A JOURNEY TO THE EDGE OF STABILITY

Anonymous authors
Paper under double-blind review

ABSTRACT

It has recently been found that deep learning often occurs at the "edge of stability (EoS)," where the maximum Hessian eigenvalue of the model is stabilized at a value reciprocal to the learning rate. However, what happens *before* we reach that regime? We fix a deep learning problem and vary first order optimization methods with dense learning rate sweeps. We then track the characterizing quantities of a learning trajectory: the loss, the sharpness, and the alignment between consecutive gradients. To our surprise, if we scale the learning rate by the dc gain of the optimizer, these traces from the sweeps from different optimizers almost perfectly overlap across a large range of learning rates. The dc-normalized optimizers have another role that only becomes apparent in high learning rates: they select when the sharpness value detaches from this universal curve and enters the edge of stability. Upon this discovery, we specify three distinct regimes with respect to the dc-adjusted learning rate: the *low-LR regime* where the trajectory is nearly insensitive to the optimizer, the *high-LR regime*, where the optimizer governs the sharpness according to the EoS reciprocal rule, and the in-between *mid-LR regime* where so-called *progressive sharpening* originates independently of the optimizer. This distinguishes the role of the optimizer, the learning rate, and the model in shaping the learning progress.

1 INTRODUCTION

Modern deep learning is mainly driven by so-called first order methods, i.e., the (stochastic) gradient descent and its variants. Under gradient descent, the parameter $\boldsymbol{\theta}$ moves according to a scaled gradient $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$, which can be viewed as a discretization of a continuous-time gradient flow $\dot{\boldsymbol{\theta}} = -\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$ with a sampling period η. However, it is recently observed that the difference between the two dynamics—the high-order terms, or the *aliasing* terms if we want to take the signal processing perspective—becomes significant in practical settings where the learning rate η operates near its maximally stable value. This aliasing effect develops into a unique characteristic feature of the discrete-time gradient descent, called the *edge of stability* (EoS) (Cohen et al., 2021). For deterministic full-batch gradient descent, the maximum sharpness $S := \lambda_{\max}(\nabla^2_{\boldsymbol{\theta}} L(\boldsymbol{\theta}))$ of the loss function $L(\boldsymbol{\theta})$ is often attracted and stabilized at the value $S \approx 2/\eta$, under which the gradient descent is certain to produce monotonic loss decrease if the loss is quadratic (Ortega & Rheinboldt, 1970; Bertsekas, 1999). Hence the name *edge of stability*.

Similar phenomenon has been observed for preconditioned and momentum-augmented gradient descent (Cohen et al., 2021; 2023). Andreyev & Beneventano (2024); Andreyev et al. (2026) have further generalized this phenomenon to more practical mini-batch settings, incorporating stochastic gradient descent (SGD) and SGD with momentum (Polyak, 1964; Nesterov, 1983). Later, Lee & Lee (2026) have added a missing term in the relationship $S \approx 2/\eta$ by treating the optimizer as a gradient filter to define its transfer function $Q(z)$, dc gain $Q(1)$, and root locus escape gain $\Gamma(Q)$:

$$S \approx \frac{2}{\eta Q(1) \Gamma(Q)}. \qquad (\spadesuit)$$

Overall, it is now well-established that there are essentially discrete-time behaviors of the first order methods in deep learning, summarized in the name: *edge of stability* (EoS) (Cohen et al., 2021). But how does this phenomenon emerge from the continuous-time gradient flow? What happens before the edge of stability? What are the roles of the optimizer, the learning rate, and the model in producing this phenomenon? When, how, and how much is each of them reponsible in steering

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

the learning progress? Our goal is to answer these questions by inspecting the learning progress of a *single* deep learning problem solved with various optimizers and learning rates.

We focus on a single large-batch stochastic scenario of learning a CIFAR-10 dataset (Krizhevsky, 2009). Inspired by previous work (Andreyev et al., 2026; Lee & Lee, 2026), we devise various optimizers, including Heavy Ball (SGDM) (Polyak, 1964), Nesterov's momentum (SGDN) (Nesterov, 1983), two-pole cascade optimizers (Lee et al., 2024), and quasi-hyperbolic momentum (QHM) optimizers (Ma & Yarats, 2019), to vary the optimizer's transfer function $Q(z)$, the gains $Q(1)$ and $\Gamma(Q)$, and the unit circle escape angle in the root locus plot. For each of the optimizer, we densely sweep the raw learning rate η in the wide range from 10^{-7} to 10^{0} in log scale. The results are beyond our expectations. We observe three distinct regimes separated in the *effective learning rate* $h := \eta Q(1)$, each of which is characterized by a unique nonlinear relationship between the learning rate η, the stabilized sharpness S, the loss L, and the optimizer's transfer function $Q(z)$. We dub these regimes as the *low-LR softening regime*, *mid-LR sharpening regime*, and *high-LR edge regime*, respectively. In the follwing sections, we will elaborate on the characteristics of the learning dynamics in each of these regimes in detail. In specific, our contributions are:

Separable roles of problem geometry and optimizer. Training with a single deep learning problem across a broad family of optimizers reveals a remarkably consistent pre-edge traces of the loss L and the sharpness S. The effect of changing an optimizer only takes place near and on the edge of stability. The problem geometry and the optimizer memory play time-scale-separable roles.

Mechanism of optimizers determining the edge of stability. We show that the relationship of type (♠) universaly occurs in diverse stochastic first order methods, including multi-pole (Lee et al., 2024) and pole-zero (Ma & Yarats, 2019) optimizers. To our knowledge, this is the first empirical evidence that the optimizer root locus determines the edge of stability in stochastic settings.

Rethinking optimizers as gain-shape compounds. The Low-LR traces of the model geometry are insensitive to the optimizer, coinciding in the *effective* lr $h := \eta Q(1)$ coordinate. This suggests that an optimizer Q is a compound of separable mechanistic parts. Its dc gain $Q(1)$ is more sensible to be absorbed into lr, together governing the universal trajectories in the Low-LR regimes. Whereas, the temporal reshaping kernel $\bar{Q}(z) := Q(z)/Q(1)$ determines the edge of stability.

Mechanism of progressive sharpening. We empirically show what determines the mysterious mechanism of *progressive sharpening* reported in the EoS literature (Cohen et al., 2021; Damian et al., 2023). The manifestation of this phenomenon is determined by the model geometry, independently of the optimizer. However, its realization depends on the effective lr $h = \eta Q(1)$.

2 Observations

Problem formulation. A *deep learning problem* is a tuple

$$\mathcal{P} = (\mathcal{D}, \mathcal{P}_b, \boldsymbol{\theta}, L(\boldsymbol{\theta}), \boldsymbol{\theta}_0, \eta, Q), \tag{1}$$

where $\xi \in \mathcal{D}$ is the dataset, $\{\xi_i\}_{i=1}^{b} \sim \mathcal{P}_b$ is the sampling distribution of size b, $\boldsymbol{\theta}$ is the parameter with a topology, $L(\cdot;\boldsymbol{\theta})$ is the model-loss compound, i.e., the manifestation of $\boldsymbol{\theta}$ on the scalar objective in response to a subset of $\mathcal{D}$, $\boldsymbol{\theta}_0$ is the initialization, η is the learning rate scalar, and Q is the optimizer *filter* (Lee et al., 2024). We write the empirical risk $L(\boldsymbol{\theta}) := \mathbb{E}_{\mathcal{B}\sim\mathcal{P}_b}[L(\mathcal{B};\boldsymbol{\theta})]$ and its gradient $\boldsymbol{g}(\boldsymbol{\theta}) := \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$ and Hessian $\boldsymbol{H}(\boldsymbol{\theta}) := \nabla^2_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$. The sharpness is defined as $S(\boldsymbol{\theta}) := \lambda_{\max}(\boldsymbol{H}(\boldsymbol{\theta}))$. Optimizer preconditioners (Duchi et al., 2011; Tieleman & Hinton, 2012; Kingma & Ba, 2015) separate sharpnesses into preconditioned and unpreconditioned variants, complicating the analysis. We treat the effect of preconditioning out of scope of this work, and focus on the memory structure of the optimizer Q.

An optimizer Q is an optionally stateful causal linear filter on a stochastic gradient stream $\boldsymbol{g}_{\le t}$. Filter means that its operation summarizes input sequences through a linear convolution with its impulse response Q_t. That is, $Q * \boldsymbol{g}_{\le t} = \sum_{j=0}^{\infty} Q_j \boldsymbol{g}_{t-j}$. Under a first order method, we update $\boldsymbol{\theta}$ as

$$\boldsymbol{\theta}_{t+1} \;=\; \boldsymbol{\theta}_t - \eta (Q * \boldsymbol{g}_{\le t})_t \,. \tag{2}$$

This class includes widely used SGD (Robbins & Monro, 1951) and the two most famous variants of momentum: Heavy Ball (Polyak, 1964) and Nesterov (Nesterov, 1983). To study optimizers with an emphasis on their signal filtering mechanism, we additionally consider the following two classes of optimizers: dual-momentum cascade of Grokfast (Lee et al., 2024), and quasi-hyperbolic momentum (QHM) (Ma & Yarats, 2019). These two additional classes correspond to well-used linear filters: two-pole and pole-zero filters, respectively. Treating optimizers as filters lets us study them in the frequency dual domain, i.e., z-domain: $Q(z) = \sum_{t=0}^{\infty} Q_t z^{-t}$. We write $Q(1)$ for the dc gain of this filter. The *effective (dc-matched) learning rate* is defined as $h := \eta Q(1)$.

Optimizer root locus. Treating gradient-based optimizers as filters unlocks a powerful tool to study its behavior: a root locus (Ogata, 1995). Following Lee & Lee (2026), we consider a dynamics along an eigenpair $(\lambda, \boldsymbol{v})$ of $\boldsymbol{H}$. Let the parameter $\boldsymbol{\theta}$ in this coordinate be $x := \boldsymbol{v}^\top(\boldsymbol{\theta} - \boldsymbol{\theta}_\star)$, where $\boldsymbol{\theta}_\star$ is the local minimum along the direction $\boldsymbol{v}$. Following the derivation in Lee & Lee (2026), we get the characteristic equation of the transfer function $Q(z)$

$$z - 1 + \eta\lambda Q(z) \;=\; 0. \tag{3}$$

We further define a dc-normalized transfer function $\bar{Q}(z) := Q(z)/Q(1)$ to separate gain control from the filter shape (Lee & Lee, 2025). Focusing on the eigenpair $(S, \boldsymbol{v}_{\max})$ of the maximum eigenvalue (sharpness) $S = \lambda_{\max}(\boldsymbol{H})$, let $k := hS$. Then the characteristic equation becomes

$$z - 1 + k\bar{Q}(z) \;=\; 0. \tag{♣}$$

This equation governs the relationship between the effective learning rate h (gain), the sharpness S, and the filter shape $\bar{Q}(z)$. In practical scenarios, we first define the filter shape $\bar{Q}(z)$, i.e., the optimizer family, such as SGD or SGDM, and then tune the gain $h = \eta Q(1)$.

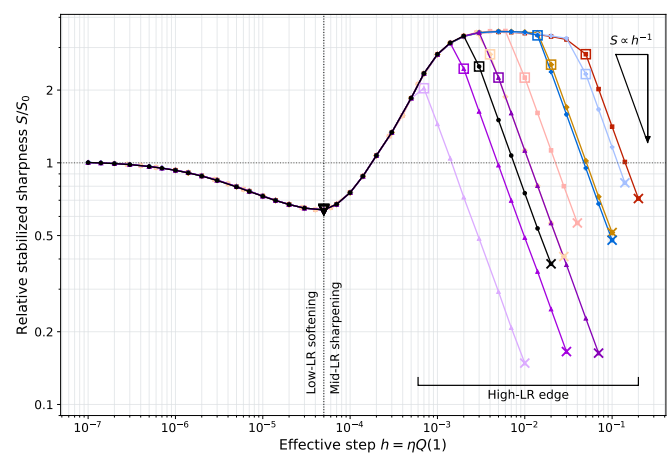
Experimental setup. Training dynamics are determined by multitudes of components. Our main focus is on the joint effect of the optimizer Q, the learning rate η, and the geometry of the loss landscape $L(\boldsymbol{\theta})$. Therefore, the remaining components are fixed throughout the analysis. We fix the task to CIFAR-10 (Krizhevsky, 2009) and consider three representative architectures: a two-layer tanh MLP having our special attention, and a four-layer tanh CNN and a small three block ViT (Dosovitskiy et al., 2021). The above number of layers did not count the common linear classification head. We fix the batch size to 4096 for all experiments, which is in the *large batch regime* specified by Andreyev et al. (2026). A diverse set of optimizers are considered in this study, classified into pole-zero structures in their transfer function of $Q(z)$: SGD, SGDM, SGDN, QHM, and two-pole cascade (Grokfast). In total of 10 optimizers are considered. Using each of these filters, we train the same model from a same initialization point. This is to fix the initial geometry and study the trajectory deviation caused by the optimizer. We vary the learning rate over a wide range, roughly equally spaced on a log scale: $\{1.0, 1.4, 2.0, 3.0, 5.0, 7.0\} \times 10^{\{-7,-6,\ldots,0\}}$.

Measurements. Throughout 200k iterations, we collect the sharpness S, the loss L, and two types of flow smoothness measures: a *consecutive gradient alignment* c_t as the cosine similarity, and a *first order alias* ρ_t as the normalized difference of the consecutive gradient pair:

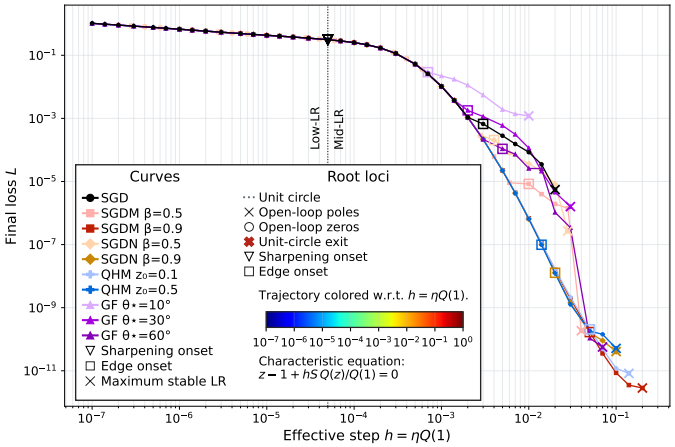
$$c_t \;:=\; \frac{\langle \boldsymbol{g}_t, \boldsymbol{g}_{t+1}\rangle}{\|\boldsymbol{g}_t\|\|\boldsymbol{g}_{t+1}\|} \;\in\; [-1, 1], \qquad \rho_t \;:=\; \left|\frac{1}{2} - \frac{Q'(1)}{Q(1)}\right| \frac{\|\boldsymbol{g}_{t+1} - \boldsymbol{g}_t\|}{\|\boldsymbol{g}_t\|} \;\in\; [0, \infty), \tag{4}$$

where time $\tau := ht = t\eta Q(1)$ is defined as the product of the effective learning rate h, i.e., the sampling period, and the algorithm iteration index t. This gets the governing gradient flow normalized as $\mathrm{d}\boldsymbol{\theta}/\mathrm{d}\tau = -\boldsymbol{g}$. The alignment c_t measures the effect of Hessian-induced gradient rotation. In Proposition B.4, the alias ρ_t is shown to be equivalent to the relative contribution of the higher order terms in the Taylor expansion up to $O(h^2)$. Among these, the sharpness $\lambda_{\max}(\boldsymbol{H}_t)$ may oscillate wildly, especially at the edge of stability. However, taking its median over a fixed window size of 51 samples greatly stabilizes the value to let us study its behavior more clearly. We use this scalar *stabilized sharpness* as the representative curvature signature.

