *arXiv preprint arXiv:1705.10941*. DOI: 10.48550/arXiv.1705.10941. URL: https://arxiv.org/abs/1705.10941.

Ziyin, Liu (2024). "Symmetry Induces Structure and Constraint of Learning". In: *International Conference on Machine Learning (ICML)*.

## Appendix A. Ablations

### A.1. Varying task-length

#### A.1.1. LEARNING RATE BEHAVIOR

### A.2. Linking existing explanations

In positively homogeneous networks, growth in L2/spectral weight norms increases network sensitivity and spectral norms (and typically loss sharpness), so both the top eigenvalue $\lambda_t$ and the *normalized (Adam-adjusted) sharpness* $\bar{\lambda}_t$ (and its variability) rise. This elevates the volatility $\mathrm{Vol}_{\bar{\lambda}}(t)$

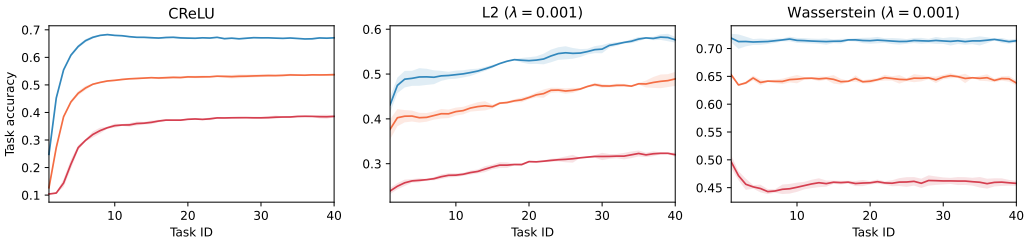


Figure 7: **Controller outcomes across methods, ablating over task-lengths.** Red: 100 epochs/task. Orange: 250 epochs/task. Blue: 500 epochs/task.

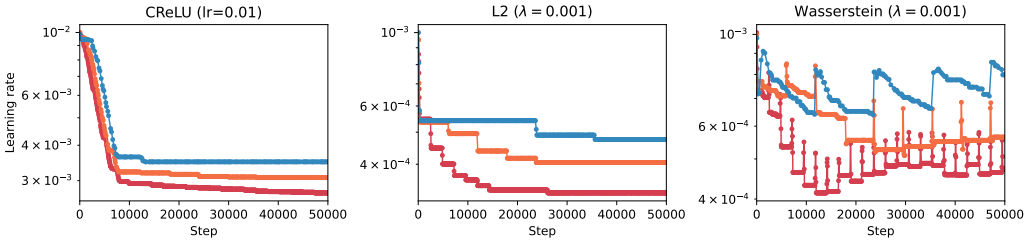


Figure 8: **Controller outcomes across methods, ablating over task-lengths.** Red: 100 epochs/task. Orange: 250 epochs/task. Blue: 500 epochs/task.

defined in Eq. (1), which *shrinks* the curvature–volatility critical step

$$\alpha^{\star}_{\mathrm{vol}}(t) \;=\; \frac{1}{\kappa\,\mathrm{Vol}_{\bar{\lambda}}(t)}\,,$$

making updates brittle (Keskar et al. 2016; Santurkar et al. 2018; Yoshida et al. 2017; Foret et al. 2021b). Apparent "decaying gradient norm" or a falling grad/param ratio are naturally interpreted through the *batch-size–aware* critical step $\alpha^{\star}_{g}(t)$ together with the effective step $\alpha_t$: if $\|\widehat{g}_t\|$ shrinks while the per-sample variance stays high (or $\alpha_t$ drifts upward), a larger share of updates satisfies $\alpha_t > \alpha^{\star}_{g}(t)$, entering the noise-dominated regime without any necessary change in rank. Conversely, high Hessian rank can coexist with LoT when either *noise-dominated* crossings $\alpha_t > \alpha^{\star}_{g}(t)$ or *volatility-dominated* crossings $\alpha_t > \alpha^{\star}_{\mathrm{vol}}(t)$ proliferate; low-rank regimes can remain trainable when both criteria are comfortably satisfied—explaining Fig. 1. We refer the reader to §A.2 for a discussion of when Hessian rank decay aligns with these signals.

