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
University of British Columbia

Alex Lewandowski LEWANDOWSKI@UALBERTA.CA
University of Alberta

Mark Schmidt SCHMIDTM@CS.UBC.CA
University of British Columbia, Canada CIFAR AI Chair (Amii)

Abstract

Loss of trainability refers to a phenomenon in continual learning where parameter updates no longer make progress on the optimization objective, so accuracy stalls or degrades as the learning problem changes over time. In this paper, we analyze loss of trainability through an optimization lens and find that the phenomenon is not reliably predicted by existing individual indicators such as Hessian rank, sharpness level, weight or gradient norms, gradient-to-parameter ratios, and unit-sign entropy. Motivated by our analysis, we introduce two complementary indicators: a batch-size-aware gradient-noise bound and a curvature volatility-controlled bound. We then combine these two indicators into a per-layer adaptive noise threshold on the effective step-size that anticipates trainability behavior. Using this insight, we propose a step-size scheduler that keeps each layer's effective parameter update below this bound, thereby avoiding loss of trainability. We demonstrate that our scheduler can improve the accuracy maintained by previously proposed approaches, such as concatenated ReLU (CReLU), Wasserstein regularizer, and L2 weight decay. Surprisingly, our scheduler produces adaptive step-size trajectories that, without tuning, mirror the manually engineered step-size decay schedules.

1. Introduction

In continual learning, a model must sustain performance over a non-stationary data stream, meaning that optimization methods must remain effective as the model's parameters evolve. A central failure mode in continual learning is *loss of trainability (LoT)*, in which parameter updates become increasingly ineffective, so training slows or stalls even when capacity and supervision are adequate (Lyle et al. 2023; Dohare et al. 2024). This phenomenon is a major impediment to learning from new data, and is distinct from catastrophic forgetting, which is concerned with the retention of what was previously learned. Empirically, LoT has been observed across architectures, datasets, and non-stationarities (e.g., random labels, permuted inputs, or incremental classes), suggesting it is a general property of gradient dynamics in non-stationary settings rather than an isolated peculiarity.

Mechanistically, several indicators have been proposed as explanations for LoT: increasing weight magnitudes (Dohare et al. 2024), decaying Hessian rank (Lewandowski et al. 2024b), reductions in gradient noise (Berariu et al. 2021), dead units due to saturation (Dohare et al. 2024; Abbas et al. 2023), unit linearization (Lyle et al. 2024), or unit sign entropy collapse (Lewandowski et al. 2024a). Yet in practice, these indicators are not reliable predictors of the phenomenon across optimizers and hyperparameters. A complete explanation that accounts for hyperparameters is particularly important because continual learning typically precludes *lifetime tuning*: we cannot reset or retune step-sizes and regularizers for each task sequence (Ash et al. 2020; Mesbahi et al. 2025). A

robust indicator must therefore account for variability across hyperparameters, and it must explain the cases for which a single indicator fails to reliably predict LoT.

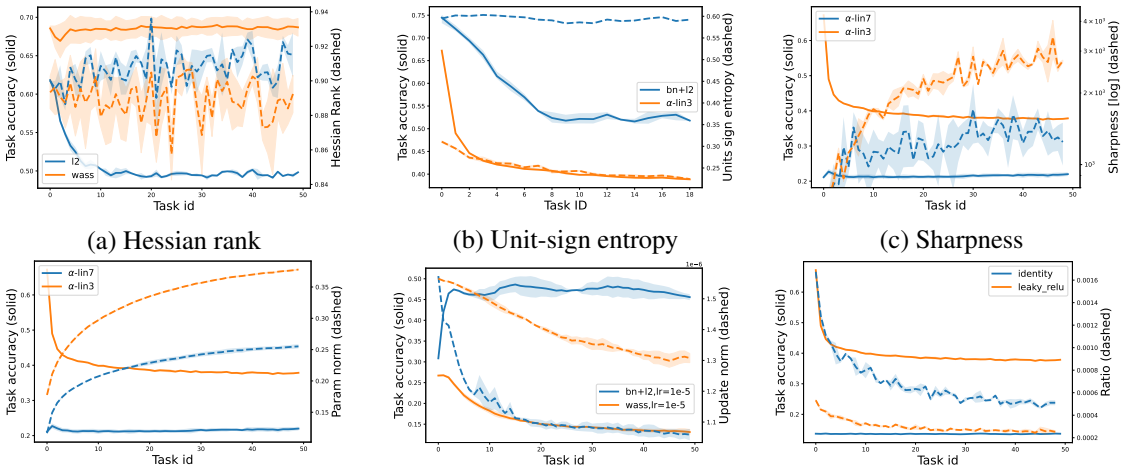
In this work, we adopt a two-signal view of trainability. We propose a *batch-size–aware* gradient-noise signal and a *sharpness volatility* signal based on the normalized (Adam-adjusted) sharpness. Together, these two signals yield a per-layer adaptive noise threshold on the effective step-size. We use this insight to implement a simple per-layer scheduler for Adam that keeps each layer's effective step below this bound. This ensures updates are neither *gradient-noise dominated* (where gradient randomness overwhelms the signal) nor *curvature-noise dominated* (where sharpness volatility causes steps to alternate between descent and instability).

To evaluate the curvature volatility diagnosis and our scheduler as a mitigator, we consider a continual learning setting where *lifetime tuning* of hyperparameters is infeasible. We then consider continual learning algorithms that surface complementary failure routes: CReLU activations (Shang et al. 2016), which ensure that every unit has a path for gradient flow, but do not prevent curvature collapse; Wasserstein regularizer (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. We demonstrate that our scheduler improves the accuracy maintained by these approaches and, without tuning, yields adaptive step-size trajectories that mirror manually engineered decay schedules.