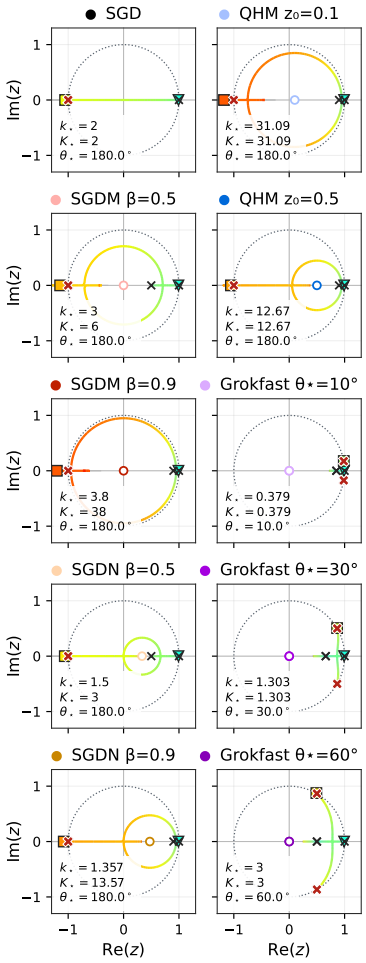
Main results. We now summarize our main findings in Figure 1 and 2. Figures 1a and 1b plot two geometric quantities, the stabilized sharpness S, and the final loss L, respectively, with respect to the aligned x-axis of effective learning rate $h = \eta Q(1)$ in logarithmic scale. Each dot summarizes a full



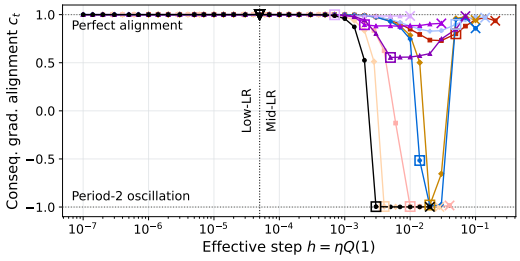
(a) Stabilized sharpness S vs. effective lr $h = \eta Q(1)$.



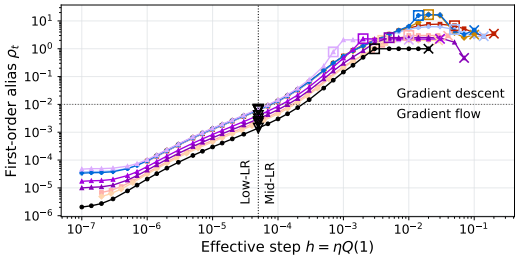
(b) Final loss L vs. effective lr $h = \eta Q(1)$.



(c) Full root locus plots.



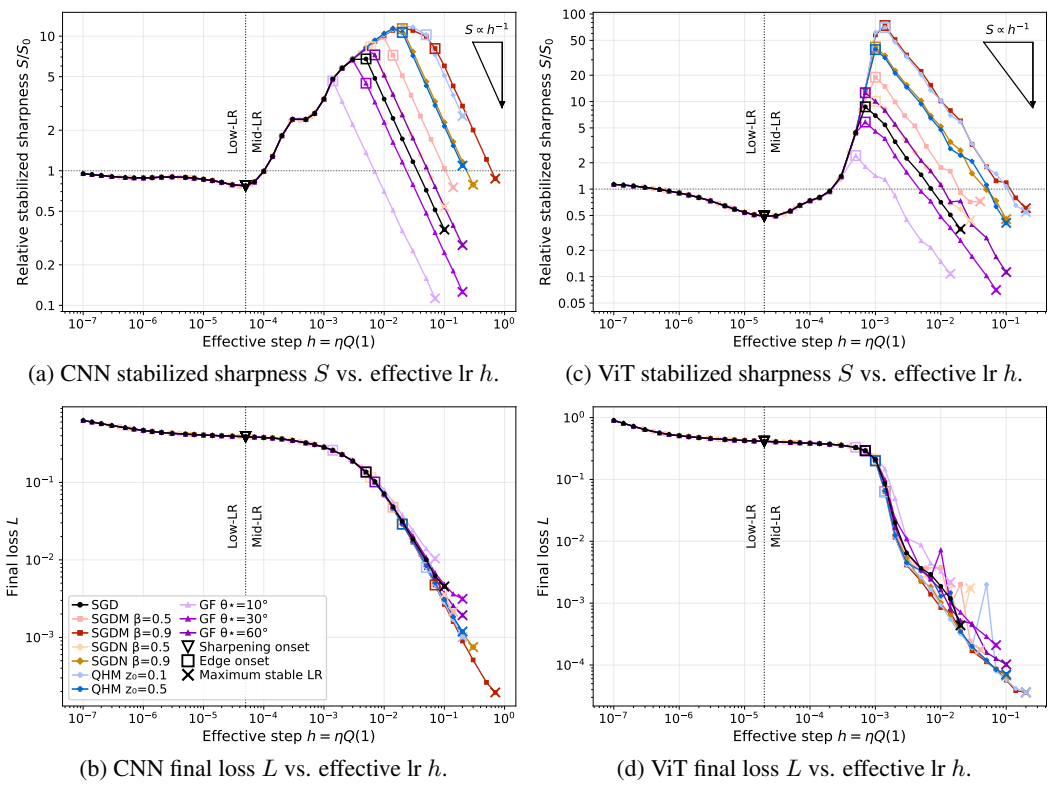
(d) Consecutive gradients alignment c_t.



(e) First order alias ρ_t of GD vs. GF.

Figure 1: **Main results.** All results drawn in the same figure to visualize coincidence. Each line connects a learning rate sweep over a single optimizer. In the effective lr h coordinate, the operating range of this learning system is naturally partitioned into three distinct regimes. In the low and mid-LR regimes, the results coincide in an optimizer-independent curves. In the edge regime, the optimizer plays a role in determining the edge of stability, marked by the strict oblique lines $S \propto 1/h$ in Figure 1a. Consecutive gradient alignment c_t and first order alias ρ_t marks the natural transition boundaries between these three regimes, marked by the triangle (between low and mid-LR) and square (between mid-LR and edge) markers.

200k training steps using a single optimizer and a learning rate, and the runs with the same optimizer with different learning rates are connected by a line, indicating a sweep. Figure 1c shows the full root locus plots for 10 optimizers with marked regimes of low-LR, mid-LR, and edge. Figures 1d and 1e plot auxiliary smoothness measures c_t and ρ_t that help distinguish the three regimes. Likewise in Figures 1a and 1b, sharpness and final loss with respect to the effective lr $h = \eta Q(1)$ are plotted for CNN and ViT models in Figure 2.



(a) CNN stabilized sharpness S vs. effective lr h.

(c) ViT stabilized sharpness S vs. effective lr h.

(b) CNN final loss L vs. effective lr h.

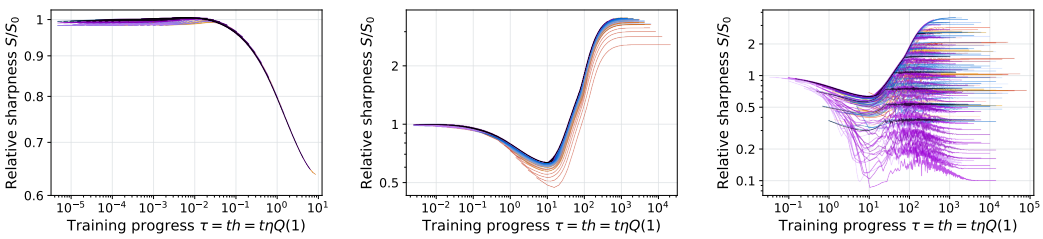
(d) ViT final loss L vs. effective lr h.

Figure 2: **Geometry repsonses of CNN and ViT.** We observe common structures of low-LR softening $\mathrm{d}S/\mathrm{d}h < 0$ and mid-LR sharpening $\mathrm{d}S/\mathrm{d}h > 0$ in the sharpness S curve. However, the curve shapes are unique to each architecture, suggesting that this common curves are the model intrinsic *material properties* independent of optimization algorithms.

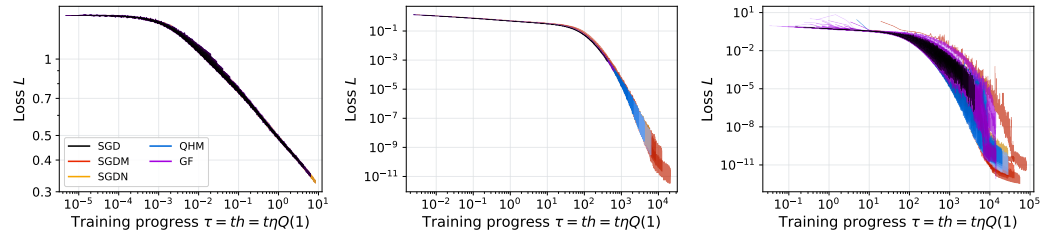
Three key observations can instantly be made from the figures. First, to our surprise, the results from different optimizer families give strikingly similar shapes, especially in the wide range of lower learning rates. If we align the results with respect to the effective leanring rate h instead of raw η, the plots almost perfectly overlap, except for the final reciprocal slope of the edge of stability (Cohen et al., 2021).

Second, the geometric responses (L, S) naturally partitions the lr sweep results into three distinct regimes with respect to the effective lr $h = \eta Q(1)$. (1) In the *low-LR softening regime*, the sharpness drops $\mathrm{d}S/\mathrm{d}h < 0$, while the loss L decreases only slightly. All optimizers give almost identical curves for this regime. (2) In the *mid-LR sharpening regime*, the sharpness starts to grow $\mathrm{d}S/\mathrm{d}h > 0$, while the loss drop becomes amplified. In this regime, the root loci traverse from their initial points to eventually escape the unit circle. As a sweep trajectory approaches near *its own* edge of stability, the consecutive gradients alignment c_t starts to break down rapidly. At the same time, the first order alias ρ_t becomes significant $\rho_t > 1\%$, and thus the discrete-time descent algorithm can no longer be accurately approximated using the continuous-time flow model. (3) In the final *high-LR edge regime*, the sharpness S becomes strictly reciprocal to the effective lr $S \propto 1/h$, following the edge of stability rule (♠). Figure 5 separates the SGD case to visualize this transition more clearly.

Third, comparing the sharpness curves of MLP (Figure 1a), CNN (Figure 2a), and ViT (Figure 2b), we can see that the shapes of the optimizer-independent sharpness S curves in the low- and mid-LR regimes are unique to each architecture. This shape is thus a model-intrinsic *material property*. We use the term *material property* to emphasize a property that is predetermined by the problem geometry and independent of the optimization algorithm. We can additionally observe that all three representative architectures share the same *three regime structures*: the sharpness initially softens in the low-LR regime, sharpens in the mid-LR regime to finally reaches the optimizer-specific edge of stability in the high-LR regime. In the following sections, we study each of the three regimes in detail, with a special focus on the MLP results in Figure 1.



(a) Low-LR τ vs. S. (b) Mid-LR τ vs. S. (c) High-LR τ vs. S.



(d) Low-LR τ vs. L. (e) Mid-LR τ vs. L. (f) High-LR τ vs. L.

Figure 3: **Geometry responses of MLP with full training traces.** Instead of picking only the final values as in Figures 1a and 1b, we sample intermediate values at every 50 iterations for each of the 338 non-blow-up runs, collecting a total of 1.352M entries. We plot each run as a connected line drawn in the same plot, aligned in the training progress $\tau = ht = \eta Q(1)t$ coordinate. The low-LR traces almost perfectly overlap, whereas the high-LR traces show clearly different trajectories.

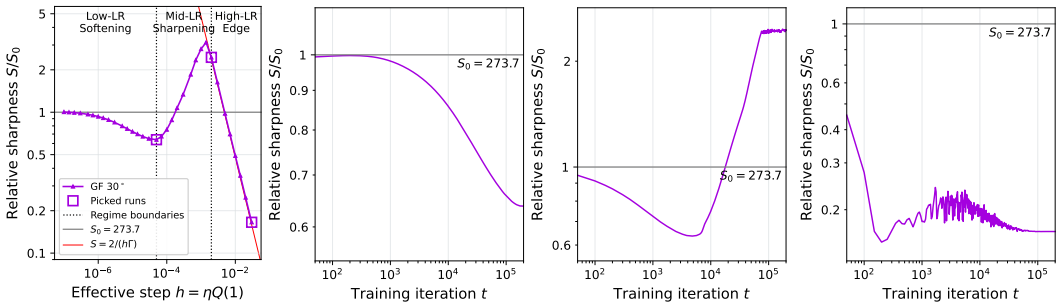
3 Regime I: Low-LR Softening

The first of the three regimes, the low-LR regime, is characterized by its (1) overlapping traces in the effective lr h and (2) in the training progress τ coordinates, (3) sharpness softening $\mathrm{d}S/\mathrm{d}h < 0$, and (4) shallow loss drop.

Overlapping traces in h coordinate. The first thing we notice in Figures 1a and 1b is that the plots from different optimizers become almost indistinguishable when we align the curves using the effective learning rate h coordinate. The difference between lr η and effective lr h is the dc gain $Q(1)$ of the optimizer. Normalized optimizers such as simple SGD or normalized Heavy Ball $\boldsymbol{\mu}_{t+1} = \beta\boldsymbol{\mu}_t + (1-\beta)\boldsymbol{g}_t$ have a unit dc gain $Q(1) = 1$, hence $h = \eta$. Conventionally, however, HB or NAG momenta are defined as unnormalized forms: $\boldsymbol{\mu}_{t+1} = \beta\boldsymbol{\mu}_t + \boldsymbol{g}_t$, yielding nontrivial dc gains $Q(1) = 1/(1-\beta)$. Normalization separates this secret multiplier $Q(1)$ and merges it into the learning rate η. This separation clearly distinguishes the role of the optimizer from that of the lr scheduler: optimizers reshape the parameter updates, while learning rates scale them. Then the plot coincidence confirms that the sharpness drop in this low-LR regime is independent of the optimizer's temporal structure, hinting a material property of the model-loss complex.

Overlapping traces in τ coordinate. In Figure 3, we trace the geometric responses (L, S) *per run* by connecting their intermediate values from each run as a line onto the same plot. It is equally interesting to see that these traces also overlap in the training progress $\tau = \eta Q(1)t$ coordinate. This graph coincidence means that the training dynamics is insensitive to the optimizer type in this regime, *effectively following the continuous-time gradient flow* $\mathrm{d}\boldsymbol{\theta}/\mathrm{d}\tau = -\boldsymbol{g}$. Optimizers in this regime only affect through its dc gain $Q(1)$ scaling the timescale τ. Additional evidences such as almost perfectly aligned gradients $c_t \approx 1$ in Figure 1d, negligible higher order aliasing $\rho_t \ll 1\%$ in Figure 1e, and root loci parked in their initial positions in Figure 1c further confirm this.

Sharpness softening. As Figures 1a, 2a and 2b show, the sharpness S decreases from its initial value $S_0 = \lambda_{\max}(\boldsymbol{H}(\boldsymbol{\theta}_0))$ until it reaches some minimum value $S_{\min}$ at $h_{\min}$. This *softening* at low effective lr h appears to be a common feature across different architectures, distinguishing the low-LR regime from other regimes with higher effective lr h, characterizing the low-LR regime as a



(a) h vs. S for GF30. (b) Low-LR t vs. S. (c) Mid-LR t vs. S. (d) High-LR t vs. S.