**When Hessian rank decay is predictive.** Rank decline aligns with our two-signal view when curvature mass concentrates into a low-dimensional dominant subspace but the gradient does not increasingly align with that subspace, and the per-sample gradient noise does not fall commensurately. Write the eigendecomposition $H_t = \sum_{i=1}^{d} \lambda_{i,t} v_{i,t} v_{i,t}^{\top}$ with $\lambda_{1,t} \geq \lambda_{2,t} \geq \ldots$, let $r_t$ be an effective rank, and $P_{r_t}$ the projector onto $\{v_{1,t}, \ldots, v_{r_t,t}\}$. Define the *projected* batch-size–aware critical step

$$\alpha^{\star}_{g,(r)}(t) \;:=\; \frac{B\,\|P_{r_t}\widehat{g}_t\|^2}{\sigma^{2(r)}_{t,\mathrm{ps}}} \;\approx\; \cos^2\theta_t \cdot \alpha^{\star}_{g}(t) \quad \text{(approx. isotropic noise)},$$

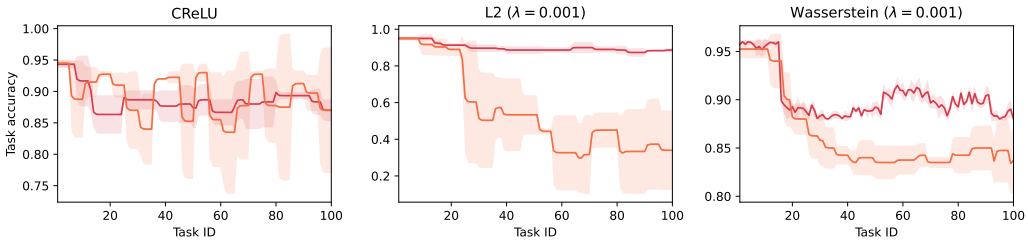


Figure 9: **Hessian rank outcomes across methods.** For each of {CReLU, L2, Wasserstein}: 2 curves (vanilla, per-layer schedule).

where $\cos^2\theta_t = \|P_{r_t}\widehat{g}_t\|^2/\|\widehat{g}_t\|^2$. If $r_t\downarrow$ while $\cos^2\theta_t$ fails to rise (or falls), then $\alpha^\star_{g,(r)}(t)$ drops and a larger fraction of steps become noise-dominated, matching accuracy decay even without explicit forgetting.

In modern networks, rank decay typically coincides with spectral mass concentrating into a few outliers (Sagun et al. 2017; Papyan 2018; Ghorbani et al. 2019), which makes $\lambda_{1,t}$ more sensitive to data/parameter drift; empirically this raises $\mathrm{Vol}_{\tilde{\lambda}}(t)$ (Eq. (1)) and hence *shrinks* $\alpha^\star_{\mathrm{vol}}(t)$ *before* accuracy collapses. Studies of alignment further show gradients tend to align with a small top-eigenspace (Arora et al. 2022; Song et al. 2025); when that space itself shrinks or drifts quickly, alignment cannot compensate, yielding a simultaneous reduction in $\alpha^\star_{g,(r)}(t)$ and $\alpha^\star_{\mathrm{vol}}(t)$. As $\alpha_t$ crosses either threshold persistently, LoT follows.

### A.3. Critical learning rates

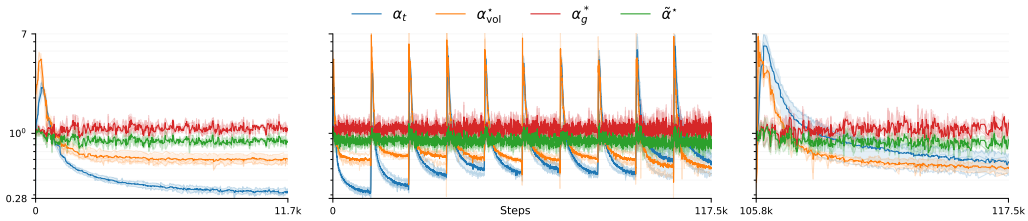


Figure 10: **Effective step vs. sharpness-aware safe bound.** We plot the first layer's effective step $\alpha_{t,l_1}$ with $\alpha^\star_{g,l_1}$, $\tilde{\alpha}^\star_{l_1}$, and $\alpha^\star_{\mathrm{vol},l_1}$ for L2 ($\lambda = 10^{-3}$). Task 1 (left), Tasks $1-10$ (middle), and Task 10 (right).