2. Related Work

Diagnosing loss of trainability A growing body of work characterizes LoT as a byproduct of stochastic gradient descent (SGD) that emerge when training on non-stationary data. Several studies emphasize geometric collapse: Lewandowski et al. (2024b) argue that reduced directions of curvature explain the onset of LoT, connecting Hessian rank degeneracy to a reduced capacity for learning. Ziyin (2024) analyze symmetries in neural network objectives and show that SGD with L2 weight decay has a propensity to saddle points that results in sparse and low-rank solutions. Other accounts highlight instability caused by growing parameter norms (Dohare et al. 2024), and more specifically growing sharpness (Foret et al. 2021a). Lyle et al. (2024) found that layer normalization with growing parameter norms results in decaying effective step-sizes that reduce trainability. Beyond geometry and noise, unit activations have been implicated: saturation and linearization reduce the diversity of unit responses (Abbas et al. 2023; Lyle et al. 2024), which can be generalized to a collapse of unit-sign entropy (Lewandowski et al. 2024a). More broadly, multiple different potential mechanisms can contribute to LoT, depending on the specific configuration of the learning algorithm. As we will show, existing individual mechanisms provide an incomplete explanation for LoT.

Step-size as noise injection and plasticity. The step-size of an optimization algorithm plays a central role in plasticity: it acts as controllable noise (Mandt et al. 2017; Smith et al. 2018), and regulates the balance between escaping sharp minima and settling into stable solutions. While classical noise-scale theory prescribes step-sizes based on gradient signal-to-noise ratios (Schaul et al. 2013), we find this insufficient in non-stationary settings where curvature volatility acts as a distinct, dominant failure mode. Our results refine this view in continual learning: larger step-sizes are not uniformly beneficial. When gradients become noise-dominated or curvature becomes fragile, *decaying* the step-size restores trainability. For instance, under L2 and Wasserstein regularizer, we often reduce the base step-size before briefly re-warming, whereas CReLU tolerates mild warm-ups only



(a) Hessian rank (b) Unit-sign entropy (c) Sharpness

(d) Weight norm (e) Grad norm (f) Grad/Param ratio

Figure 1: **Single indicators are incomplete explanations for loss of trainability.** In each panel, we compare two configurations (blue vs. orange). If an indicator were reliable, a collapse in the metric (dashed) should consistently predict a collapse in accuracy (solid). However, we observe contradictions; for example, in **(a)**, the L2 model maintains high Hessian rank yet suffers catastrophic accuracy loss, whereas the Wasserstein model maintains accuracy despite similar rank behavior.

after volatility drops (Sec. 5). This aligns 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).

3. No Complete Single Explanation

Here we present experimental findings in which no existing individual indicator provides a complete explanation of LoT across architectures or hyperparameters (see Appendix D for details). We see that in Fig. 1: *(a) Hessian rank:* High rank is often associated with trainability, yet under L2 weight decay, the model sustains a high rank but suffers a total collapse in accuracy (≈ 0.1). Conversely, the Wasserstein regularizer maintains high task accuracy (≈ 0.65) despite exhibiting a similarly stable rank profile. *(b) Unit-sign entropy:* While low entropy (feature collapse) can signal LoT, BatchNorm with weight decay maintains stable entropy yet loses trainability. In contrast, LeakyReLU (ρ=0.3) shows a drop in entropy but preserves better trainability. *(c) Sharpness:* Both LeakyReLU (ρ=0.7) and LeakyReLU (ρ=0.3) exhibit rising sharpness, yet only the former suffers catastrophic trainability loss. *(d) Weight Norm:* Comparing the same pair of networks, they show similar growth in weight norm, but exhibit differing trainability behavior. *(e) Gradient Norm:* Gradient decay appears in both the failing configuration (BatchNorm with weight decay) and the successful configuration (Wasserstein regularizer), yet the latter, with faster decay, remains trainable. *(f) Grad/Param Ratio:* This ratio falls for both deep linear and LeakyReLU networks, but only the latter loses trainability.

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

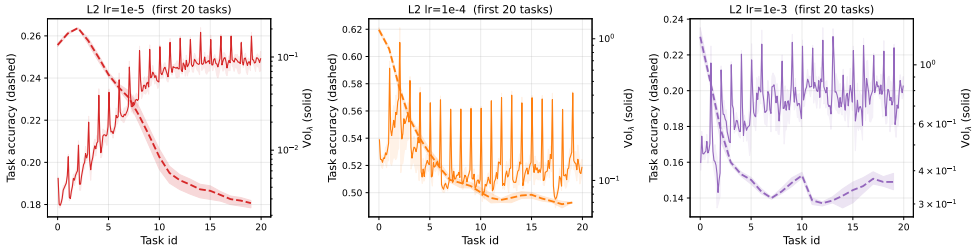


Figure 2: **Sharpness volatility as an indicator.** Under L2 regularization, the rise in curvature volatility (solid) consistently precedes and mirrors the collapse in task accuracy (dashed).

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-size from gradient noise We analyze the conditions for expected descent along the update direction (see derivation in Appendix B.2). While Adam determines the update direction via preconditioning, the step-size magnitude determines the regret guarantee (Kingma et al. 2015). In general, the optimal choice of step-size depends on both the curvature of the optimization landscape and the gradient noise (Schaul et al. 2013). Assume the loss J is locally L-smooth along the update direction. Let the minibatch gradient be $\widehat{g}_t = g_t + \xi_t$, where $g_t = \mathbb{E}[\widehat{g}_t]$ is the true gradient, $\sigma^2_{t,\mathrm{ps}}$ a per-sample variance proxy, and B the batch size. A quadratic expansion of J yields the sufficient condition for expected descent:

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

This recovers the optimal step-size form derived by Schaul et al. (2013), adapted here to the batch-size–aware setting. Defining the noise-critical step-size $\alpha^\star_g(t) := B\|g_t\|^2/\sigma^2_{t,\mathrm{ps}}$, we obtain the compact criterion $\alpha_t \;\le\; 2\,\alpha^\star_g(t)/L$. Empirically, $\alpha^\star_g$ hovers near 1 early (see $\alpha^\star_g$ in Fig. 3) and becomes more noisy late in tasks. As the effective step-size α_t drifts upward, the fraction of steps where $\alpha_t > \alpha^\star_g(t)$ grows. This noise-dominated phase is evident in L2 where accuracy deteriorates although the curvature signal remains consistent. (center in Fig. 3).