Figure 4: **Progress sharpening occurrence depends on the effective learning rate h.** We plot traces of the sharpness S over the *true training iterations* t (not τ this time!) using a single exemplar case of a 30-degree escape Grokfast optimizer (GF30). Progress sharpening only occurs when the stabilized sharpness is larger than the initial sharpness $S/S_0 > 1$, condition of which is governed by the effective learning rate h.

sharpness softening regime. Overlapping $S(\tau)$ traces in Figure 3 further identify the sharpness drop $F := \mathrm{d}S/\mathrm{d}\tau < 0$ as a material property in this regime. We call F the *sharpening speed*.

Shallow loss drop. For completeness, we inspect the loss L vs h plot in Figure 1b and Figures 2b and 2d. Consistently, in all the three curves, the loss does not change much from the initial value until the sharpness turns around at $h_{\min}$ to start increasing.

4 Regime II: Mid-LR Sharpening

Past the minimum stabilized sharpness $S(h_{\min}) = S_{\min}$, the training dynamics enters the mid-LR regime, which possesses several distinct characteristics than the low-LR regime. This regime is mainly characterized by (1) *progressive sharpening* of S (Cohen et al., 2021) and (2) rapid loss drop L. Also, the mid-LR regime is a *transient* regime: it bridges between the material intrinsic continuous-time gradient flow in the low-LR regime and the optimizer-dependent edge of stability in the high-LR regime. We see this (3) emergence of optimizer-dependence in multiple signatures.

Progressive sharpening. The most salient feature of the mid-LR regime is the sharpening. This can be observed in two different perspectives: First, the final stabilized sharpness $S(h)$ at the end of training with respect to the effective step size h becomes higher than the initial sharpness $S(h) > S_0$, as Figures 1 and 2 show. Second, in a single training run, sharpening happens after the sharpness hits the minimum stabilized sharpness $S_{\min}$ as shown in Figure 3b and in Figure 4c. Then the effect manifests as a *progressive sharpening* $F = \mathrm{d}S/\mathrm{d}\tau > 0$: the model sharpens itself along the gradient flow. We can make several observations about this effect: First, the sharpening first happens before the edge regime, where the geometry responses are material properties, rather than optimizer-dependent. Second, the sharpening does not always take place, as the previous works often presume (Cohen et al., 2021; Damian et al., 2023). Rather, it is only observed when the final stabilized sharpness $S(h)$ is higher than the initial sharpness S_0. Such condition is determined by the optimizer filter Q and the effective lr h. For example, in Figure 4d, we do not observe progressive sharpening when the learning rate is high enough to make the sharpness stabilized below the initial value S_0. Third, it is shown in Figure 3c that there exist a certain *material intrinsic upper envelope of the sharpness curve* $S(\tau)$ with respect to the training progress $\tau = \eta Q(1)t$ that upper-bounds the sharpness evolution. This envelope is dependent on the model, but is independent of the optimizer filter Q. Overall, the progressive sharpening effect is a *material property* of the loss landscape, but it is only realized when the optimizer and the learning rate *jointly approve* the sharpness $S(h)$ to be stabilized at higher values than the initial sharpness S_0.

Accelerated loss drop. The loss drop curve with respect to the training progress time $\tau = \eta Q(1)t$ in Figure 3e shows that the loss drop begins to speed up. This can also be observed in Figures 1b, 2b, and 2d, where the final loss traces become steeper and steeper as the effective lr increases. The trajectories from different optimizers start to deviate from each other, but this deviation is still small in this regime, limiting the role of the optimizer family in this mid-LR regime.

Emergence of optimizer-dependence. The mid-LR regime connects the optimizer-agnostic low-LR regime and the optimizer-dependent edge regime. The emergence of optimizer-dependence happens in multiple signatures: (1) aliasing effect $\rho(h)$ consistently increases according to h and becomes significant, (2) consecutive gradients become less aligned, changing its cosine c_t from 1 to the lowest value, and (3) the root locus traverses from the initial pole position to the escape of the unit circle. Escaping from the unit circle marks the entrance into the edge regime.

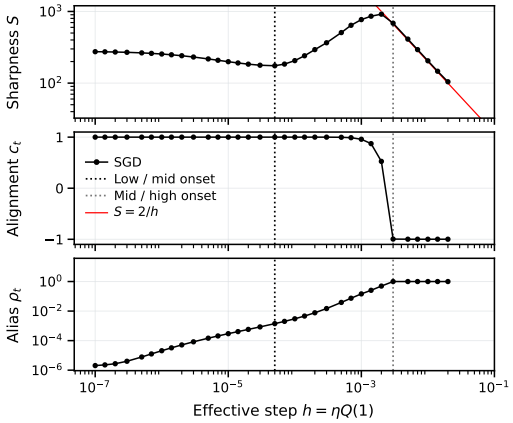


Figure 5: **A journey to the edge.** A subset of Figure 1 showing the SGD curves.

We extract the SGD subset of Figure 5 from Figure 1 to visualize what is happening during this transition more clearly. In the mid-LR regime, the first order alias $\rho(h)$ increases from less than 1% gradually to 100% (depending on the optimizer family) along the effective lr h. Higher order terms of the Taylor series expansion of the gradient descent step $\mathrm{d}\boldsymbol{\theta}/\mathrm{d}\tau - \Delta\boldsymbol{\theta}/h = O(h)$ become dominant, making the continuous-time gradient flow no longer a good approximation of the training dynamics. Pure discrete-time effects take place. The consecutive gradient pairs begin to misalign. The cosine c_t decreases from 1. As we prove in Proposition B.5, the first non-zero derivative term of the Taylor series expansion of the cosine c_t is $O(h^2)$, thereby making the alignment c_t as a relatively lagging indicator than the alias ρ_t of this approach to the edge.

At the same time, the optimizer pole traversal in the root loci becomes salient. In the low-LR regime, the poles remain in the vicinity of the original position. As the effective lr h and the sharpness S increase, so does the characteristic gain $k = hS$. The poles traverse across the loci according to the characteristic equation (♣) and towards the prescribed unit circle escape locations. Figure 1c shows how this happens for different optimizers. It is noteworthy that the escape location on the unit circle only depends on the optimizer shape $\bar{Q}(z)$ and independent of the learning rate η, the gain $Q(1)$, the model and other components of the learning system. We now reach the edge of stability.

5 Regime III: High-LR Edge

In the past two regimes, we find that the model geometry changes mostly intrinsically, independently of the optimizer Q. However, we now find that the optimizer Q starts to play a role in determining the training dynamics in this high-LR regime. Specifically, the unique shape of each optimizer's root locus determines the escape location on the unit circle, the gain at that location determines the position of the edge, and the effective time quantum $h = \eta Q(1)$ determines the sharpness S according to that edge. We elaborate on these findings in the following.

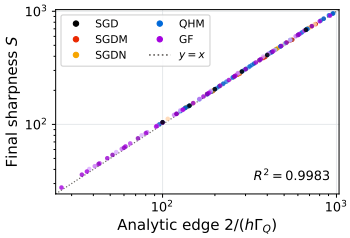


Figure 6: **The edge of stability.** Eq. (♠) holds for all optimizers and lrs.

Root locus prescribes the unit circle escape. Figure 1c draws the root loci of the 10 optimizers used throughout this paper, each of which have unique shape of the loci (the curve). Despite the different shapes, all the loci shares same three-regime behaviors. The loci start from the initial pole positions ($\times$). During the low-LR regime, the loci remains relatively at fixed point near the initial point until the minimum sharpness is reached ($\bigtriangledown$). Then, as mentioned in Section 4, the loci traverses from this point ($\bigtriangledown$) to the edge entrance ($\square$), which happens at the unit circle escape. In the high-LR regime, the loci stays at that point ($\square$) until the training breaks down due to unstably high learning rates. For one-pole optimizers, this escape point is at the Nyquist frequency ($z = -1$), i.e., the highest frequency indicating a period-2 oscillation. However, more exotic optimizers with more than one pole, such as Grokfast (Lee et al., 2024), can escape at other angles. The gain of the optimizer at this escape point Γ is solely determined by the normalized kernel of the optimizer filter $Q(z)/Q(1)$ (Lee & Lee, 2026):

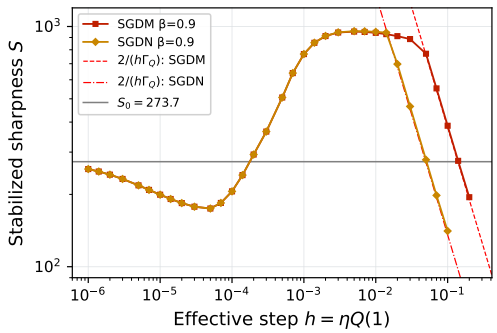


Figure 7: **The role of optimizer.** The final sharpness S of SGD with Heavy Ball and with Nesterov's momentum having the same hyperparameter $\beta = 0.9$.

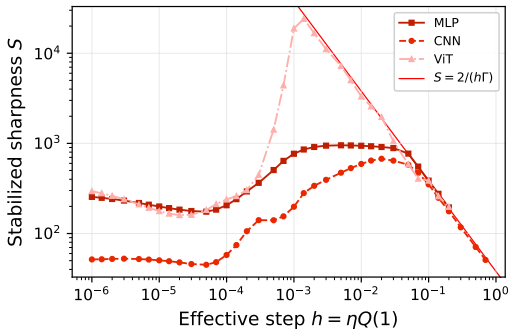


Figure 8: **Model chooses the edge entrance, the edge chooses the model sharpness.** Models sharing an SGDM ($\beta = 0.9$) optimizer share the same edge.

$$\Gamma(Q) = \max_{\omega \in \Omega_+(Q)} \left| \frac{2Q(e^{i\omega})}{(1-e^{i\omega})Q(1)} \right|, \quad \text{where } \Omega_+(Q) := \left\{ \omega \in (0,\pi] : \frac{1-e^{i\omega}}{Q(e^{i\omega})} \in \mathbb{R}_{>0} \right\}. \tag{5}$$

Escape gain determines the edge. Figure 6 shows the strict reciprocal relationship (♠) between the final sharpness S and the effective lr h, mediated by the root locus escape gain Γ in this high-LR regime. The stabilized sharpness S can be accurately predicted by the predetermined, optimizer-specific escape gain Γ, the dc gain $Q(1)$, and the learning rate η by $\tilde{S} = 2/(\eta Q(1)\Gamma)$. Figure 7 further shows the significance of this escape gain in determining the edge. SGD with Heavy Ball and with Nesterov's momentum having the same hyperparameter $\beta = 0.9$ share the same dc gain $Q(1) = 1/(1-\beta)$, resulting in almost identical sharpness traces until the edge. However, their escape gains Γ determined by the optimizer root loci are different, resulting in different entrance point into the edge. Hence the different sharpness only in the high-LR regime.

Effective lr determines the sharpness along the edge. Figure 8 shows the stabilized sharpness traces of three different models, MLP, CNN, and ViT, trained with the same SGDM ($\beta = 0.9$) optimizer, drawn with respect to the effective learning rate h. Different models have different *material* sharpness responses in the low- and mid-LR regimes, resulting in different entrance points into the edge of stability. However, since they share the same optimizer, they all share the same edge of stability. Once the learning system enters the edge regime, the learning rate h takes full charge of determining the sharpness S along the edge, regardless of the model architecture, until the training collapses due to high learning rate. And this concludes our journey to the edge of stability.

6 Summary and Conclusion

We have explored the behavior of a deep learning system under various first order optimization methods. To our surprise, we found that in the lower learning rates before the edge of stability is reached, the traces of the geometry evolution (L, S) are insensitive to the specific optimization algorithm in use. The optimizer plays a role in the higher learning rates to steer the learning system into the edge of stability. Only then, according to the EoS reciprocal rule, the sharpness S becomes predictable from the effective learning rate $h = \eta Q(1)$ until the training goes unstable.

The shared sharpness curve $S(h)$ separates the *material property of the learning system* from the *optimizer-induced edge of stability*. The evolution of the learning system below the edge of stability is largely prescribed by the model before we define what optimizer to use, but the edge is placed by the optimizer. The optimizer's role is separated into different mechanistic parts. The dc gain $Q(1)$ controls the time scale along with the learning rate η. Placing the edge of stability is done by the root locus escape gain Γ of the remaining normalized filter $Q(z)/Q(1)$, independently of the learning rate and the model being trained. Therefore, our observations suggest not only to separate the *gain* $\eta Q(1)$ from the *shaping* $Q(z)/Q(1)$ in the description of the first order optimization algorithm, but also to distinguish the problem-specific, *material* properties from the optimizer-induced, *edge of stability*. It is worth noting that the most effective learning happens at the edge of stability, where the optimizer, the model, the loss, and the data all play their roles in a delicate balance.

AI USE STATEMENT

In this work, we used AI assistants solely to assist with writing code and in assisting with mathematical proofs. All mathematical claims or proofs augmented with AI assistance were thoroughly reviewed, verified, and rewritten as needed by the human authors to ensure correctness and rigor. We take full responsibility for the final content of this work, including all text, code, and mathematical results.

REPRODUCIBILITY STATEMENT

We have made every effort to ensure that our results are reproducible. Sufficient explanations of our methods and setups are provided in Section 2 and Section 3. All figures presented in the paper can be fully reproduced using the code and data included in the supplementary material.

REFERENCES

Atish Agarwala, Fabian Pedregosa, and Jeffrey Pennington. Second-order regression models exhibit progressive sharpening to the edge of stability. In *ICML*, volume 202, pp. 169–195, 2023.

Kwangjun Ahn, Jingzhao Zhang, and Suvrit Sra. Understanding the unstable convergence of gradient descent. In *ICML*, volume 162, 2022.

Arseniy Andreyev and Pierfrancesco Beneventano. Edge of stochastic stability: Revisiting the edge of stability for SGD. *arXiv preprint arXiv:2412.20553*, 2024.

Arseniy Andreyev, Advikar Ananthkumar, Marc Walden, Tomaso Poggio, and Pierfrancesco Beneventano. Momentum further constrains sharpness at the edge of stochastic stability. In *ICML*, 2026.

Sanjeev Arora, Zhiyuan Li, and Abhishek Panigrahi. Understanding gradient descent on the edge of stability in deep learning. In *ICML*, volume 162, pp. 948–1024, 2022.

David G. T. Barrett and Benoit Dherin. Implicit gradient regularization. In *ICLR*, 2021.

Dimitri P Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 2nd edition, 1999.

Lei Chen and Joan Bruna. Beyond the edge of stability via two-step gradient updates. In *ICML*, volume 202, pp. 4330–4391, 2023.

Xuxing Chen, Krishnakumar Balasubramanian, Promit Ghosal, and Bhavya Agrawalla. From stability to chaos: Analyzing gradient descent dynamics in quadratic regression. *Transactions on Machine Learning Research*, 2024.

Jeremy M. Cohen, Simran Kaur, Yuanzhi Li, J. Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. In *ICLR*, 2021.

Jeremy M. Cohen, Behrooz Ghorbani, Shankar Krishnan, Naman Agarwal, Sourabh Medapati, Michal Badura, Daniel Suo, David Cardoze, Zachary Nado, George E. Dahl, and Justin Gilmer. Adaptive gradient methods at the edge of stability. In *NeurIPS 2023 Workshop on Heavy Tails in Machine Learning: Structure, Stability, and Dynamics*, 2023.

Jeremy M. Cohen, Alex Damian, Ameet Talwalkar, J. Zico Kolter, and Jason D. Lee. Understanding optimization in deep learning with central flows. In *ICLR*, 2025.

Alex Damian, Eshaan Nichani, and Jason D. Lee. Self-stabilization: The implicit bias of gradient descent at the edge of stability. In *ICLR*, 2023.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12(61):2121–2159, 2011.