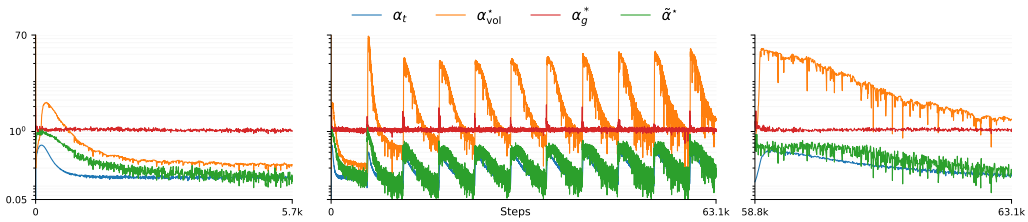


Figure 11: **Effective step vs. sharpness-aware safe bound.** We plot the first layer's effective step $\alpha_{t,l_1}$ with $\alpha^\star_{g,l_1}$, $\tilde{\alpha}^\star_{l_1}$, and $\alpha^\star_{\mathrm{vol},l_1}$ for CReLU. Task 1 (left), Tasks $1-10$ (middle), and Task 10 (right).

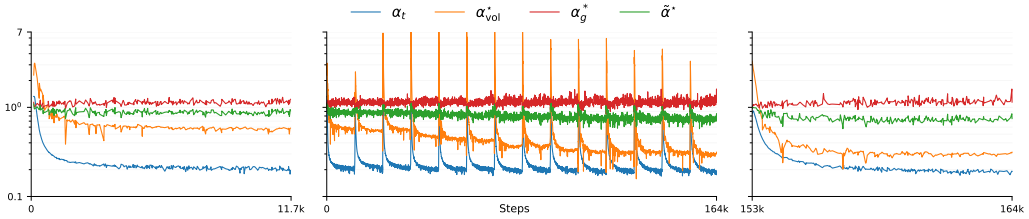


Figure 12: **Effective step vs. sharpness-aware safe bound.** We plot the first layer's effective step $\alpha_{t,l_1}$ with $\alpha^{\star}_{g,l_1}$, $\tilde{\alpha}^{\star}_{l_1}$, and $\alpha^{\star}_{\mathrm{vol},l_1}$ for Wasserstein regularizer ($\lambda = 10^{-3}$). Task 1 (left), Tasks $1-10$ (middle), and Task 10 (right).

## Appendix B. Theoretical details

### B.1. Normalized sharpness

Consider the local quadratic model with a preconditioned step $w_{t+1} = w_t - \eta_t D_t \widehat{g}_t$, where $D_t = \mathrm{diag}\big(1/(\sqrt{\widehat{v}_t} + \epsilon)\big) \succ 0$. The second-order term governing stability along this update is

$$\Delta w_t^{\top} H_t \Delta w_t \;=\; (\eta_t \widehat{g}_t)^{\top} D_t^{1/2} H_t D_t^{1/2} (\eta_t \widehat{g}_t).$$