Critical step-size from sharpness volatility We typically lack access to the true Hessian at every step, and must instead rely on stochastic estimates. To account for Adam's preconditioning, we track the *normalized* sharpness $\bar{\lambda}_t$ (the curvature projected along the preconditioned update direction, see Appendix B.3). We estimate the raw top eigenvalue λ_t efficiently at every step via power iteration using a single Hessian-vector product, incurring negligible overhead (see Appendix C.1). Since curvature fluctuates during training, we estimate its temporal signal-to-noise ratio. We use an exponential moving average mean μ_t and variance σ^2_t of $\bar{\lambda}_t$ to define the dimensionless *curvature volatility*:

$$\mu_t := \frac{(1-\nu)\mu_{t-1} + \nu\bar{\lambda}_t}{1-\nu^t}, \quad \sigma^2_t := (\bar{\lambda}_t - \mu_t)^2, \quad \mathrm{Vol}_{\bar{\lambda}}(t) := \frac{\sigma^2_t}{\mu_t + \varepsilon}. \tag{1}$$

High volatility indicates that the local curvature is noisy or changing rapidly, destabilizing large steps. In Appendix B.4, we use a quadratic expansion to show that ensuring stability with high probability requires the step-size to scale inversely with this volatility. Capturing the constant scaling factors in κ, we obtain the bound: $\alpha_t \;\le\; \alpha^\star_{\mathrm{vol}}(t) \;:=\; (\kappa\,\mathrm{Vol}_{\bar{\lambda}}(t))^{-1}$.

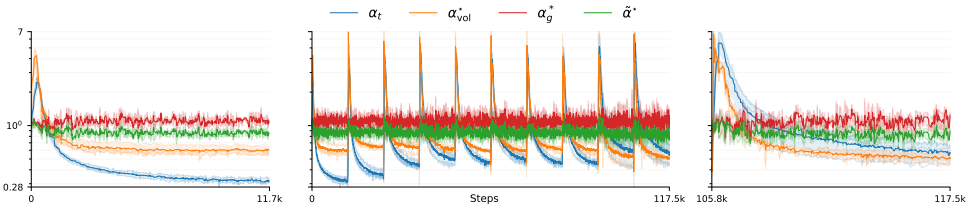


Figure 3: **First layer step-size dynamics when training with L2 weight decay ($\lambda = 10^{-3}$).** The effective step-size α_t (blue) exhibits upward drift across tasks due to weight-norm growth, which Adam's preconditioning converts into an increased step-size.

This bound tightens as sharpness variance exceeds average sharpness, which we refer to as fragile (or unstable) sharpness.

Interpretation. The bound shows that larger μ_t and smaller σ_t^2 make curvature along the preconditioned direction more *predictable*, permitting larger safe steps. In continual learning, volatility can arise when the optimizer moves through non-stationary landscapes with shifting geometry or slips into sharp regions where curvature estimates fail. This pattern has been observed empirically when weight-norm growth or feature collapse drives the optimizer into regions of the parameter-space where local geometry varies drastically across mini-batches. Consequently, when $\alpha^{\star}_{\mathrm{vol}}(t)$ *decays* ($\sigma_t^2 \gg \mu_t$), the curvature signal weakens; persistent exceedance $\alpha_t > \alpha^{\star}_{\mathrm{vol}}(t)$ indicates curvature-noise dominated updates and precedes LoT (Fig. 2).

Effective step-size drift Even with L2 weight decay, the layerwise effective step-size α_t drifts upward across tasks (Fig. 3). As α_t approaches or crosses thresholds $\alpha^{\star}_g(t)$ and $\alpha^{\star}_{\mathrm{vol}}(t)$, an increasing share of updates becomes gradient noise- or curvature-noise dominated. In principle, LoT can be mitigated by reducing α_t or, equivalently, by increasing the batch size (B).

5. Adaptive Noise Threshold and Per-Layer Scheduler

While classical noise-scale theory provides an upper bound on step-size based on gradient variance (Schaul et al. 2013), we find it insufficient for non-stationary continual learning. As shown in our ablation (Fig. 7), trainability is only consistently restored when satisfying a joint constraint. We therefore combine the two critical effective step-sizes using a curvature volatility-aware noise proxy ($\tilde{\sigma}^2$). This tightens the bound precisely when curvature becomes fragile by inflating the effective noise. We define this per-layer (ℓ) noise proxy and the corresponding critical effective step-size as:

$$\tilde{\sigma}_t^{2(\ell)} := \sigma_{t,\mathrm{ps}}^{2(\ell)} + \beta \left\|\widehat{g}_t^{(\ell)}\right\|^2 \mathrm{Vol}_{\tilde{\lambda}}^{(\ell)}(t), \quad \beta \in [0,1]^{1}; \qquad \widetilde{\alpha}_t^{\star(\ell)} := \frac{B \left\|\widehat{g}_t^{(\ell)}\right\|^2}{\tilde{\sigma}_t^{2(\ell)}}. \tag{2}$$

Simple adaptive per-layer scheduler From Fig. 4 (top), we see that this combined bound predicts trainability behavior most consistently, where the predicted accuracy serves as a proxy for the frequency of step-size violations (see details in Appendix C.2). The scheduler tracks $\widetilde{\alpha}_t^{\star(\ell)}$ and defines a per-layer safe bound $\alpha_t^{\mathrm{safe}(\ell)} := (1-\epsilon)\,\widetilde{\alpha}_t^{\star(\ell)}$. Every K steps, Adam's per-layer effective step-size is adjusted: if $\alpha_t^{(\ell)}$ exceeds $\alpha_t^{\mathrm{safe}(\ell)}$ the layer's base LR is cooled, and if it lies conservatively below this bound early in training, it is warmed. Refer to Appendix C (Alg. 1) for implementation details and Appendix A.3 for an empirical comparison of the per-layer vs. full-network controller performance.

1. where β can be tuned to control the inflation, which we set at 1.