Justin Gilmer, Behrooz Ghorbani, Ankush Garg, Sneha Kudugunta, Behnam Neyshabur, David Cardoze, George Edward Dahl, Zachary Nado, and Orhan Firat. A loss curvature perspective on training instabilities of deep learning models. In *ICLR*, 2022.

Igor Gitman, Hunter Lang, Pengchuan Zhang, and Lin Xiao. Understanding the role of momentum in stochastic gradient methods. In *NeurIPS*, 2019.

Thomas Hofmann. The map behind the flow: Finite-step gradient descent as a dynamical system. *arXiv preprint arXiv:2607.04993*, 2026.

Rustem Islamov, Michael Crawshaw, Jeremy Cohen, and Robert Gower. Non-euclidean gradient descent operates at the edge of stability. In *ICML*, 2026.

Gaurav Iyer, Boris Hanin, and David Rolnick. Maximal initial learning rates in deep relu networks. In *ICML*, Proceedings of Machine Learning Research, 2023.

Stanisław Jastrzębski, Zachary Kenton, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. On the relation between the sharpest directions of DNN loss and the SGD step length. In *ICLR*, 2019.

Dayal Singh Kalra and Maissam Barkeshli. Phase diagram of early training dynamics in deep neural networks: Effect of the learning rate, depth, and width. In *NeurIPS*, 2023.

Dayal Singh Kalra and Maissam Barkeshli. Why warmup the learning rate? underlying mechanisms and improvements. In *NeurIPS*, 2024.

Dayal Singh Kalra, Tianyu He, and Maissam Barkeshli. Universal sharpness dynamics in neural network training: Fixed point analysis, edge of stability, and route to chaos. In *ICLR*, 2025.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Jaerin Lee and Kyoung Mu Lee. Greedy alignment principle for optimizer selection. *arXiv preprint arXiv:2512.06370*, 2025.

Jaerin Lee and Kyoung Mu Lee. The road taken: The role of optimizers at the edge of stability. *arXiv preprint arXiv:2608.18415*, 2026.

Jaerin Lee, Bong Gyun Kang, Kihoon Kim, and Kyoung Mu Lee. Grokfast: Accelerated grokking by amplifying slow gradients. *arXiv preprint arXiv:2405.20233*, 2024.

Sungyoon Lee and Cheongjae Jang. A new characterization of the edge of stability based on a sharpness measure aware of batch gradient distribution. In *ICLR*, 2023.

Laurent Lessard, Benjamin Recht, and Andrew Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1):57–95, 2016. doi: 10.1137/15M1009597.

Aitor Lewkowycz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. The large learning rate phase of deep learning: the catapult mechanism. *arXiv preprint arXiv:2003.02218*, 2020.

Qianxiao Li, Cheng Tai, and Weinan E. Stochastic modified equations and adaptive stochastic gradient algorithms. In *ICML*, Proceedings of Machine Learning Research, 2017.

Qianxiao Li, Cheng Tai, and Weinan E. Stochastic modified equations and dynamics of stochastic gradient algorithms i: Mathematical foundations. *Journal of Machine Learning Research*, 20(40): 1–47, 2019a.

Yuanzhi Li, Colin Wei, and Tengyu Ma. Towards explaining the regularization effect of initial large learning rate in training neural networks. In *NeurIPS*, 2019b.

Zhiyuan Li, Kaifeng Lyu, and Sanjeev Arora. Reconciling modern deep learning with traditional optimization analyses: The intrinsic learning rate. In *NeurIPS*, 2020.

Elon Litman. The origin of edge of stability. *arXiv preprint arXiv:2604.20446*, 2026.

James Lucas, Shengyang Sun, Richard Zemel, and Roger Grosse. Aggregated momentum: Stability through passive damping. In *ICLR*, 2019.

Kaifeng Lyu, Zhiyuan Li, and Sanjeev Arora. Understanding the generalization benefit of normalization layers: Sharpness reduction. In *NeurIPS*, 2022.

Jerry Ma and Denis Yarats. Quasi-hyperbolic momentum and adam for deep learning. *ICLR*, 2019.

Aaron Mishkin, Ahmed Khaled, Yuanhao Wang, Aaron Defazio, and Robert M. Gower. Directional smoothness and gradient methods: Convergence and adaptivity. In *NeurIPS*, 2024.

Taiki Miyagawa. Toward equation of motion for deep neural networks: Continuous-time gradient descent and discretization error analysis. In *NeurIPS*, 2022. doi: 10.52202/068431-2738.

Michael Muehlebach and Michael I. Jordan. Optimization with momentum: Dynamical, control-theoretic, and symplectic perspectives. *Journal of Machine Learning Research*, 22(73):1–50, 2021.

Yurii E. Nesterov. A method for solving the convex programming problem with convergence rate $o(1/k^2)$. *Doklady Akademii Nauk SSSR*, 269(3):543–547, 1983. English translation: Soviet Mathematics Doklady, 27(2):372–376, 1983.

Lorenzo Noci, Alexandru Meterez, Thomas Hofmann, and Antonio Orvieto. Super consistency of neural network landscapes and learning rate transfer. In *NeurIPS*, 2024.

Katsuhiko Ogata. *Discrete-Time Control Systems*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition, 1995.

James M Ortega and Werner C Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York-London, 1970.

Matteo Pagliardini, Pierre Ablin, and David Grangier. The AdEMAMix optimizer: Better, faster, older. In *ICLR*, 2025.

Boris T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3):400–407, 1951.

Leslie N. Smith. Cyclical learning rates for training neural networks. In *WACV*. IEEE, 2017. doi: 10.1109/WACV.2017.58.

Samuel L. Smith and Quoc V. Le. A bayesian perspective on generalization and stochastic gradient descent. In *ICLR*, 2018.

Samuel L. Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V. Le. Don't decay the learning rate, increase the batch size. In *ICLR*, 2018.

Samuel L. Smith, Benoit Dherin, David G. T. Barrett, and Soham De. On the origin of implicit regularization in stochastic gradient descent. In *ICLR*, 2021.

Minhak Song and Chulhee Yun. Trajectory alignment: Understanding the edge of stability phenomenon via bifurcation theory. In *NeurIPS*, 2023.

Weijie Su, Stephen Boyd, and Emmanuel J. Candès. A differential equation for modeling nesterov's accelerated gradient method: Theory and insights. *Journal of Machine Learning Research*, 17 (153):1–43, 2016.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *ICML*, Proceedings of Machine Learning Research, 2013.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5—-RMSProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

Zixuan Wang, Zhouzi Li, and Jian Li. Analyzing sharpness along GD trajectory: Progressive sharpening and edge of stability. In *NeurIPS*, 2022.

Xingyu Zhu, Zixuan Wang, Xiang Wang, Mo Zhou, and Rong Ge. Understanding edge-of-stability training dynamics with a minimalist example. In *ICLR*, 2023.

A Related Work

Edge of stability. Performing a gradient descent according to a quadratic loss function $L(\theta) = \frac{1}{2}\theta^\top \boldsymbol{H}\theta$ has a maximum learning rate $\eta_{\max} = 2/\lambda_{\max}(\boldsymbol{H})$ beyond which the learning starts to diverge. On the contrary, the edge of stability (EoS) (Cohen et al., 2021) states that in a large-scale deep learning, this boundary is often violated without instability. Instead, it is observed in multiple works (Lyu et al., 2022; Cohen et al., 2023; Damian et al., 2023; Agarwala et al., 2023; Zhu et al., 2023; Andreyev & Beneventano, 2024; Chen et al., 2024; Cohen et al., 2025; Islamov et al., 2026; Andreyev et al., 2026; Litman, 2026; Lee & Lee, 2026) that the sharpness of the loss landscape $S := \lambda_{\max}(\boldsymbol{H})$ hovers around this value during deep learning practice. As a relatively new concept, the governing equation of the edge of stability has been proposed in multiple versions, each of which is based on different scenarios.

- Original formulation reported for full-batch gradient descent (Cohen et al., 2021):
$$S \;:=\; \lambda_{\max}(\nabla^2_{\boldsymbol{\theta}} L(\boldsymbol{\theta})). \tag{6}$$
- Preconditioner $\boldsymbol{P}$ updates the eigenspectrum defining sharpness (Cohen et al., 2023):
$$S \;:=\; \lambda_{\max}(\boldsymbol{P}^{-1/2}\nabla^2_{\boldsymbol{\theta}} L(\boldsymbol{\theta})\boldsymbol{P}^{-1/2}). \tag{7}$$
- Momentum β multiplies a factor $\Gamma(\beta)$ on the left side (Cohen et al., 2023):
$$S \;\leq\; \frac{2}{\eta\Gamma(\beta)}. \tag{8}$$
- Stochastic regime changes the definition of sharpness S to Batch Sharpness (Andreyev & Beneventano, 2024), which is a batch-averaged directional curvature (Lee & Jang, 2023; Mishkin et al., 2024) of the loss:
$$S_{\mathrm{B}} \;:=\; \mathbb{E}_{\mathcal{B}\sim\mathcal{P}_b}\left[\frac{\nabla_{\boldsymbol{\theta}} L_{\mathcal{B}}(\boldsymbol{\theta})^\top \nabla^2_{\boldsymbol{\theta}} L_{\mathcal{B}}(\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} L_{\mathcal{B}}(\boldsymbol{\theta})}{\|\nabla_{\boldsymbol{\theta}} L_{\mathcal{B}}(\boldsymbol{\theta})\|^2}\right] \leq \frac{2}{\eta}, \tag{9}$$
where $\mathcal{P}_b$ is the distribution of mini-batches $\mathcal{B}$ and $L_{\mathcal{B}}(\boldsymbol{\theta})$ is the loss from a mini-batch $\mathcal{B}$.
- Mini-batch size modulates the momentum-induced factor $\Gamma(\beta, |\mathcal{B}|)$ (Andreyev et al., 2026):
$$S_{\mathrm{B}} \;\leq\; \frac{2}{\eta\Gamma(\beta, |\mathcal{B}|)}. \tag{10}$$

Overall, the governing inequality has continuously been evolving based on empirical studies. In this paper, we fix the batch size in the large batch regime specified by Andreyev et al. (2026) and focus on the unpreconditioned maximum Hessian eigenvalue $S = \lambda_{\max}(\boldsymbol{H})$ as the sharpness of the loss landscape, and empirically show that the edge of stability with respect to this sharpness is governed by equation (♠) with a correction term $\Gamma(Q)$ that is the root locus escape gain of the optimizer filter Q (Lee & Lee, 2026).

Driving mechanism of the edge of stability and progressive sharpening. Multiple works have been searching for the mechanistic insight into the edge of stability. Related phenomena of progressive sharpening (Cohen et al., 2021; Damian et al., 2023), training instabilities (Gilmer et al., 2022), and structured dynamics beyond the quadratic stability boundary are reported to be connected to the edge of stability (Jastrzębski et al., 2019; Wang et al., 2022; Ahn et al., 2022; Arora et al., 2022). To understand the driving mechanism, self-stabilization by higher-order geometry, simplified regression and network models, period-2 oscillation and two-step stability, bifurcation dynamics, and continuous reductions of edge motion (Damian et al., 2023; Agarwala et al., 2023; Zhu et al., 2023; Chen & Bruna, 2023; Song & Yun, 2023; Cohen et al., 2025; Hofmann, 2026) have been proposed. Phase-diagram analyses show that the leanring rate, the width of the model, the initialization, and the warmpup algorithm participate in sharpness reduction, catapult (Lewkowycz et al., 2020) mechanism, progressive sharpening, and edge of stability into reproducible regimes (Kalra & Barkeshli, 2023; 2024; Kalra et al., 2025). Our contribution is complementary to these works. We study how the model geometry evolves across diverse set of optimizers and learning rates and uncover a three-regime behavior starting from a shared pre-edge material response to traversal mechanism in the mid-LR regime, and finally to an optimizer-specific root locus escape. Noteably, previous works (Cohen et al., 2021; Damian et al., 2023) have presumed the existence of progressive sharpening that transports the model onto the edge of stability without explicitly defining the mechanism. From a comprehensive study, we show that the progressive sharpening is a model's material property that happens only when the stabilized sharpness is greater than the initial sharpness, and when the effective learning rate allows the sharpness to increase.

First order methods and momentum. Classical Heavy Ball (Polyak, 1964; Sutskever et al., 2013) and Nesterov's Accelerated Gradient (Nesterov, 1983; Sutskever et al., 2013) introduce memory into first-order optimization (Robbins & Monro, 1951). Optimizer with states have been studied in the context of control theory (Su et al., 2016; Lessard et al., 2016; Gitman et al., 2019; Muehlebach & Jordan, 2021; Lee et al., 2024; Lee & Lee, 2026). In this context more advanced form of state is proposed, such as quasi-hyperbolic momentum (Ma & Yarats, 2019), AggMo (Lucas et al., 2019), Grokfast (Lee et al., 2024), and AdEMAMix (Pagliardini et al., 2025), that extend the conventional momentum-based optimizers by allowing more than one pole or zeros. We also treat an unpreconditioned first-order method as a causal gradient filter Q and study its transfer function. Our observation further suggests to separate its scalar dc gain $Q(1)$ that is absorbed into the effective learning rate $h = \eta Q(1)$ from the normalized temporal shape $\bar{Q}(z) = Q(z)/Q(1)$.

Learning-rate sweeps and finite-step theory. From the decades of practice in deep learning, learning rate has been established as a primary control variable in deep learning (Smith, 2017), and scaling-rule and noise-scale studies linked learning rate to batch size and stochastic dynamics (Smith et al., 2018; Smith & Le, 2018). It has been shown that the learning rate can control more than just a time scale of the optimizaiton process. Large initial learning rates can alter the order of feature acquisition and implicit bias (Li et al., 2019b), and the maximal usable learning rate can differ between initialization and later training (Iyer et al., 2023). Distinction between the discrete-time gradient descent and the continuous-time gradient flow has been formalized by backward-error and modified-equation analyses (Li et al., 2017; 2019a; Barrett & Dherin, 2021; Smith et al., 2021; Miyagawa, 2022). Recent work also shows that the landscape statistics and transferable learning rates can exhibit architecture-dependent yet highly structured scaling (Li et al., 2020; Noci et al., 2024). Most of the works focus on the learning rate selection and schedule design, whereas we use a dense fixed-task sweep to study the governing dynamics of the *learning system*. We find that the dc-adjusted step $h = \eta Q(1)$ indexes the common loss and sharpness responses in the pre-edge regimes, whereas the filter shape $\bar{Q}(z)$ determines the entry point and the trajectory to the edge.

B Some Mathematics behind the Transition to the Edge

This section discusses the mathematics behind the transition from the pre-edge gradient flow dynamics to the edge of stability. In specific, we provide Taylor expansions of the filtered gradient descent that connects the various measurements, i.e., the sharpness S, the loss L, the alignment c_t, and the alias ρ_t, from the main text into the same dynamics. All expansions below are local small h expansions for a fixed smooth objective and a fixed stable filter at an anchored iteration t.