Thus the relevant curvature scale is $\lambda_{\max}\big(D_t^{1/2} H_t D_t^{1/2}\big)$. By submultiplicativity of the spectral norm,

$$\lambda_{\max}\big(D_t^{1/2} H_t D_t^{1/2}\big) \;\le\; \|D_t\|_2 \, \lambda_{\max}(H_t) \;=\; d_{\max,t}\, \lambda_t,$$

where $d_{\max,t}$ is the largest diagonal entry of $D_t$. Replacing $D_t$ by a scalar surrogate $d_t I$ yields a tractable approximation $\lambda_{\max}\big(D_t^{1/2} H_t D_t^{1/2}\big) \approx d_t\, \lambda_t$. We take $d_t := \alpha_{\mathrm{agg},t}/\eta_t$ with

$$\alpha_{\mathrm{agg},t} \;=\; \frac{\eta_t}{\mathrm{RMS}(\sqrt{\widehat{v}_t}) + \epsilon},$$

which lies between $\eta_t d_{\min,t}$ and $\eta_t d_{\max,t}$. Hence

$$\eta_t^2\, \lambda_{\max}\big(D_t^{1/2} H_t D_t^{1/2}\big) \;\lesssim\; \big(\alpha_{\mathrm{agg},t}\, \lambda_t\big) \cdot \frac{d_{\max,t}}{d_t},$$

so $\bar{\lambda}_t := \alpha_{\mathrm{agg},t}\lambda_t$ is a conservative scalar proxy up to a factor depending on the spread of $D_t$ (bounded by its condition number). In practice this surrogate tracks the preconditioned curvature closely enough for reliable volatility detection and decision thresholds in our scheduler. A per-layer version (replacing the global RMS with layerwise RMS) is analogous and can tighten the bound further.

### B.2. Volatility

**Setup.** With Adam, an update takes the preconditioned form

$$w_{t+1} = w_t - \alpha_t\, \widehat{g}_t^{(\mathrm{eff})}, \qquad \widehat{g}_t^{(\mathrm{eff})} := D_t\, \widehat{g}_t, \quad D_t = \mathrm{diag}\Big(\tfrac{1}{\sqrt{\widehat{v}_t}+\epsilon}\Big).$$

Let $\lambda_t = \lambda_{\max}(H_t)$. We work with the normalized (Adam-adjusted) sharpness $\bar{\lambda}_t := \alpha_{\mathrm{agg},t}\lambda_t$ where $\alpha_{\mathrm{agg},t} = \eta_t/(\mathrm{RMS}(\sqrt{\widehat{v}_t}) + \epsilon)$. Over a short window $W$, treat $\{\bar{\lambda}_s\}$ as a scalar stochastic process with temporal mean and variance

$$\mu_t := \mathrm{EMA}_W[\bar{\lambda}], \qquad \sigma_t^2 := \mathrm{Var}_W[\bar{\lambda}]\,.$$

We write the one-step quadratic model along the (preconditioned) update direction:

$$\Delta\ell_s \;\lesssim\; -\alpha_t\,\|\widehat{g}_s^{(\mathrm{eff})}\|^2 \;+\; \tfrac{1}{2}\alpha_t^2\,\bar{\lambda}_s\,\|\widehat{g}_s^{(\mathrm{eff})}\|^2. \tag{A.1}$$

The first term is the *deterministic* descent, the second is the *curvature* penalty, whose *temporal* fluctuation we will control.

**A small-step contraction criterion via log-linearization.** Consider the scalar dynamics of the top mode (after preconditioning): $z_{s+1} \approx (1-\alpha_t\bar{\lambda}_s)z_s$. For $0 \le x < 1$, the inequality $\log(1-x) \le -x - \frac{1}{2}x^2$ yields

$$\mathbb{E}\big[\log|1-\alpha_t\bar{\lambda}_s|\big] \;\le\; -\alpha_t\,\mu_t \;-\; \tfrac{1}{2}\alpha_t^2\,\mathbb{E}[\bar{\lambda}_s^2] \;=\; -\alpha_t\,\mu_t \;-\; \tfrac{1}{2}\alpha_t^2(\mu_t^2+\sigma_t^2). \tag{A.2}$$

A sufficient condition for expected contraction is that the volatility (quadratic) term remain a chosen fraction $c \in (0,1)$ of the linear term:

$$\tfrac{1}{2}\alpha_t^2(\mu_t^2+\sigma_t^2) \;\le\; c\,\alpha_t\,\mu_t \quad\Longrightarrow\quad \alpha_t \;\le\; \frac{2c\,\mu_t}{\mu_t^2+\sigma_t^2}. \tag{A.3}$$