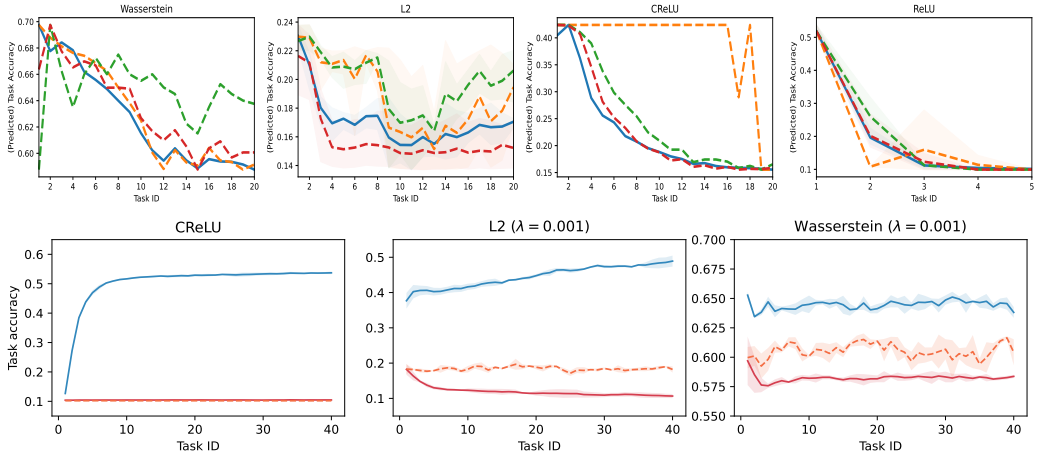


Figure 4: **Diagnosis and Mitigation. Prediction (top):** The combined bound $\tilde{\alpha}^*$ (red) most accurately anticipates the onset of accuracy drops (blue) compared to individual metrics (α^*_{Vol}, red; α^*_g, green). **Performance (bottom):** The per-layer scheduler (blue) consistently restores trainability across all baselines, significantly outperforming vanilla (red) training and task resets (orange).

6. Experiments

To evaluate the curvature volatility diagnosis and our scheduler as a mitigator, we investigate LoT on a sequence of 40 non-stationary tasks using a random-label MNIST. Each task involves memorizing randomized target labels to isolate optimization dynamics from semantic transfer. We use a two-layer MLP (width 256) trained with Adam, where we use hyperparameters tuned for a single task (*no lifetime tuning*). More details on the setup in Appendix D.

Figure 4 (bottom) compares vanilla, reset-at-task, and our per-layer scheduler across methods. With CReLU, our scheduler steadily improves on each task, whereas vanilla and reset stay near random accuracy. We attribute this initial failure of the vanilla baseline to the absence of regularization constraints (present in L2 and Wasserstein), which otherwise implicitly moderate the effective step-size and prevent immediate entry into a noise-dominated regime (see Appendix A.3). Under L2, our scheduler continues improving, while vanilla decays to random accuracy as the effective step-size gradually increases. Lastly, Wasserstein regularization exhibits mild decay in trainability, yet our controller maintains a higher and more stable average task accuracy.

Figure 5 shows the step-size trajectories produced by our scheduler. The step-size decreases as training progresses within a task, mirroring standard schedules that are know to yield improved performance. Moreover, the step-size schedule is adaptive to the effect of non-stationarity in combination with the learning algorithm. For L2, the step-size 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 step-size warms moderately, then plateaus with within-task cool-downs, matching the drop in sharpness volatility and the rise in accuracy. These step-size trajectories are significant because they are effective at improving performance, are achieved without tuning, and validate our analysis of the loss of trainability phenomenon.

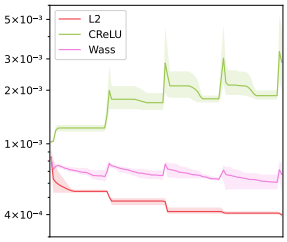


Figure 5: Scheduled step-sizes on the first 4 tasks.

7. Acknowledgements

The work was partially supported by the Canada CIFAR AI Chair Program and NSERC Discovery Grant RGPIN-2022-036669

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.

Kingma, Diederik P. and Jimmy Ba (2015). "Adam: A Method for Stochastic Optimization". In: *International Conference on Learning Representations*.

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.

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.

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.

Schaul, Tom, Sixin Zhang, and Yann LeCun (2013). *No More Pesky Learning Rates*. arXiv: 1206.1106 [stat.ML]. URL: https://arxiv.org/abs/1206.1106.

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. Experiments

A.1. Varying task-length

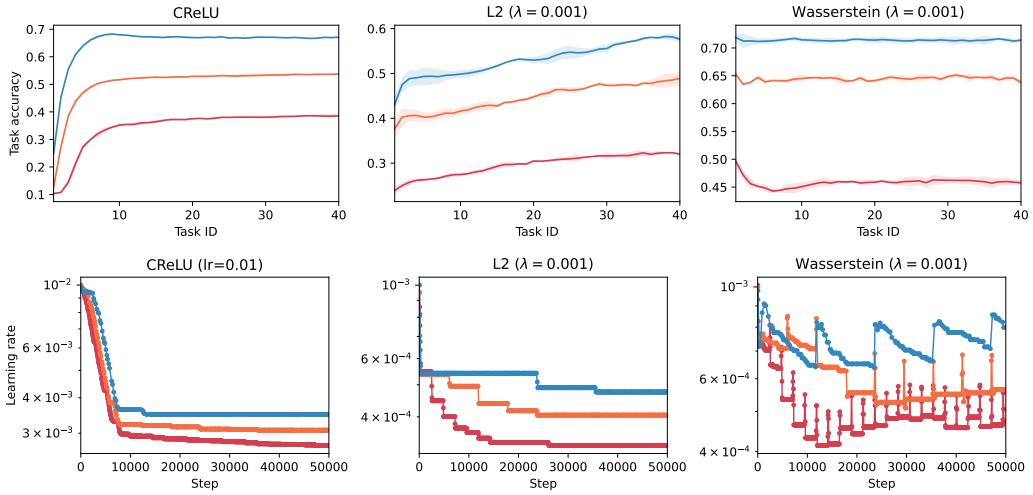


Figure 6: **Task accuracy (top) and learning rate (bottom) behavior across methods, ablating over task-lengths.** Red: 100 epochs/task. Orange: 250 epochs/task. Blue: 500 epochs/task. The scheduler successfully adapts to longer horizons, maintaining higher terminal learning rates and leveraging the increased budget to improve accuracy.