Basic notations. Let $L(\theta)$ be a scalar objective function and $Q(z)$ be a causal gradient filter with an impulse response q_j likewise in the main text. A learning rate $\eta > 0$ is a *sampling period* or a *time scale* that realizes a continuous-time gradient flow into a discrete-time gradient descent. Define $\boldsymbol{g} := \nabla L(\theta)$ and $\boldsymbol{H} := \nabla^2 L(\theta)$. Let $\boldsymbol{g}_{\leq t}$ be a causal sequence of gradients up to *training iteration index* t. Filtered gradient descent updates the parameter $\boldsymbol{\theta}$ as:

$$\boldsymbol{\theta}_{t+1} \;=\; \boldsymbol{\theta}_t - \eta (Q * \boldsymbol{g}_{\leq t})_t, \qquad (Q * \boldsymbol{g}_{\leq t})_t \;=\; \sum_{j \geq 0} q_j \boldsymbol{g}_{t-j}. \tag{11}$$

$Q(1) = q_0 + q_1 + q_2 + \cdots \in (0, \infty)$ is the dc gain of the filter Q. Using the *effective learning rate* $h := \eta Q(1)$, we can rewrite the update into a normalized form as:

$$\boldsymbol{\theta}_{t+1} \;=\; \boldsymbol{\theta}_t - h (\bar{Q} * \boldsymbol{g}_{\leq t})_t, \qquad \bar{Q}(z) \;=\; \frac{Q(z)}{Q(1)}, \tag{12}$$

where $\bar{Q}(1) = 1$. We additionally define a *canonical training progress* $\tau = ht$ that denotes the time index in the continuous-time gradient flow. Unless otherwise specified, we simply call τ the *time*.

Flow, descent, and aliasing. Consider a continuous-time gradient flow as a reference dynamics

$$\frac{\mathrm{d}\boldsymbol{\theta}}{\mathrm{d}\tau} \;=\; -\boldsymbol{g}. \tag{13}$$

This corresponds to the discrete-time update with a time step h. We can *sample* from this continuous-time dynamics to obtain an equivalent discrete-time gradient descent algorithm in the parameter space. However, due to the discretization, we have an *aliasing term* that signifies the difference between the ideal flow and the acutal learning dynamics. We can evaluate the corresponding discrete-time gradient descent step in the parameter space from the Taylor expansion of the continuous-time dynamics of equation (13).

$$\boldsymbol{\theta}(\tau + h) \;=\; \boldsymbol{\theta}(\tau) - h\boldsymbol{g} + \frac{h^2}{2}\boldsymbol{H}\boldsymbol{g} + O(h^3). \tag{14}$$

Memoryless gradient descent takes $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - h\boldsymbol{g}_t$ instead of the exact gradient-flow segment of equation (14), and therefore differs from the exact gradient-flow segment by

$$\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}(\tau + h) \;=\; -\frac{h^2}{2}\boldsymbol{H}\boldsymbol{g}_t + O(h^3). \tag{15}$$

This is the main source of the trajectory bias relative to gradient flow.

Gradient gradient flow. We can also think of a *gradient flow of the gradients* that happens along the *gradient flow of the parameters*. From the chain rule, we have

$$\frac{\mathrm{d}\boldsymbol{g}}{\mathrm{d}\tau} \;=\; \frac{\mathrm{d}}{\mathrm{d}\tau}\left(\nabla L(\boldsymbol{\theta})\right) \;=\; \nabla^2 L(\boldsymbol{\theta})\frac{\mathrm{d}\boldsymbol{\theta}}{\mathrm{d}\tau} \;=\; -\nabla^2 L(\boldsymbol{\theta})\boldsymbol{g} \;=\; -\boldsymbol{H}\boldsymbol{g}. \tag{16}$$

In other words, the continuous-time gradient flow not only transfers the parameter, but also moves the gradient itself. The Hessian $\boldsymbol{H}$ is the *rate of change* of the gradient along the first order ODE. From this, we can derive the *acceleration* of the parameter along the gradient flow as well by

$$\frac{\mathrm{d}^2\boldsymbol{\theta}}{\mathrm{d}\tau^2} \;=\; -\frac{\mathrm{d}\boldsymbol{g}}{\mathrm{d}\tau} \;=\; \boldsymbol{H}\boldsymbol{g}. \tag{17}$$

Note that this is automatically given from equation (13) by the chain rule. Likewise, any order derivative of the parameter along the flow is automatically given. If we take another derivative from equation (16), we introduce the third order term

$$\frac{\mathrm{d}^2\boldsymbol{g}}{\mathrm{d}\tau^2} \;=\; -\frac{\mathrm{d}\boldsymbol{H}}{\mathrm{d}\tau}\boldsymbol{g} - \boldsymbol{H}\frac{\mathrm{d}\boldsymbol{g}}{\mathrm{d}\tau}. \tag{18}$$

The Hessian $\boldsymbol{H}$ moves along the gradient flow as well, giving us

$$\frac{\mathrm{d}\boldsymbol{H}}{\mathrm{d}\tau} \;=\; \nabla^3 L\left[\frac{\mathrm{d}\boldsymbol{\theta}}{\mathrm{d}\tau}\right] \;=\; -\nabla^3 L[\boldsymbol{g}]. \tag{19}$$

Then

$$\frac{\mathrm{d}^2\boldsymbol{g}}{\mathrm{d}\tau^2} \;=\; \boldsymbol{H}^2\boldsymbol{g} + \nabla^3 L[\boldsymbol{g},\boldsymbol{g}] \;=\; -\frac{\mathrm{d}^3\boldsymbol{\theta}}{\mathrm{d}\tau^3}. \tag{20}$$

This gives us the Taylor expansion of the gradient gradient flow up to the second order term

$$\boldsymbol{g}(\tau+h) \;=\; \boldsymbol{g}(\tau) - h\boldsymbol{H}(\tau)\boldsymbol{g}(\tau) + \frac{h^2}{2}\big(\boldsymbol{H}^2(\tau)\boldsymbol{g}(\tau) + \nabla^3 L(\tau)[\boldsymbol{g}(\tau),\boldsymbol{g}(\tau)]\big) + O(h^3). \tag{21}$$

Plugging $-jh$ instead of h into the above equation, and let $\boldsymbol{g}(\tau) = \boldsymbol{g}_t$ and $\boldsymbol{H}(\tau) = \boldsymbol{H}_t$, we get

$$\boldsymbol{g}(\tau - jh) \;=\; \boldsymbol{g}_t + jh\boldsymbol{H}_t\boldsymbol{g}_t + \frac{j^2h^2}{2}\big(\boldsymbol{H}_t^2\boldsymbol{g}_t + \nabla^3 L_t[\boldsymbol{g}_t,\boldsymbol{g}_t]\big) + O(h^3). \tag{22}$$

Aliasing in gradient gradient flow. Taking a simple gradient descent $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - h\boldsymbol{g}_t$, we get

$$\boldsymbol{g}_{t+1} \;=\; \nabla L(\boldsymbol{\theta}_{t+1}) \;=\; \nabla L(\boldsymbol{\theta}_t - h\boldsymbol{g}_t) \;=\; \boldsymbol{g}_t - h\boldsymbol{H}_t\boldsymbol{g}_t + \frac{h^2}{2}\nabla^3 L_t[\boldsymbol{g}_t,\boldsymbol{g}_t] + O(h^3), \tag{23}$$

directly from the Taylor expansion of $L(\boldsymbol{\theta})$ around $\boldsymbol{\theta}_t$. Meanwhile, the Taylor expansion of the *gradient gradient flow* in equation (21) just gave us

$$\boldsymbol{g}(\tau+h) \;=\; \boldsymbol{g}_t - h\boldsymbol{H}_t\boldsymbol{g}_t + \frac{h^2}{2}\big(\boldsymbol{H}_t^2\boldsymbol{g}_t + \nabla^3 L_t[\boldsymbol{g}_t,\boldsymbol{g}_t]\big) + O(h^3), \tag{24}$$

and thus

$$\boldsymbol{g}_{t+1} - \boldsymbol{g}(\tau+h) \;=\; -\frac{h^2}{2}\boldsymbol{H}_t^2\boldsymbol{g}_t + O(h^3). \tag{25}$$

Therefore, we cannot simply assume $\boldsymbol{g}_{t+j} = \boldsymbol{g}(\tau+jh)$ for any $j \geq 0$. Comparing equation (15) and equation (25), we see that the truncation error in the gradient space has one more Hessian than that of the parameter space. If we define the parameter update vector $\boldsymbol{u}_t := (\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t)/h$ and define

$$\boldsymbol{u}_{t:t+j} \;:=\; \frac{\boldsymbol{\theta}_{t+j} - \boldsymbol{\theta}_t}{h} \;=\; \sum_{i=t}^{t+j-1} \boldsymbol{u}_i, \tag{26}$$

then we have the exact Taylor expansion of the gradient update $\boldsymbol{g}_{t+j}$ as

$$\boldsymbol{g}_{t+j} \;=\; \boldsymbol{g}_t - h\boldsymbol{H}_t\boldsymbol{u}_{t:t+j} + \frac{h^2}{2}\nabla^3 L_t[\boldsymbol{u}_{t:t+j},\boldsymbol{u}_{t:t+j}] + O(h^3\|\boldsymbol{u}_{t:t+j}\|^3). \tag{27}$$

However, this formulation complicates the analysis. The problem is that $\boldsymbol{u}_{t:t+j}$ is not a simple vector, but an aggregation of the sequence of parameter updates, which incorporates changes in mini-batch, evolution of local geometry along the algorithmic trace, filtering effect of the optimizer, and so on.

Quasi-dc expansion and filter effect. Fortunately, we can simplify the analysis by assuming the learning rate is sufficiently low, allowing us to use a quasi-dc expansion

$$\boldsymbol{u}_{t+i} \;=\; \boldsymbol{g}_t + O(h), \quad \text{and} \quad \boldsymbol{u}_{t:t+j} \;=\; j\boldsymbol{g}_t + O(h). \tag{28}$$

Then

$$\boldsymbol{g}_{t-j} \;=\; \boldsymbol{g}_t + jh\boldsymbol{H}_t\boldsymbol{g}_t + O(h^2). \tag{29}$$

If we plug this into the convolution of the (normalized) filter $\bar{Q} * \boldsymbol{g}_{\leq t}$, we get

$$(\bar{Q} * \boldsymbol{g}_{\leq t})_t \;=\; \sum_{j\geq 0} \bar{q}_j \boldsymbol{g}_{t-j} \tag{30}$$

$$\;=\; \sum_{j\geq 0} \bar{q}_j(\boldsymbol{g}_t + jh\boldsymbol{H}_t\boldsymbol{g}_t) + O(h^2) \tag{31}$$

$$\;=\; \Big(\sum_{j\geq 0}\bar{q}_j\Big)\boldsymbol{g}_t + \Big(\sum_{j\geq 1} j\bar{q}_j\Big)h\boldsymbol{H}_t\boldsymbol{g}_t + O(h^2) \tag{32}$$

$$\;=\; \boldsymbol{g}_t - h\frac{Q'(1)}{Q(1)}\boldsymbol{H}_t\boldsymbol{g}_t + O(h^2). \tag{33}$$

The last line follows from the fact that $Q'(1) = -\sum_{j\geq 0} jq_j = -Q(1)\sum_{j\geq 0} j\bar{q}_j$. Therefore,

$$\boldsymbol{u}_t \;=\; (\bar{Q} * \boldsymbol{g}_{\leq t})_t \;=\; \boldsymbol{g}_t - h\frac{Q'(1)}{Q(1)}\boldsymbol{H}_t\boldsymbol{g}_t + O(h^2). \tag{34}$$

This deserves a couple of theorems.

Theorem B.1 (Filtered gradient descent.)**.** *For any causal filter Q with an impulse response q_j, the filtered gradient descent updates the parameter by*

$$\boldsymbol{\theta}_{t+1} \;\leftarrow\; \boldsymbol{\theta}_t - h\boldsymbol{g}_t + h^2\frac{Q'(1)}{Q(1)}\boldsymbol{H}_t\boldsymbol{g}_t + O(h^3). \tag{35}$$

Proof. From $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - h\boldsymbol{u}_t$ and equation (34), we get the result. □

Theorem B.2 (Augmented gradient flow.)**.** *Given a causal filter Q with an impulse response q_j, the filtered gradient descent corresponds to a continuous-time gradient flow of the modified differential equation*

$$\frac{\mathrm{d}\boldsymbol{\theta}}{\mathrm{d}\tau} \;=\; -\boldsymbol{g} - h\alpha_Q\boldsymbol{H}\boldsymbol{g}, \tag{♡}$$

where

$$\alpha_Q \;:=\; \frac{1}{2} - \frac{Q'(1)}{Q(1)}, \tag{36}$$

up to $O(h^2)$.

Proof. From the Taylor expansion of equation (♡) to $O(h^2)$, we get

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - h\boldsymbol{g}_t - h^2\alpha_Q\boldsymbol{H}_t\boldsymbol{g}_t + \frac{h^2}{2}\boldsymbol{H}_t\boldsymbol{g}_t + O(h^3), \tag{37}$$

where the last term comes from expanding $\boldsymbol{g}(\boldsymbol{\theta}_t - h\boldsymbol{g}_t) \approx \boldsymbol{g}_t - h\boldsymbol{H}_t\boldsymbol{g}_t$.

Combining like h^2 terms, the update becomes

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - h\boldsymbol{g}_t + h^2\left(\frac{1}{2} - \alpha_Q\right)\boldsymbol{H}_t\boldsymbol{g}_t + O(h^3). \tag{38}$$

Comparing this to equation (35), we match coefficients and require

$$\alpha_Q = \frac{1}{2} - \frac{Q'(1)}{Q(1)}. \tag{39}$$

Thus, the filtered gradient descent update is matched by the ODE (♡) up to $O(h^2)$, with α_Q as defined in (36). □

We can likewise compare the gradient trajectories from the filtered gradient descent with that of the augmented gradient flow. By applying Theorem B.1 j times with a fixed number of steps j, we get

$$\begin{aligned}
\boldsymbol{u}_{t+i} \;&=\; (\bar{Q} * \boldsymbol{g}_{\leq t+i})_{t+i} && (40)\\
&=\; \sum_{l\geq 0}\bar{q}_l\boldsymbol{g}_{t+i-l} && (41)\\
&=\; \sum_{l\geq 0}\bar{q}_l\big(\boldsymbol{g}_t + (l-i)h\boldsymbol{H}_t\boldsymbol{g}_t\big) + O(h^2) && (42)\\
&=\; \boldsymbol{g}_t - h\left(\frac{Q'(1)}{Q(1)} + i\right)\boldsymbol{H}_t\boldsymbol{g}_t + O(h^2). && (43)
\end{aligned}$$

Summing over i from 0 to $j-1$, we get

$$\boldsymbol{u}_{t:t+j} \;=\; \sum_{i=0}^{j-1}\boldsymbol{u}_{t+i} \;=\; j\boldsymbol{g}_t - h\left(j\frac{Q'(1)}{Q(1)} + \frac{j(j-1)}{2}\right)\boldsymbol{H}_t\boldsymbol{g}_t + O(h^2). \tag{44}$$

Plugging this into equation (27), we get

$$\boldsymbol{g}_{t+j} \;=\; \boldsymbol{g}_t - jh\boldsymbol{H}_t\boldsymbol{g}_t + h^2\left[\left(j\frac{Q'(1)}{Q(1)} + \frac{j(j-1)}{2}\right)\boldsymbol{H}_t^2\boldsymbol{g}_t + \frac{j^2}{2}\nabla^3 L_t[\boldsymbol{g}_t,\boldsymbol{g}_t]\right] + O(h^3). \quad (45)$$

On the other hand, plugging jh in the exact Taylor expansion of $\boldsymbol{g}(\tau)$ gives

$$\boldsymbol{g}(\tau + jh) \;=\; \boldsymbol{g}_t - jh\boldsymbol{H}_t\boldsymbol{g}_t + \frac{j^2h^2}{2}\Big(\boldsymbol{H}_t^2\boldsymbol{g}_t + \nabla^3 L_t[\boldsymbol{g}_t,\boldsymbol{g}_t]\Big) + O(h^3). \quad (46)$$

Comparing these two equations, we see that they match up to $O(h^2)$.