*Volatility-dominated regime.* When $\sigma_t^2 \gg \mu_t^2$, (A.3) reduces to

$$\alpha_t \;\lesssim\; \frac{2c\,\mu_t}{\sigma_t^2} \;=\; \frac{2c}{\mathrm{Vol}_{\bar{\lambda}}(t)}\,, \qquad \mathrm{Vol}_{\bar{\lambda}}(t) := \frac{\sigma_t^2}{\mu_t+\varepsilon}. \tag{A.4}$$

This motivates the practical curvature-driven critical step

$$\alpha_{\mathrm{vol}}^\star(t) \;:=\; \frac{1}{\kappa\,\mathrm{Vol}_{\bar{\lambda}}(t)}$$

with $\kappa$ absorbing the constants (the choice of $c$, the small $\varepsilon$, and mismatch between the scalar surrogate $\bar{\lambda}$ and $\lambda_{\max}(D_t^{1/2}H_tD_t^{1/2})$). Intuitively, (A.3) comes from requiring the *temporal* curvature fluctuation penalty to stay subdominant to the deterministic descent; in the high-volatility regime the controlling quantity is $\sigma_t^2/\mu_t$, i.e., $\mathrm{Vol}_{\bar{\lambda}}$.

From (A.1), a step is unstable if $\alpha_t\bar{\lambda}_s \ge 2$. Cantelli's inequality gives $\mathbb{P}(\bar{\lambda}_s - \mu_t \ge a) \le \frac{\sigma_t^2}{\sigma_t^2+a^2}$. Setting $a = 2/\alpha_t - \mu_t$ and requiring this probability $\le \delta$ yields a sufficient condition

$$\alpha_t \;\le\; \frac{2}{\mu_t + \sigma_t\sqrt{(1-\delta)/\delta}}\,. \tag{A.5}$$

Eq. (A.5) is tighter when $\mu_t$ is not dwarfed by $\sigma_t$; Eq. (A.3) (hence $\alpha_{\mathrm{vol}}^\star \propto 1/\mathrm{Vol}_{\bar{\lambda}}$) is simpler and conservative when volatility dominates. In practice, we use $\alpha_{\mathrm{vol}}^\star$ for fast control and optionally cap $\alpha_t$ by (A.5) for extra safety.

### B.3. Link to symmetry-induced curvature collapse

Ziyin (2024) demonstrate that common mirror symmetries in deep networks impose algebraic constraints on the Hessian. Under weight decay or large gradient noise, these constraints bias SGD toward symmetric, low-rank or sparse solutions, where so-called "collapse" phenomena emerge. In our framework, such phases correspond to decreased $\mu_t$ and/or increased $\sigma_t^2$, leading to a contraction of $\alpha^\star_{\text{vol}}$ and signaling the onset of untrainability prior to an observable accuracy drop.

Let $H$ denote the Hessian of the loss at parameters $\theta_t$, and let $P$ be the orthogonal projector onto the subspace associated with a given symmetry (e.g., $O$-mirror symmetry). Write $d_t$ for the normalized update direction. By the Courant–Fischer theorem,

$$\lambda_{\max}(PHP) \;=\; \max_{\substack{\|v\|=1 \\ v\in\text{range}(P)}} v^\top H v \;\le\; \lambda_{\max}(H),$$

and analogously,

$$\lambda_{\min}(PHP) \;\ge\; \lambda_{\min}(H).$$

For the Ritz value $\rho_t = d_t^\top H d_t$, we thus obtain

$$\lambda_{\min}(PHP) \;\le\; \rho_t \;\le\; \lambda_{\max}(PHP) \;\le\; \lambda_{\max}(H).$$

Define the overlap factor $\alpha_t = \|Pd_t\|/\|d_t\|$. Expanding in terms of $Pd_t$,

$$\rho_t = (Pd_t)^\top H(Pd_t) \;\pm\; \text{cross terms}.$$

If $d_t$ lies predominantly in the symmetry subspace (so cross terms are negligible),

$$\rho_t \;\approx\; \alpha_t^2\,\lambda_{\max}(PHP) \;\le\; \alpha_t^2\lambda_{\max}(H).$$

As symmetry constraints "lock in" during training, $\alpha_t \to 1$, and the Ritz value $\rho_t$ becomes tightly bracketed by the projected and global sharpness.