In this section, we analyze the sensitivity of our proposed scheduler to the duration of tasks. Figure 6 illustrates the evolution of average task accuracy and the adaptive learning rate trajectories for CReLU, L2 weight decay, and Wasserstein regularizer across three distinct task lengths: 100, 250, and 500 epochs per task.

We observe that increasing the task horizon consistently leads to improved asymptotic performance across all methods. This behavior is notable because standard implementations of these baselines often degrade on longer horizons; without adaptive step-size control, the accumulation of gradient noise and curvature volatility typically accelerates the onset of loss of trainability as the number of update steps increases. The monotonic improvement in accuracy (top row) suggests that our scheduler successfully decouples trainability preservation from task duration, allowing the model to effectively leverage the additional optimization budget.

The learning rate trajectories (bottom row) elucidate the mechanism behind this robustness. We observe that the adaptive schedules are qualitatively similar across task lengths but scale in magnitude. Specifically, for longer task durations (e.g., 500 epochs, shown in blue), the learning rate anneals less aggressively and stabilizes at a higher terminal value compared to shorter horizons (e.g., 100 epochs, shown in red). This indicates that the scheduler effectively identifies that the "safe" effective step-size region remains accessible for longer periods when the optimization is not rushed, thereby maintaining a higher learning rate to maximize convergence without crossing the adaptive noise threshold.

A.2. Ablation study

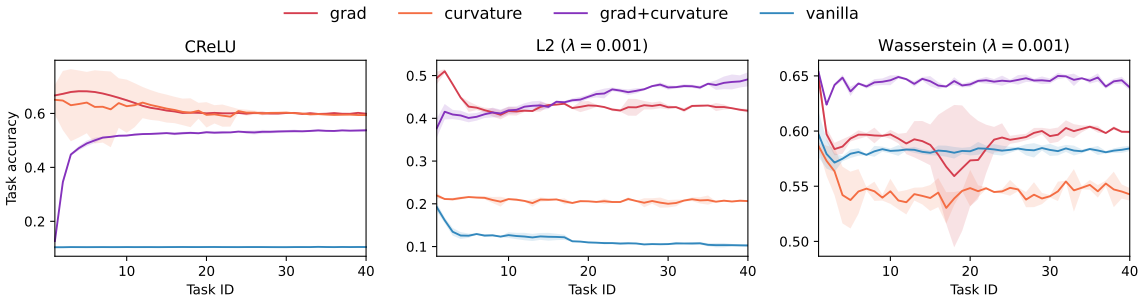


Figure 7: **Impact of removing different signals from combined bound.** The combined gradient and curvature bound consistently yields superior and more stable trainability across CReLU, L2, and Wasserstein regularizer, highlighting the complementary roles of both signals.

For CReLU, both grad-only (using α_g^*) and curvature-only (using α_{vol}^*) signals initially lead to a temporary loss of trainability before eventually stabilizing at a higher accuracy than vanilla or the combined method. This might suggest that relying on a single signal can lead to an overestimation of the safe step-size during initial tasks, causing transient instability.

With L2 regularization, the grad-only-signal method incurs loss of trainability at the start, but is able to maintain a higher accuracy compared to vanilla. In contrast, the curvature-only signal shows a slight improvement over vanilla but does not fully maximize performance. This behavior is elucidated by examining the critical step-size trajectories in Figure 8 (top row, L2 regularizer). The $\alpha_g^\star$ (red) bound for L2 often hovers at relatively high values, indicating that if only gradient noise is considered, the scheduler would perceive a safer regime than is actually present, leading to insufficient learning rate decay. The $\alpha_{\text{vol}}^\star$ (orange) bound, while more responsive, might preclude optimal performance. The combined bound successfully navigates this by dynamically weighing both factors, leading to a significant and sustained increase in accuracy.

For Wasserstein regularizer, the ablation results are particularly insightful. The grad-only signal performs poorly, the curvature-only signal (orange curve) yields performance worse than even the vanilla baseline after the initial tasks. This suggests that the $\alpha_{\text{vol}}^\star$ signal, when used in isolation for Wasserstein, may lead to excessive and premature learning rate decay, unduly limiting the model's capacity to learn. As observed in Figure 8 (bottom row, Wasserstein regularizer), $\alpha_{\text{vol}}^\star$ is indeed much more unstable and generally lower than $\alpha_g^\star$. Surprisingly, we see that the combined threshold is also lower than the gradient-only threshold, yet the learning rate, when controlled to lower than the combined bound leads to better performance.

The combined-bound method demonstrates consistent effectiveness in preserving trainability and achieving a high and stable average task accuracy across the above regularization methods.