Corollary B.3 (Gradient consistency.)**.** *Gradients from the continuous-time gradient flow* $\mathrm{d}\boldsymbol{g}/\mathrm{d}\tau = -\boldsymbol{H}\boldsymbol{g}$ *and those from the filtered gradient descent of equation (35) agree up to* $O(h^2)$ *following*

$$\boldsymbol{g}_{t+j} - \boldsymbol{g}(\tau + jh) \;=\; -jh^2\alpha_Q\boldsymbol{H}_t^2\boldsymbol{g}_t + O(h^3). \quad (47)$$

Proof. Subtracting equation (46) from equation (45) obtains the result. □

Local geometric responses in the low-LR regime. Once the accurate approximations for the parameter and gradient propagation are established, approximating the local geometric responses is straightforward. From the chain rule,

$$\frac{\mathrm{d}L}{\mathrm{d}\tau} \;=\; \nabla L^\top\frac{\mathrm{d}\boldsymbol{\theta}}{\mathrm{d}\tau}, \quad (48)$$

Let $S(\boldsymbol{\theta}) = \lambda_{\max}(\boldsymbol{H}(\boldsymbol{\theta}))$. Likewise,

$$\frac{\mathrm{d}S}{\mathrm{d}\tau} \;=\; \nabla S^\top\frac{\mathrm{d}\boldsymbol{\theta}}{\mathrm{d}\tau}, \quad (49)$$

where

$$\nabla S \;=\; \nabla^3 L[\boldsymbol{v}_{\max},\boldsymbol{v}_{\max}]. \quad (50)$$

Here, $\boldsymbol{v}_{\max}$ is the maximum eigenvector of the Hessian $\boldsymbol{H}$. From the observations in the main text, we see that in the low-LR regime, the geometric responses are almost identical, implying that the traces follow the same, continuout-time gradient flow in the progress coordinate $\tau = ht$. Mathematically, this corresponds to $\mathrm{d}\boldsymbol{\theta}/\mathrm{d}\tau = -\boldsymbol{g}$.

$$\frac{\mathrm{d}L}{\mathrm{d}\tau} \;=\; -\nabla L^\top\boldsymbol{g} \;=\; -\|\boldsymbol{g}\|^2, \quad (51)$$

$$\frac{\mathrm{d}S}{\mathrm{d}\tau} \;=\; -\nabla S^\top\boldsymbol{g} \;=\; -\nabla^3 L[\boldsymbol{v}_{\max},\boldsymbol{v}_{\max},\boldsymbol{g}]. \quad (52)$$

First order alias ρ_t. The first order alias ρ_t defined in Section 2 actually was a scaled normalized gradient difference,

$$\rho_t \;:=\; |\alpha_Q|\frac{\|\boldsymbol{g}_{t+1} - \boldsymbol{g}_t\|}{\|\boldsymbol{g}_t\|}. \quad (53)$$

We now prove that ρ_t *is* the *first order alias*, the magnitude of the higher order aliasing terms normalized by the norm of the ideal first order gradient flow $\dot{\boldsymbol{\theta}}_t = -\boldsymbol{g}_t$.

Proposition B.4 (First order alias.)**.** *Assume a parameter update is made by the filtered gradient descent* $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \eta(Q * \boldsymbol{g}_{\le t})_t$*. By Theorem B.2, a continuous-time gradient flow defined by*

$$\dot{\boldsymbol{\theta}}_t^{\text{equiv}} \;=\; -\boldsymbol{g}_t - h\alpha_Q\boldsymbol{H}_t\boldsymbol{g}_t, \quad (54)$$

matches this discrete-time update up to $O(h^2)$*. Then* ρ_t *in equation (53) is equal to the normalized difference between this augmented flow and the ideal gradient flow* $\dot{\boldsymbol{\theta}}_t = -\boldsymbol{g}$ *defined with respect to the progress coordinate* $\tau = ht = t\eta Q(1)$*, up to* $O(h^2)$*.*

$$\rho_t \;=\; \frac{\|\dot{\boldsymbol{\theta}}_t^{\text{equiv}} - \dot{\boldsymbol{\theta}}_t\|}{\|\dot{\boldsymbol{\theta}}_t\|} + O(h^2). \quad (55)$$

Proof. From the difference between the consecutive gradients

$$\Delta \boldsymbol{g}_t \;:=\; \boldsymbol{g}_{t+1} - \boldsymbol{g}_t \;=\; -h\boldsymbol{H}_t\boldsymbol{g}_t + O(h^2). \tag{56}$$

Plugging this into the definition of ρ_t, we get

$$\rho_t \;=\; |\alpha_Q| \frac{\|-h\boldsymbol{H}_t\boldsymbol{g}_t + O(h^2)\|}{\|\boldsymbol{g}_t\|} \;=\; \frac{\|(-\boldsymbol{g}_t - h\alpha_Q\boldsymbol{H}_t\boldsymbol{g}_t + O(h^2)) - (-\boldsymbol{g}_t)\|}{\|-\boldsymbol{g}_t\|}. \tag{57}$$

Substituting the definitions of the augmented flow and the ideal gradient flow returns the result. □

From equation (57), we further get

$$\rho_t \;=\; h|\alpha_Q| \frac{\|\boldsymbol{H}_t\boldsymbol{g}_t\|}{\|\boldsymbol{g}_t\|} + O(h^2) \;=\; O(h). \tag{58}$$

That is, the first order alias ρ_t is a *first-order observable of the aliasing effect* of the equivalent continuous-time gradient flow. Figure 1e shows the first order alias ρ_t as a function of the timescale $h = \eta Q(1)$, which is approximately linear in the low-LR and mid-LR regimes, until the learning system starts to approach the edge of stability where the first order approximation breaks down.

Why consecutive gradient alignment c_t is a lagging indicator. In Section 2, we also defined the cosine of the consecutive gradients as a measure of gradient alignment,

$$c_t \;:=\; \frac{\langle \boldsymbol{g}_t, \boldsymbol{g}_{t+1}\rangle}{\|\boldsymbol{g}_t\|\|\boldsymbol{g}_{t+1}\|}. \tag{59}$$

We now prove that c_t is a *second-order observable of the aliasing effect*.

Proposition B.5 (Second order alias.)**.** *Define c_t as in equation (59), and ρ_t as in equation (53). For a small drift in gradient $\Delta\boldsymbol{g}_t = \boldsymbol{g}_{t+1} - \boldsymbol{g}_t$, we have*

$$1 - c_t \;=\; O(\rho_t^2). \tag{60}$$

Proof. Define a projection operator $\Pi^{\perp}_{\boldsymbol{g}_t}$ as

$$\Pi^{\perp}_{\boldsymbol{g}_t} \;=\; I - \frac{\boldsymbol{g}_t\boldsymbol{g}_t^{\top}}{\|\boldsymbol{g}_t\|^2}, \tag{61}$$

that normalizes a vector to be perpendicular to $\boldsymbol{g}_t$. Then the cosine c_t can be expanded as

$$c_t \;=\; 1 - \frac{\left\|\Pi^{\perp}_{\boldsymbol{g}_t}\Delta\boldsymbol{g}_t\right\|^2}{2\|\boldsymbol{g}_t\|^2} + O\left(\frac{\|\Delta\boldsymbol{g}_t\|^3}{\|\boldsymbol{g}_t\|^3}\right). \tag{62}$$

This completes the proof. □

Moreover, since $\Pi^{\perp}_{\boldsymbol{g}_t}$ is a projection operator, we can define a sine function of the gradient drift as

$$\sin\psi_t \;:=\; \frac{\left\|\Pi^{\perp}_{\boldsymbol{g}_t}\Delta\boldsymbol{g}_t\right\|}{\|\Delta\boldsymbol{g}_t\|}. \tag{63}$$

Then c_t can be rewritten as

$$1 - c_t \;=\; \frac{\rho_t^2}{2\alpha_Q^2}\sin^2\psi_t + O(\rho_t^3). \tag{64}$$

Immediately, we can see that $1 - c_t$ is a second-order observable of the aliasing effect.

$$\rho_t \;=\; O(h), \qquad 1 - c_t \;=\; O(h^2). \tag{65}$$

Therefore, the cosine c_t is a lagging indicator of the aliasing effect, compared to the alias ρ_t. In the main text, by comparing Figure 1d and Figure 1e, we can observe a clear difference between the two measurements: the alias ρ_t gradually increases linearly from insignificant to significant values according to the effective learning rate h throughout the low-LR and mid-LR regimes, while the cosine c_t remains near 1 until the learning system closely approaches the edge of stability. Only then c_t suddenly breaks down. The edge of stability emerges as a pure discrete-time phenomenon after the first and the second order aliasing become significant.

C Full Experimental Results

We finally provide the full tables and figures for the experiments summarized in the main text.

C.1 Optimizer Specification

This section states the hyperparameters and characteristic quantities of the 10 optimizers we used. First, Table 1 gives the full specification of the optimizers. In specific, we use one SGD (Robbins & Monro, 1951), two SGD with Heavy Ball (SGDM) (Polyak, 1964), two SGD with Nesterov's Acceleration (SGDN) (Nesterov, 1983), two QHM (Ma & Yarats, 2019) and three GF (Lee et al., 2024) optimizers. We vary the momentum hyperparameter β for SGDM and SGDN, the zero position ν for QHM, and the root locus escape angle θ for GF. The chosen hyperparameters are denoted in their name suffixes so that we can easily identify them in the tables and figures. All optimizers are based on stochastic mini-batch gradient descent with a fixed batch size of 4096. This batch size is chosen to be large enough to be inside the reported *large batch regime* of Andreyev et al. (2026) above the critical batch size of CIFAR-10 task (Krizhevsky, 2009).

Each column of Table 1 is a derived quantity from the optimizer transfer function $Q(z)$ in equation (♣): $z - 1 + k\bar{Q}(z) = 0$ for some gain $k = hS = \eta Q(1)\lambda_{\max}(\boldsymbol{H})$, its first derivative $Q'(z)$, and its root locus. $Q(1)$ is the dc gain and $Q'(1)$ is the first order Taylor coefficient of $Q(z)$ around the dc point $z = 1$. The quantity $\alpha_Q = \frac{1}{2} - \frac{Q'(1)}{Q(1)}$ is the second order Taylor coefficient of the filtered gradient descent, used as a scaling factor in evaluating the first order alias ρ_t in Section 2 in the main text and later used in the proof of Theorem B.2 in the previous section. Define $k_\star$ as the value of coefficient k in the characteristic equation $z - 1 + k\bar{Q}(z) = 0$ for which the first root reaches the unit circle. Then the unit circle escape gain Γ_Q of the optimizer $Q(z)$ is the reciprocal of $k_\star$: $\Gamma_Q = 2/k_\star$. Therefore, the sharpness S at the edge of stability is given by

$$S = \frac{2}{h\Gamma_Q} = \frac{2}{\eta Q(1)\Gamma_Q} = \frac{k_\star}{\eta Q(1)}. \tag{66}$$

This prediction is empirically justified in the main text, e.g., in Figure 1 and 2.

The quantities $k_\star$ and Γ_Q can be analytically derived from the normalized optimizer transfer function $\bar{Q}(z)$ as follows. SGD, SGDM, SGDN, and QHM all have one pole. Their transfer function $\bar{Q}(z)$ is of the form

$$\bar{Q}(z) = \frac{b_0 + b_1 z^{-1}}{1 - \beta z^{-1}}, \qquad k_\star = \frac{2(1+\beta)}{b_0 - b_1}, \tag{67}$$

with its unit circle escape at the Nyquist frequency $z = -1$. On the other hand, Grokfast with an cascade of two identical filters of momentum coefficient β has two poles. Its normalized transfer function $\bar{Q}(z)$ is of the form

$$\bar{Q}(z) = \frac{c}{(1 - \beta z^{-1})^2}, \qquad \cos\theta_\star = \frac{\beta^2 + 2\beta - 1}{2\beta^2}, \qquad k_\star = \frac{1 + 2\beta - \beta^2 - 2\cos\theta_\star}{c}, \tag{68}$$

with its unit circle escape at the angle $\theta_\star$. All these values are summarized in Table 1.

C.2 Regime Boundaries

Table 2 gives the regime boundaries and the number of runs fall into each category of the regimes for the MLP experiments. Since all curves coincide in the low-LR and early mid-LR regimes in the effective learning rate h coordinate, all of the 10 optimizers share the same low-LR/mid-LR boundaries $h_{\text{low|mid}} = 5 \times 10^{-5}$. As we showed in Figure 1a in the main manuscript, the curves start to deviate from each other in the mid-LR regime, giving different mid-LR/high-LR boundaries $h_{\text{mid|high}}$ for each optimizer. Dividing this value by the dc gain $Q(1)$ gives the edge position in the raw learning rate coordinate $\eta_{\text{edge}} = h_{\text{mid|high}}/Q(1)$. The columns N_{low}, N_{mid}, and N_{high} are the numbers of learning rate sweep runs fall into each of the regimes.

C.3 Individual Plots

We finally draw each run summarized as a single dot in the plots of the main text into full graphs.

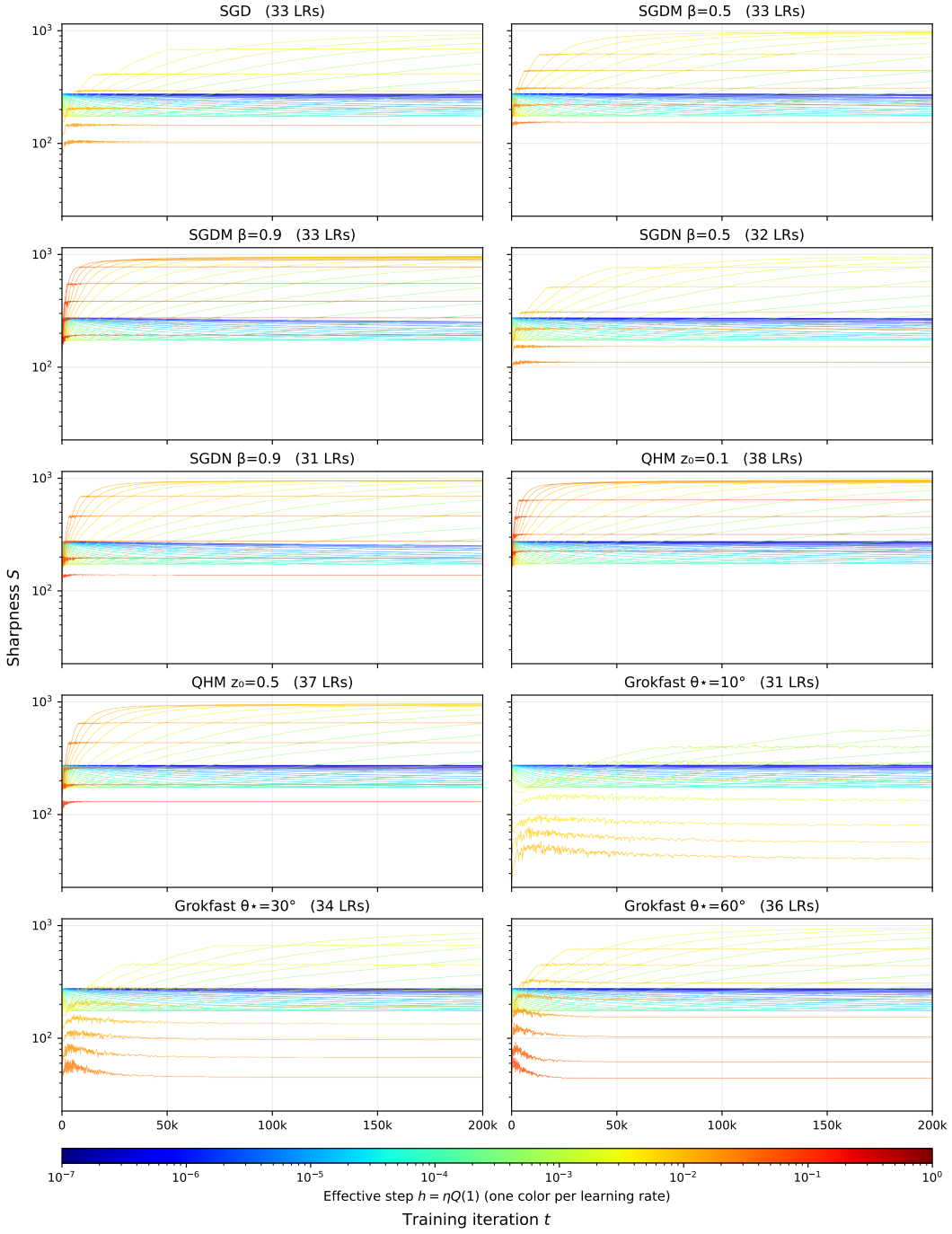


Figure 9: **Individual plots of MLP sharpness S.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