Let $\sigma_\rho^2 = \text{Var}[\rho_t]$ and $\sigma_\lambda^2 = \text{Var}[\lambda_{\max}(H)]$. If $\alpha_t$ varies slowly and remains close to one, then

$$\sigma_\rho^2 \;\approx\; \sigma_\lambda^2.$$

Hence, fluctuations in $\lambda_{\max}(H)$ provide a practical proxy for the volatility of the stability-critical curvature.

Finally, Ziyin (2024) derive a Lyapunov-type stability criterion in symmetry-restricted directions, expressed in terms of the mean and variance of curvature:

$$\text{Collapse if:} \quad \eta\,\bar{\xi} \;\gtrsim\; \frac{-2\,\mathbb{E}[\xi+\gamma]}{\mathbb{E}[(\xi+\gamma)^2]}.$$

Substituting $\xi \mapsto \rho_t$ and applying the above bounds yields a conservative but practical collapse rule:

$$\eta_t \cdot \lambda_{\max}(H) \;\lesssim\; \text{margin}(\text{mean}, \text{var}).$$

## Appendix C. Implementation details

### C.1. Metrics

**Effective step (global & per-layer).** We report the *effective step size* $\alpha_t$ actually applied by the optimizer. For Adam/ClampedAdam we average, over parameters, the elementwise multiplier

$$\frac{\eta_t}{(1-\beta_1^t)\,(\sqrt{\hat{v}_t}+\varepsilon)}\,.$$

We also compute a *per-layer* effective step $\alpha_{t,\ell}$ by aggregating within each top-level module (e.g., `fc1`, `fc2`). These $\alpha_t$/$\alpha_{t,\ell}$ are compared against the safety thresholds below.

**Sharpness and normalized sharpness.** We estimate the top Hessian eigenvalue $\lambda_t$ via power iteration with Hessian–vector products ($k$=1, $\sim$100 steps). For Adam we form a *normalized sharpness* scalar

$$\bar{\lambda}_t \;=\; \alpha_{\mathrm{agg},t}\,\lambda_t, \qquad \alpha_{\mathrm{agg},t} \;=\; \frac{\eta_t}{\mathrm{RMS}(\sqrt{\hat{v}_t})+\varepsilon},$$

which tracks the preconditioned curvature along the update; for SGD we set $\bar{\lambda}_t = \lambda_t$. For each layer $\ell$ we maintain windowed statistics of $\bar{\lambda}_{t,\ell}$ using an EMA plus a finite queue (length 30): the running mean $\mu_{t,\ell}$ and the instantaneous squared deviation $(\bar{\lambda}_{t,\ell} - \mu_{t,\ell})^2$, which we use for the volatility threshold in Eq. 1.

**Batch-size–aware gradient noise.** At log intervals we compute the exact within-minibatch per-sample gradient variance

$$\hat{\sigma}^2_{\mathrm{mb}}(t) \;=\; \frac{1}{B}\sum_{i=1}^{B}\big\|g_i-\bar{g}\big\|^2,$$

by looping over per-sample losses.

### C.2. Experiments

In Fig. 4, we compute per-layer predictions as follows: for each layer $\ell$, we evaluate the critical step $\tilde{\alpha}_t^{\star(\ell)}(t)$ and calculate a predicted loss-of-trainability as:

$$\widehat{\rho} \;=\; \frac{1}{T}\sum_{t=1}^{T}\mathbf{1}\Big\{\exists\,\ell:\;\alpha_t^{(\ell)} > \widetilde{\alpha}_t^{\star(\ell)}(t)\Big\}\,.$$

Since $\widehat{\rho}$ only captures the relative training behavior, we scale it to fit the range of the actual trainability curve.