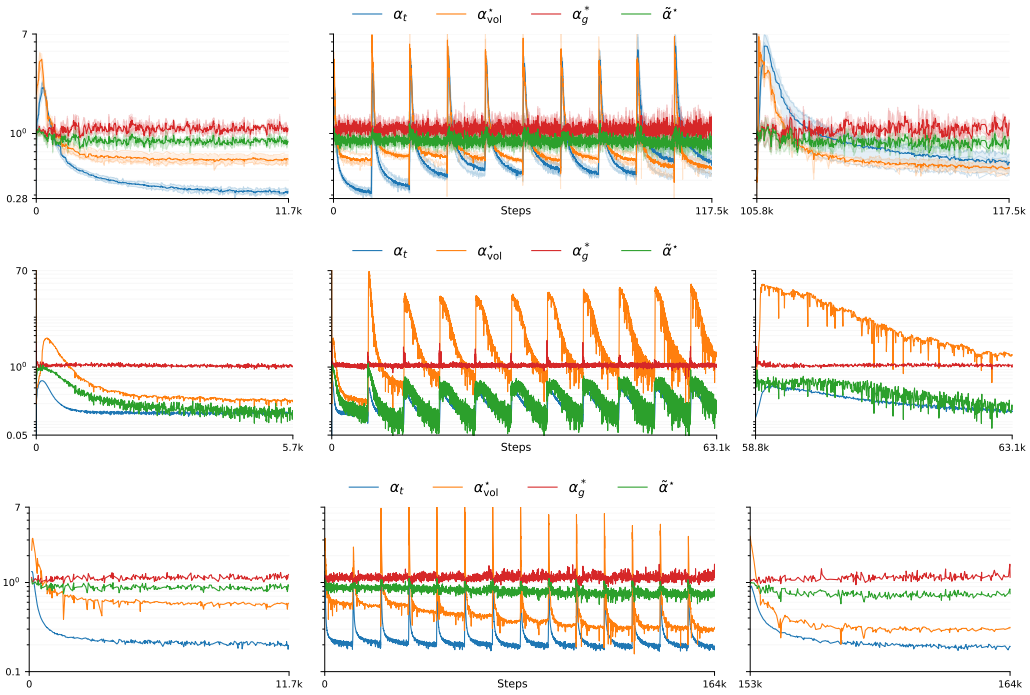


Figure 8: **Effective step-size vs. sharpness-aware safe bound.** We plot the first layer's effective step-size α_{t,l_1} with $\alpha^\star_{g,l_1}$, $\tilde{\alpha}^\star_{l_1}$, and $\alpha^\star_{\text{vol},l_1}$ for L2 regularizer (top), CReLU (middle), and Wasserstein regularer (bottom) ($\lambda = 10^{-3}$). Task 1 (left), Tasks $1-10$ (middle), and Task 10 (right).

A.3. Per-layer vs Full-network

In this section, we investigate whether loss of trainability is a global phenomenon or if it is localized to specific layers. Figure 9 presents a comparison between global (full-network) tracking and per-layer tracking of the effective step-size α_t against the critical stability bound $\tilde{\alpha}^*_t$.

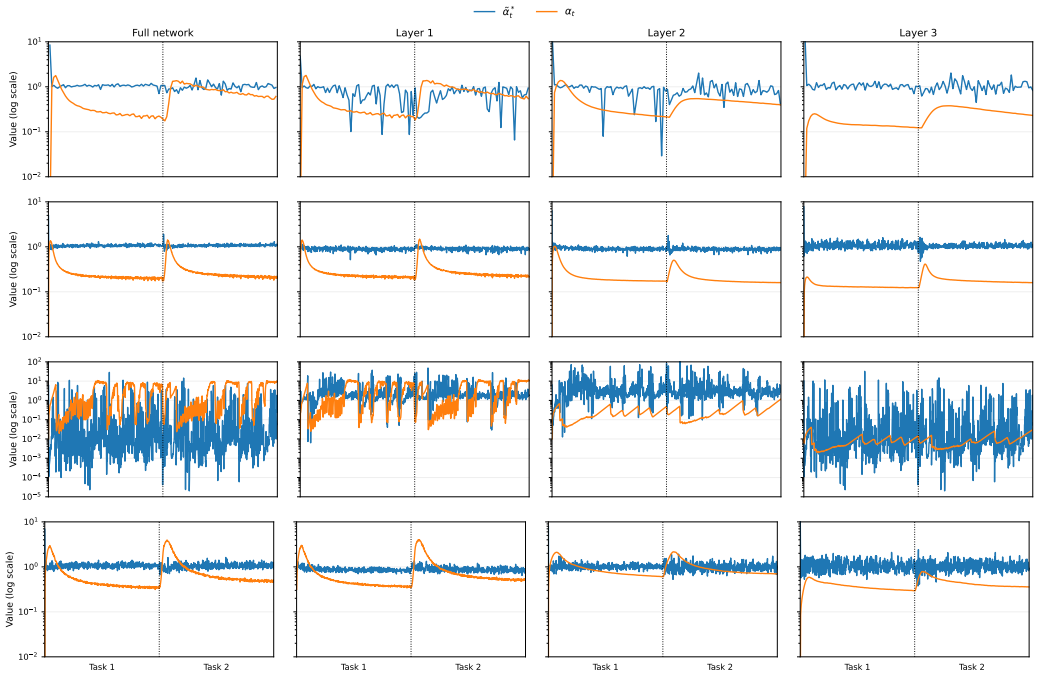


Figure 9: **Layer behavior.** From top to bottom : ReLU, Wasserstein regularizer, CReLU, L2 regularizer. The effective step-size α_t (orange) and critical bound $\tilde{\alpha}_t^*$ (blue) for the full network and individual layers. The global signal effectively mirrors Layer 1, often masking the stability of deeper layers.

Heterogeneity of layer dynamics (Figure 9). The first column displays the metrics calculated over the full network parameters. We observe that the global signal (leftmost column) is heavily correlated with the dynamics of the first layer (second column). Specifically, when the full network suggests that α_t exceeds the critical bound $\tilde{\alpha}_t^*$, it is almost exclusively driven by the instability in Layer 1. We can also see (Fig. 9) that for ReLU, at the start of task 2, the effective step size in Layer 1 is much higher above the threshold than is noted using the full network optimizer and sharpness statistics. In contrast, Layer 2 and Layer 3 often remain well within the safe regime (effective step-size below the critical threshold). Intuitively, the first layer is directly exposed to the raw input stream and thus bears the immediate brunt of the covariate shift at task boundaries. Deeper layers operate on latent representations which may be more stable, or at least exhibit different sensitivity scales.

This analysis also sheds light on the early failure of the vanilla CReLU baseline noted in Section 6. In the third row of Figure 9, we observe that the critical stability bound ($\tilde{\alpha}_t^*$, blue) for CReLU is highly volatile and frequently drops below the effective step-size (α_t, orange), especially in Layer 1. In the absence of regularization (like L2 or Wasserstein) to implicitly constrain the step-size or smooth the curvature volatility, the vanilla optimizer operates in a violation regime ($\alpha_t > \tilde{\alpha}_t^*$) immediately.