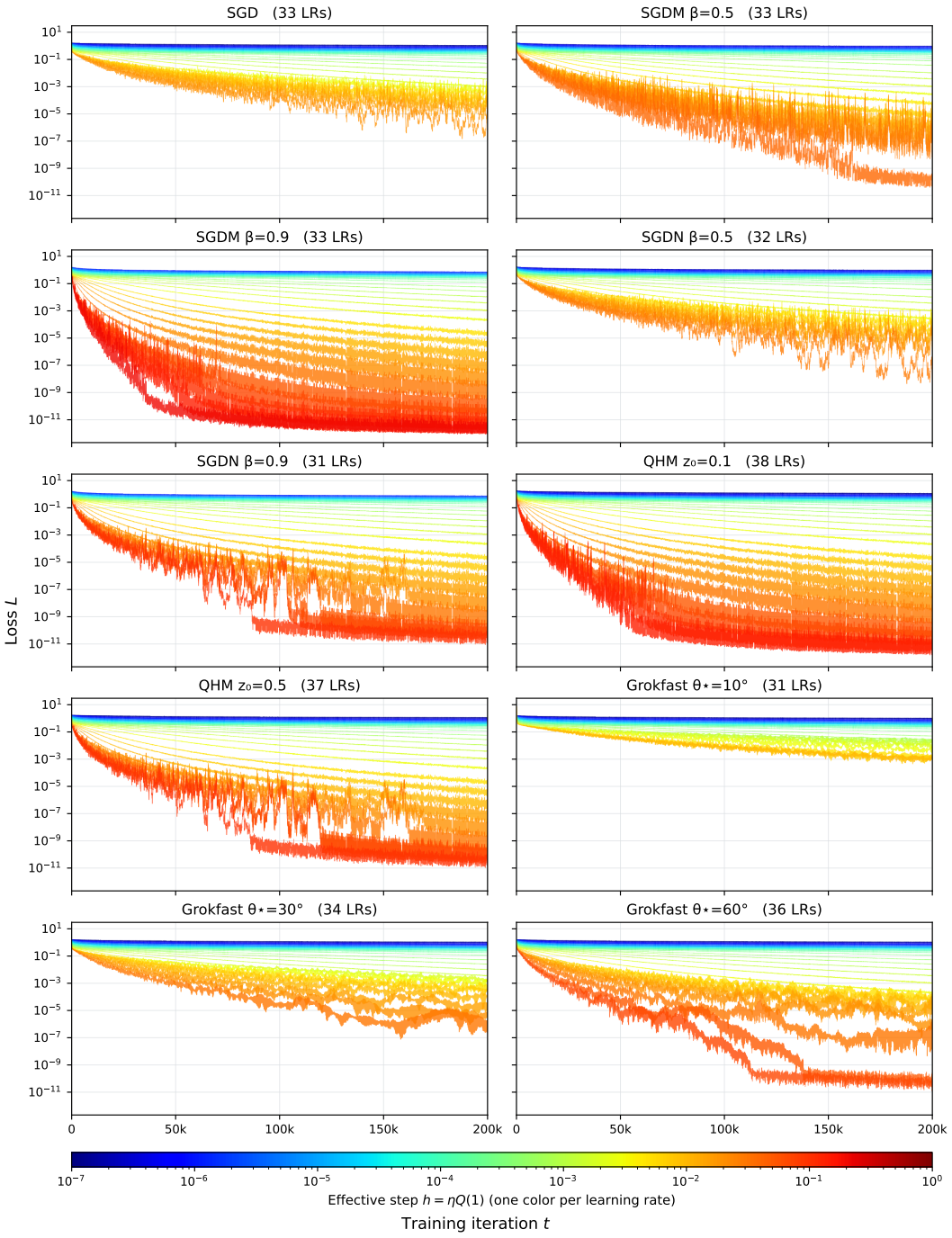


Figure 10: **Individual plots of MLP loss L.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

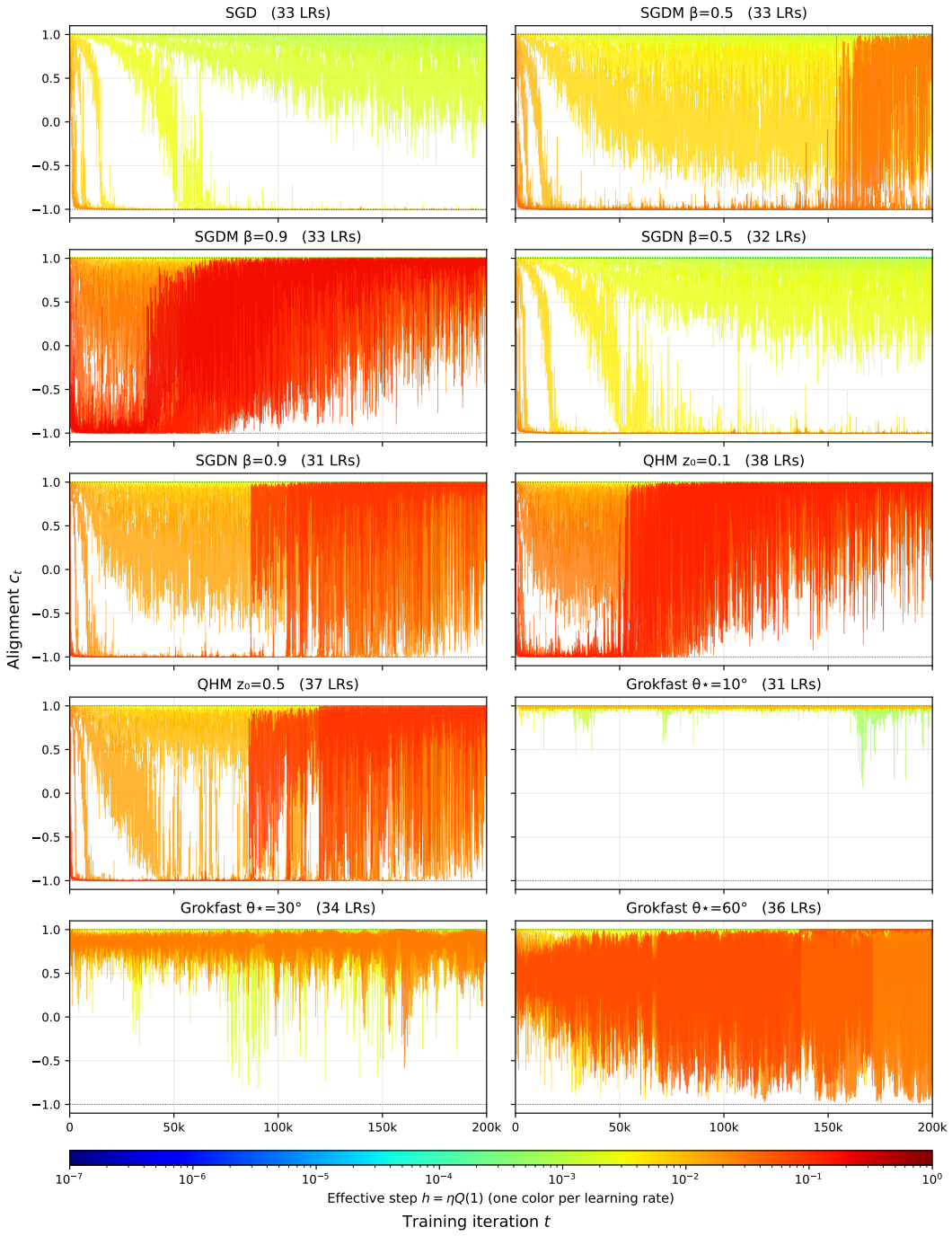


Figure 11: **Individual plots of MLP alignment c_t.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

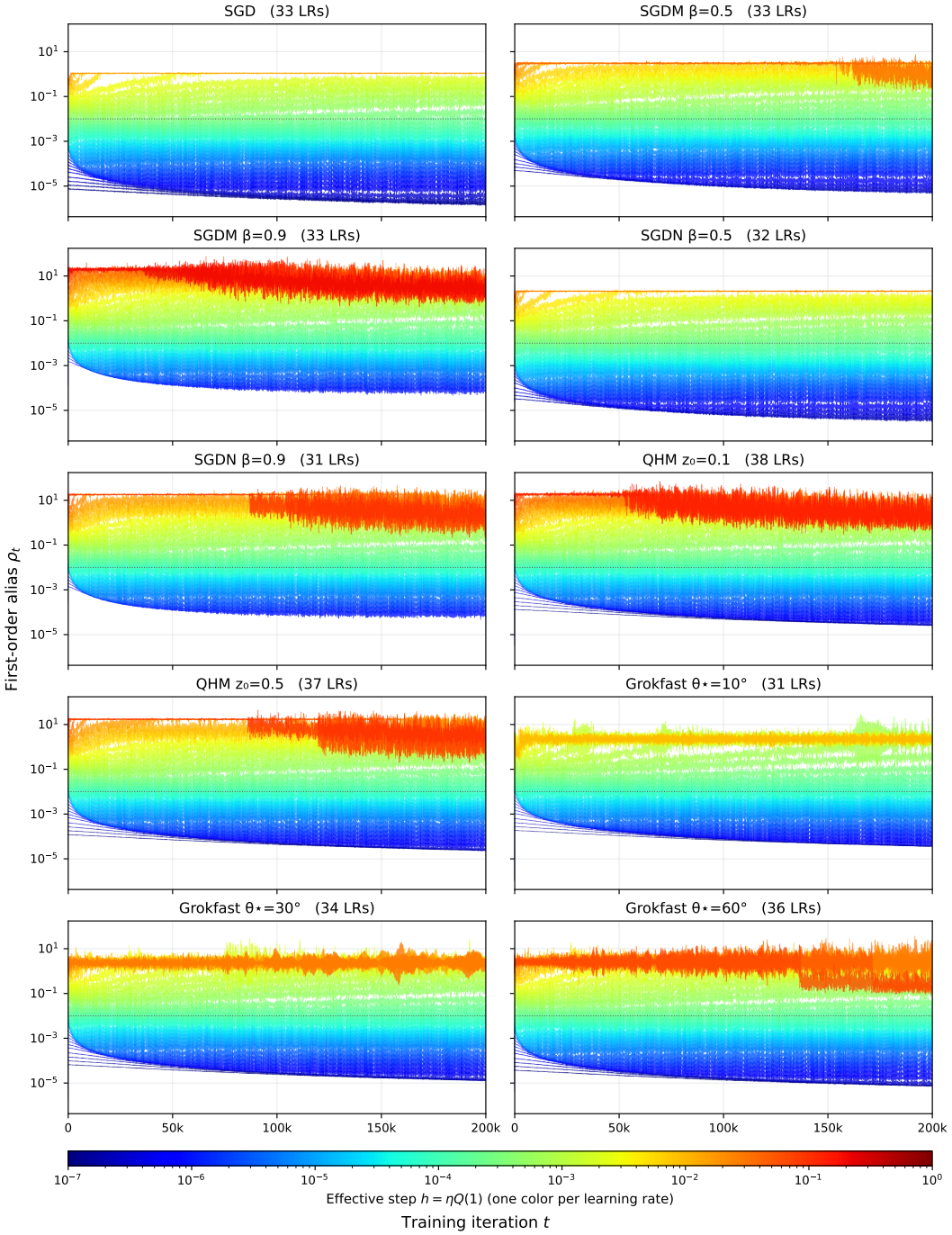


Figure 12: **Individual plots of MLP aliasing ρ_t.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

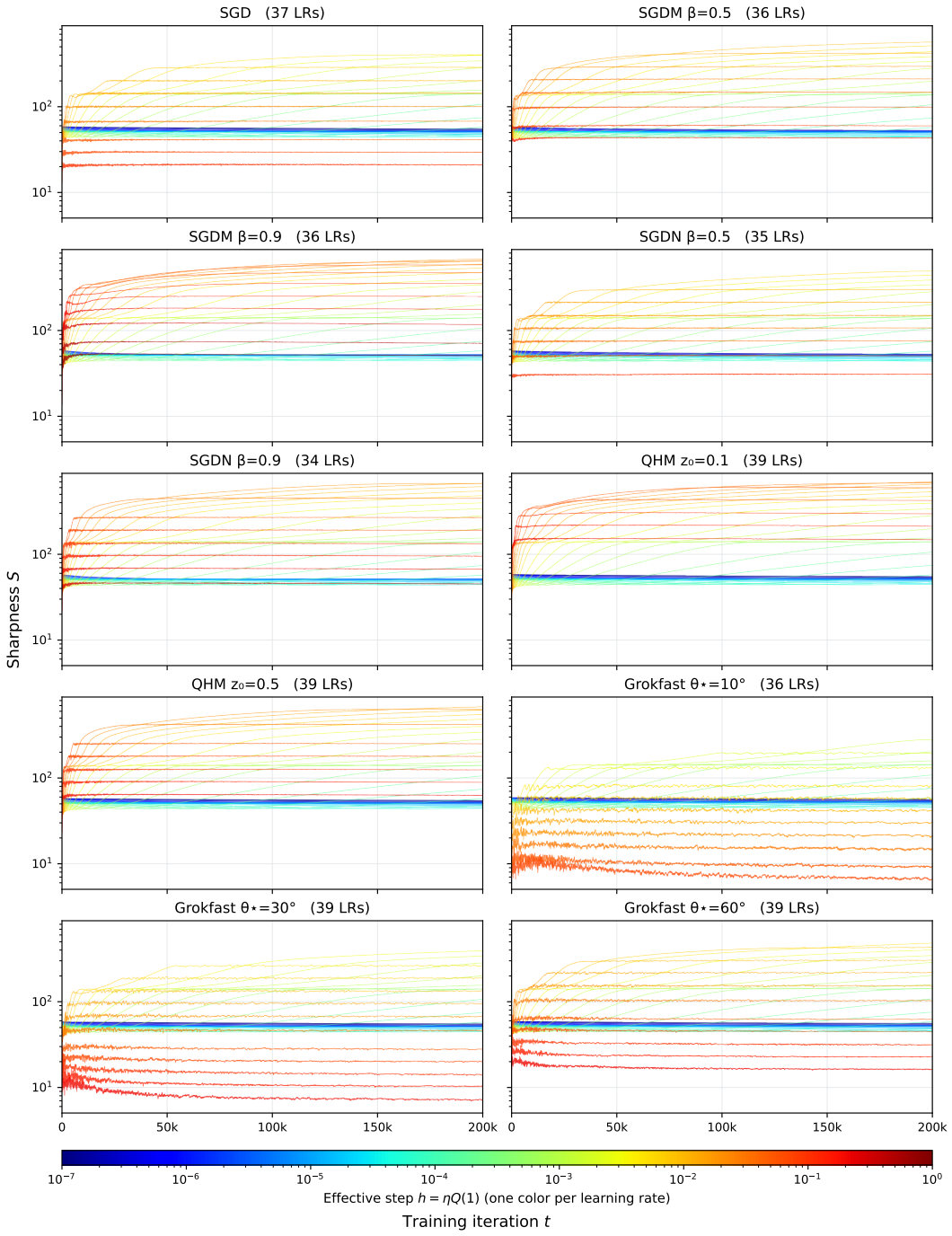


Figure 13: **Individual plots of CNN sharpness S.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