### C.3. Algorithm

## Appendix D. Full experimental setup

Experiments use a random-label MNIST task sequence of 40 tasks with 250 epochs per task; scheduler decisions are taken every $K$ steps with small warming/cooling updates, and power-iteration budgets are kept short to bound overhead . With longer tasks, our controller maintains trainability.

**Algorithm 1: Per-layer sharpness-aware LR scheduler (ours), invoked every $K$ steps**

**Input:** layer params $w^{(\ell)}$, base LR $\eta^{(\ell)}$, EMA state $\mathcal{E}^{(\ell)}$, decision-window $W$, safety factor $\gamma$, cool rate $c = 1 - \epsilon$, warm rate $u = 1 + \epsilon$; $\epsilon \approx 10^{-3}$, interval $K$

**for** $t \in \{K, 2K, 3K, \dots\}$ *until* $t \geq T$ **do**

  Estimate per-layer sharpness $S_t^{(\ell)}$ (top Hessian eigenvalue) and minibatch grad variance $\sigma_{t,\mathrm{mb}}^{2(\ell)}$

  Update EMA state $\mathcal{E}^{(\ell)}$ with $(S_t^{(\ell)}, \alpha_t^{(\ell)})$

  Compute $\tilde{\sigma}_t^{2(\ell)}$, $\tilde{\alpha}_t^{\star(\ell)}$, and safe bound $\tilde{\alpha}_t^{\mathrm{safe}(\ell)}$ `// via (??)--(??)`

  Form effective step $\alpha_t^{(\ell)}$ from optimizer state `// e.g. Adam bias-corrected factor`

  Form current effective step $\alpha_t^{(\ell)}$ from optimizer state `// e.g. Adam bias-corrected factor`

  **if** $\alpha_t^{(\ell)} > \gamma \tilde{\alpha}_t^{\mathrm{safe}(\ell)}$ ***and*** $\alpha_t^{(\ell)} > 0.12$ **then**

    $\eta^{(\ell)} \leftarrow c \cdot \eta^{(\ell)}$ `// cool overly aggressive layer`

  **else**

    **if** $t < 0.3T$ ***and*** $\alpha_t^{(\ell)} \ll \gamma \tilde{\alpha}_t^{\mathrm{safe}(\ell)}$ **then**

      $\eta^{(\ell)} \leftarrow u \cdot \eta^{(\ell)}$ `// warm timid layer early on`

    **end**

  **end**

**end**

### D.1. Dataset

All figures (unless stated otherwise) use a random-label MNIST stream. Starting from the MNIST training split, we randomly subsample $21{,}000$ images once and normalize inputs using the dataset mean and standard deviation returned by our loader. At the beginning of each task we resample targets with full dataset randomization, so input statistics remain MNIST-like while class semantics are destroyed. For the controller experiments we run a stream of $40$ tasks with $250$ epochs per task; for the ablations we keep everything else fixed and vary the per-task budget across $100$, $250$, and $500$ epochs. All runs are averaged over $3$ seeds.

### D.2. Models

Unless specified, the backbone is a two-layer MLP with a hidden width of $256$ and a $10$-way classifier.

### D.3. Hyperparameters

Table 1: Hyperparameters used across experiments. Unless noted, the controller experiments use the "Main" settings; ablations vary only the indicated field(s).

| **Hyperparameter** | **Value** |
|---|---|
| Label randomization | Fraction $\texttt{ns} = 1.0$ (full dataset re-label each task) |
| Batch size | $256$ |
| Model | 2-layer MLP (hidden width $256$) |
| Initialization | Kaiming (default) |
| Optimizer | Adam ($\beta_1$=0.9, $\beta_2$=0.999) |
| Base learning rate | $\eta = 10^{-3}$ (all of Fig. 1 unless specified) |
| Weight decay (L2) | Only when `reg=l2`; when used, $\lambda_{\mathrm{L2}} = 10^{-3}$ |
| Wasserstein reg. | When used, $\lambda_{\mathrm{Wass}} = 10^{-3}$ |
| Per-layer controller | safety $0.8$, cool $0.99$, warm $1.01$, window $30$ |
| Logging | Log interval $40$ updates |