Per-layer Calculation. To capture this heterogeneity, we compute the effective step-size $\alpha_t^{(\ell)}$ for a specific layer ℓ by averaging the element-wise Adam step multiplier across all parameters within that layer's top-level module. Formally, for a parameter p at step t, the effective step is

$\eta(1-\beta_1^t)^{-1}/(\sqrt{\widehat{v}_t}+\epsilon)$. We aggregate this scalar value for all weights and biases within a layer to derive $\alpha_t^{(\ell)}$. This allows us to detect when a specific layer enters a curvature-noise-dominated regime, even if the global aggregate statistics suggest stability (or vice-versa).

Per-layer vs. full-network control (Figure 10). The consequence of this heterogeneity is shown in the performance comparison. A global controller, which modulates the learning rate based on the full-network signal, is forced to penalize the entire model based on the worst-case behavior (typically Layer 1). This unnecessarily constrains deeper layers that are statistically healthy, reducing their capacity to learn. By contrast, our per-layer controller (purple curve) acts independently: it cools the learning rate only for the specific layers violating their stability bounds (typically Layer 1 during high-volatility phases) while allowing stable layers (Layers 2 and 3) to maintain higher learning rates. This results in significantly higher task accuracy compared to the full-network controller (red curve), confirming that there is no single step-size that is optimal for all layers simultaneously.

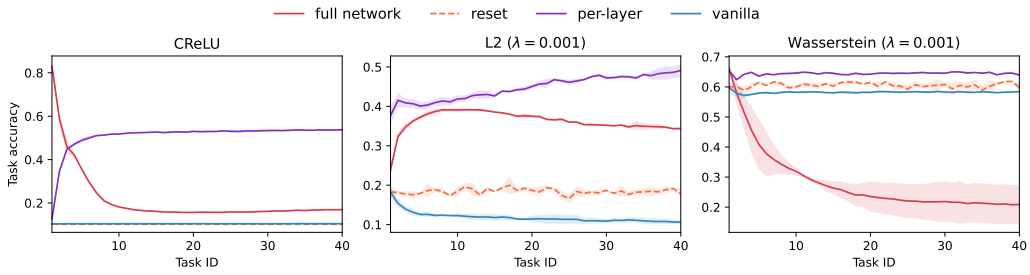


Figure 10: **Per-layer vs. Full-network controller performance.** Task accuracy comparing a controller driven by full-network statistics (red) versus per-layer statistics (purple). The per-layer approach significantly outperforms the global approach.

Appendix B. Theoretical details

B.1. 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 λ_t and the *normalized (Adam-adjusted)* sharpness $\bar{\lambda}_t$ (and its variability) rise. This elevates the volatility $\mathrm{Vol}_{\bar{\lambda}}(t)$ defined in Eq. (1), which *shrinks* the curvature–volatility critical step-size

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

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-size $\alpha^{\star}_g(t)$ together with the effective step-size α_t: if $\|\widehat{g}_t\|$ shrinks while the per-sample variance stays high (or α_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. Crucially, because LoT is generally assessed as a reduction in average task accuracy, perceived loss of trainability might occur without a measured effect in the "indicators" if Adam functions in a

regime of noise-dominated steps even when the curvature remains healthy (see α_t at the start of tasks in Fig. 3).

Conversely, high Hessian rank can coexist with LoT when either *noise-dominated* crossings $\alpha_t > \alpha_g^\star(t)$ or *curvature-noise-dominated* crossings $\alpha_t > \alpha_{\text{vol}}^\star(t)$ proliferate; low-rank regimes can remain trainable when both criteria are comfortably satisfied—explaining Fig. 1. We refer the reader to Appendix B.1 for a discussion of when Hessian rank decay aligns with these signals.

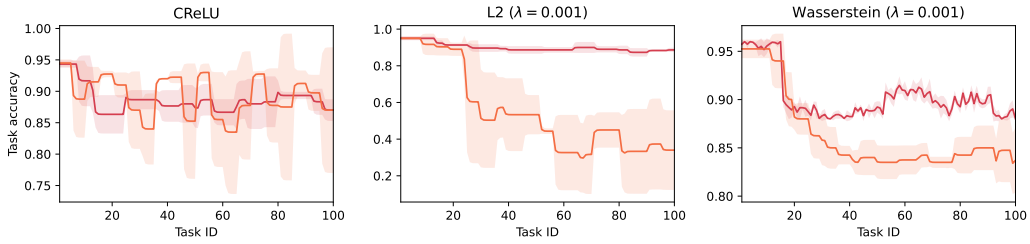


Figure 11: **Hessian rank outcomes across methods.** The adaptive scheduler (red) helps mitigate the spectral collapse often observed in vanilla training (orange), particularly under L2 regularization where it maintains a higher effective rank throughout the task sequence.

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-size

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

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 $\alpha_{g,(r)}^\star(t)$ 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 $\text{Vol}_{\tilde{\lambda}}(t)$ (Eq. (1)) and hence *shrinks* $\alpha_{\text{vol}}^\star(t)$ *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 $\alpha_{g,(r)}^\star(t)$ and $\alpha_{\text{vol}}^\star(t)$. As α_t crosses either threshold persistently, LoT follows.

B.2. Gradient Noise

Here we detail the derivation of $\alpha_g^\star(t)$ in Section 4. While Adam employs preconditioned updates, we analyze the stability of the effective step magnitude α_t along the chosen update direction. Assume the loss J is locally L-smooth along this direction. Let the update be $w_{t+1} = w_t - \alpha_t \widehat{g}_t$, where $\widehat{g}_t$ is the stochastic update vector (e.g., the preconditioned gradient in Adam). Let $g_t = \mathbb{E}[\widehat{g}_t]$ be the true direction and $\widehat{g}_t = g_t + \xi_t$ with noise ξ_t such that $\mathbb{E}[\xi_t] = 0$ and $\mathbb{E}[\|\xi_t\|^2] = \text{Var}[\widehat{g}_t]$.

A quadratic expansion of the loss yields:

$$J(w_{t+1}) \approx J(w_t) - \alpha_t \langle \nabla J(w_t), \widehat{g}_t \rangle + \frac{1}{2}\alpha_t^2 \lambda_t \|\widehat{g}_t\|^2,$$

where λ_t is the curvature along the update direction ($\lambda_t \leq L$). Taking expectations over the noise:

$$\mathbb{E}[J(w_{t+1}) - J(w_t)] \leq -\alpha_t \|g_t\|^2 + \frac{1}{2}\alpha_t^2 L \left(\|g_t\|^2 + \mathrm{Var}[\widehat{g}_t]\right).$$

For expected descent (LHS < 0), we require:

$$\alpha_t \|g_t\|^2 > \frac{1}{2}\alpha_t^2 L \left(\|g_t\|^2 + \mathrm{Var}[\widehat{g}_t]\right) \implies \alpha_t < \frac{2}{L}\frac{\|g_t\|^2}{\|g_t\|^2 + \mathrm{Var}[\widehat{g}_t]}.$$

Using the per-sample variance proxy $\sigma_{t,\mathrm{ps}}^2$ where $\mathrm{Var}[\widehat{g}_t] \approx \sigma_{t,\mathrm{ps}}^2/B$, and noting that typically $\|g_t\|^2 \ll \sigma_{t,\mathrm{ps}}^2/B$ in the noise-dominated regime, we obtain the bound proportional to the signal-to-noise ratio derived by Schaul et al. (2013):

$$\alpha_t \lesssim \frac{2}{L}\frac{B\|g_t\|^2}{\sigma_{t,\mathrm{ps}}^2}.$$

B.3. 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,

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

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.4. 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, we treat $\{\bar{\lambda}_s\}$ as a scalar stochastic process. We track its statistics using an exponential moving average with decay ν (e.g., $\nu = 0.9$):

$$\mu_t := \frac{(1-\nu)\mu_{t-1} + \nu\bar{\lambda}_t}{1-\nu^t}, \quad \sigma_t^2 := \mathrm{EMA}_\nu[(\bar{\lambda}-\mu)^2]_t.$$

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

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

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:

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

Noise-dominated regime. When $\sigma_t^2 \gg \mu_t^2$, Eq. (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-size

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

with κ absorbing the constants (the choice of c, the small ε, and mismatch between the scalar surrogate $\bar{\lambda}$ and $\lambda_{\max}(D_t^{1/2}H_tD_t^{1/2})$). Intuitively, Eq. (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 σ_t^2/μ_t, i.e., $\mathrm{Vol}_{\bar{\lambda}}$.

From Eq. (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 μ_t is not dwarfed by σ_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 α_t by Eq. (A.5) for extra safety.

B.5. 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 μ_t and/or increased σ_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 θ_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 ρ_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 α_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-size (global & per-layer). We report the *effective step-size* α_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-size $\alpha_{t,\ell}$ 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 λ_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 ℓ 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.

Hessian rank. Following Lewandowski et al. (2024b), we approximate the local curvature complexity using the effective rank of the Empirical Fisher Information Matrix (EFIM). Given a subset of M samples (here $M = 100$), we collect per-sample gradients into a matrix $G \in \mathbb{R}^{M\times P}$, where P is the number of parameters. To avoid the computational cost of the full $P \times P$ matrix, we compute the spectrum of the Gram matrix $K = GG^\top \in \mathbb{R}^{M\times M}$.

C.2. Experiments

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

Refer to D.3 for the model and hyperparameter specifications.

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 γ, cool rate $c = 1 - \varepsilon$, warm rate $u = 1 + \varepsilon$; $\varepsilon \approx 10^{-2}$, interval K

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

  Estimate per-layer sharpness $\lambda_t^{(\ell)}$ (top Hessian eigenvalue), calculate normalized sharpness $\bar{\lambda}_t^{(\ell)}$, and minibatch grad variance $\sigma_{t,\mathrm{mb}}^{2(\ell)}$

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

  Compute $\tilde{\sigma}_t^{2(\ell)}$, $\tilde{\alpha}_t^{\star(\ell)}$, and safe bound $\tilde{\alpha}_t^{\mathrm{safe}(\ell)} = (1-\epsilon)\tilde{\alpha}_t^{\star(\ell)}$ `// via Eq. (2)`

  Form effective step-size $\alpha_t^{(\ell)}$ from optimizer state

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

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

  else

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

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

    end

  end

end

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 optimizer steps with small warming/cooling updates, and power-iteration budgets are kept short to bound overhead . With longer tasks, our controller maintains trainability.

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 width of 256.

D.3. Hyperparameters

General Hyperparameters	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
β_1	0.9
β_2	0.999
Base learning rate	$\eta = 10^{-3}$ ($10^{-2},\ 10^{-4},\ 10^{-5}$ when specified)
L2 weight decay (λ_{L2})	10^{-3}
Wasserstein reg. (λ_{Wass})	10^{-3}
Leaky-ReLU factor (ρ)	$0.7,\ 0.3$
Safety factory γ	0.8
Cool rate c	0.99
Warm rate u	1.01
Logging interval	40
Controlling interval (K)	40

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

Figure 1(a) Hyperparameters	**Value**
Learning rate	10^{-4}
L2 weight decay (λ_{L2})	10^{-3}
Wasserstein reg. (λ_{Wass})	10^{-3}

Figure 1(b) Hyperparameters	**Value**
Learning rate	10^{-3}
L2 weight decay (λ_{L2})	10^{-5}
Leaky-ReLU factor (ρ)	0.3

Figure 1(c), (d) Hyperparameters	**Value**
Learning rate	10^{-3}
Leaky-ReLU factor (ρ)	$0.3,\ 0.7$

Figure 1(e) Hyperparameters	**Value**
Learning rate	10^{-5}
L2 weight decay (λ_{L2})	10^{-3}
Wasserstein reg. (λ_{Wass})	10^{-3}

Figure 1(f) Hyperparameters	**Value**
Learning rate	10^{-3}
Leaky-ReLU factor (ρ)	0.01

Table 2: Method-based hyperparameters used across experiments in Figure 1. Other hyperparameters are the same as mentioned in Table 1.