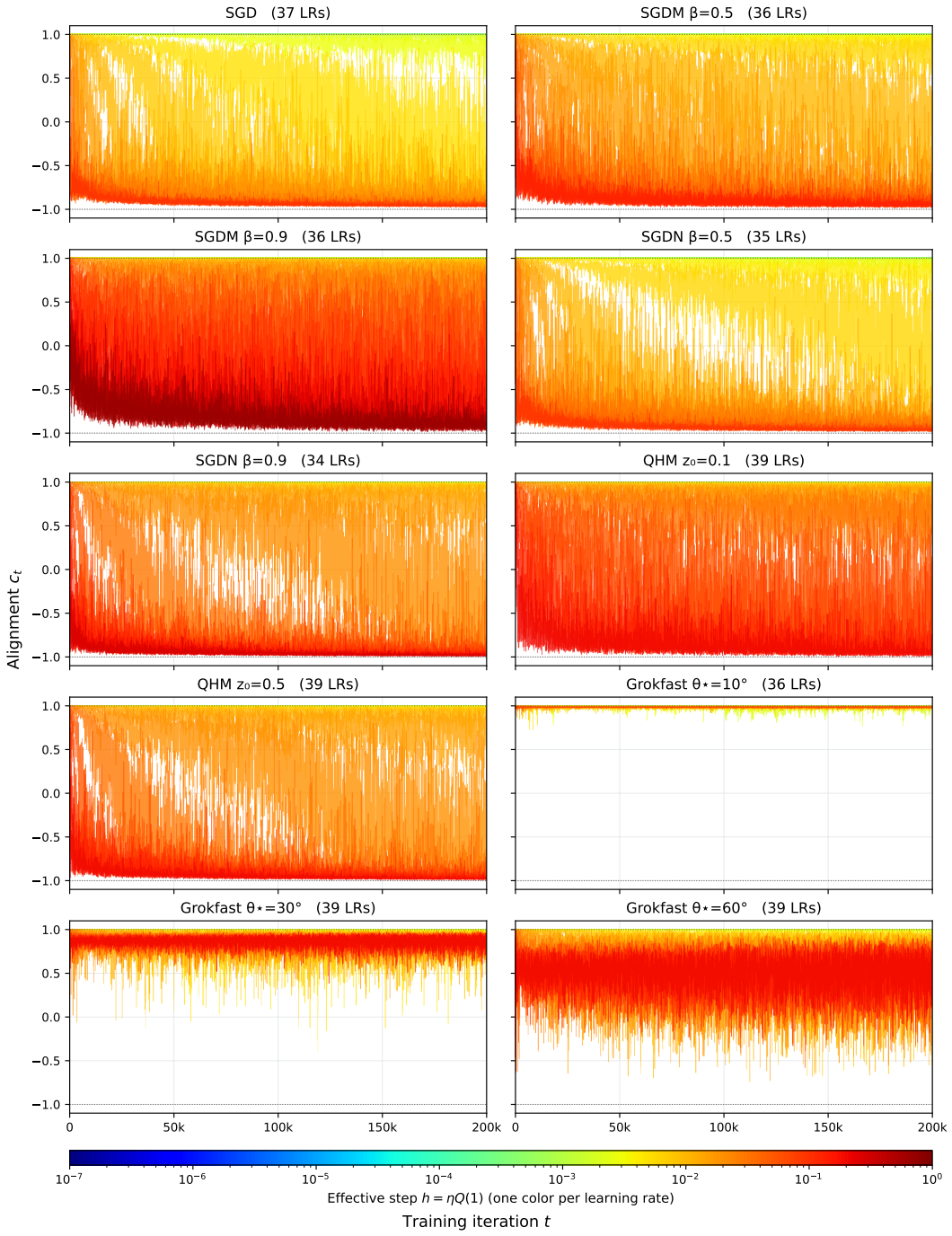


Figure 14: **Individual plots of CNN alignment c_t.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

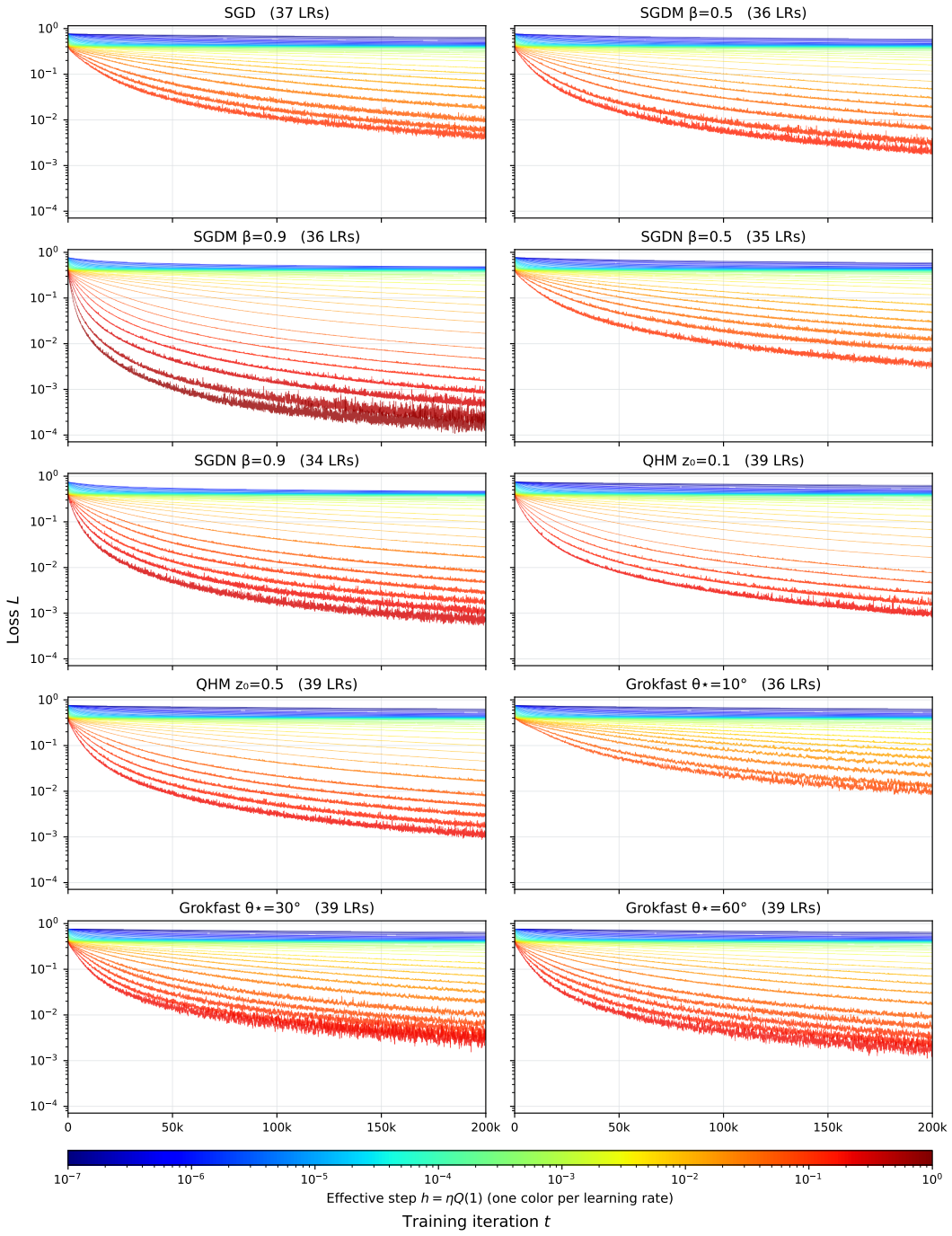


Figure 15: **Individual plots of CNN loss L.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

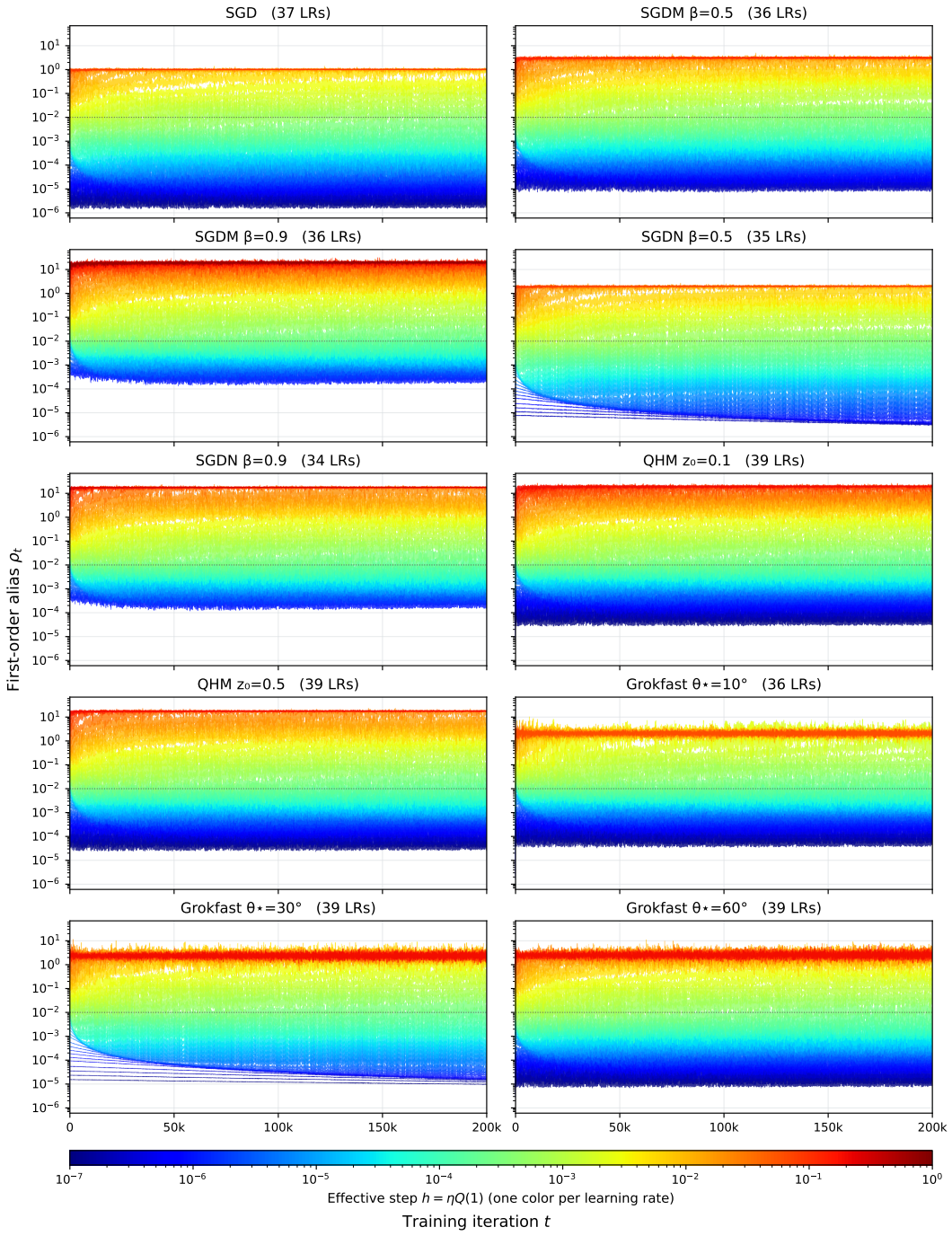


Figure 16: **Individual plots of CNN aliasing ρ_t.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

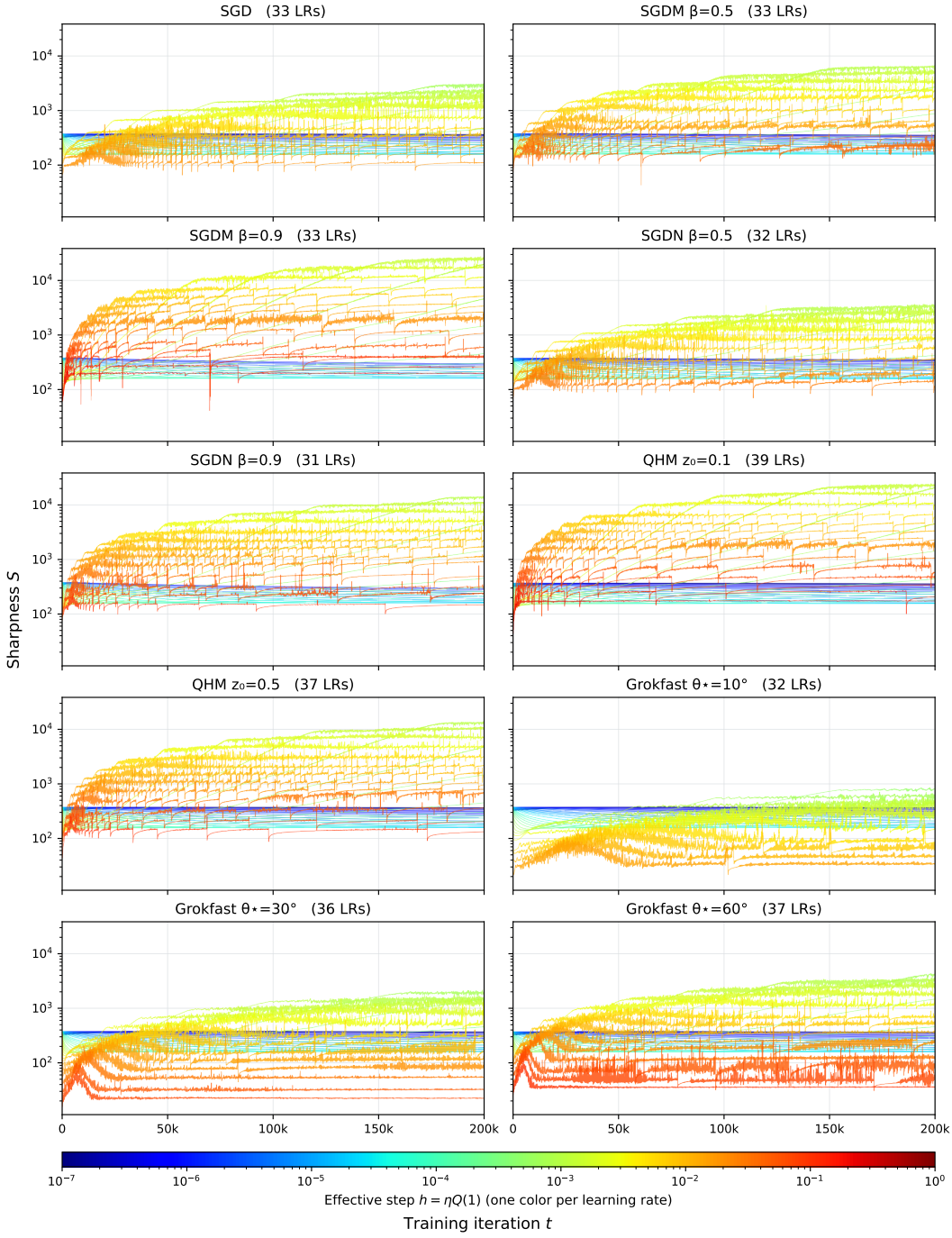


Figure 17: **Individual plots of ViT sharpness S.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

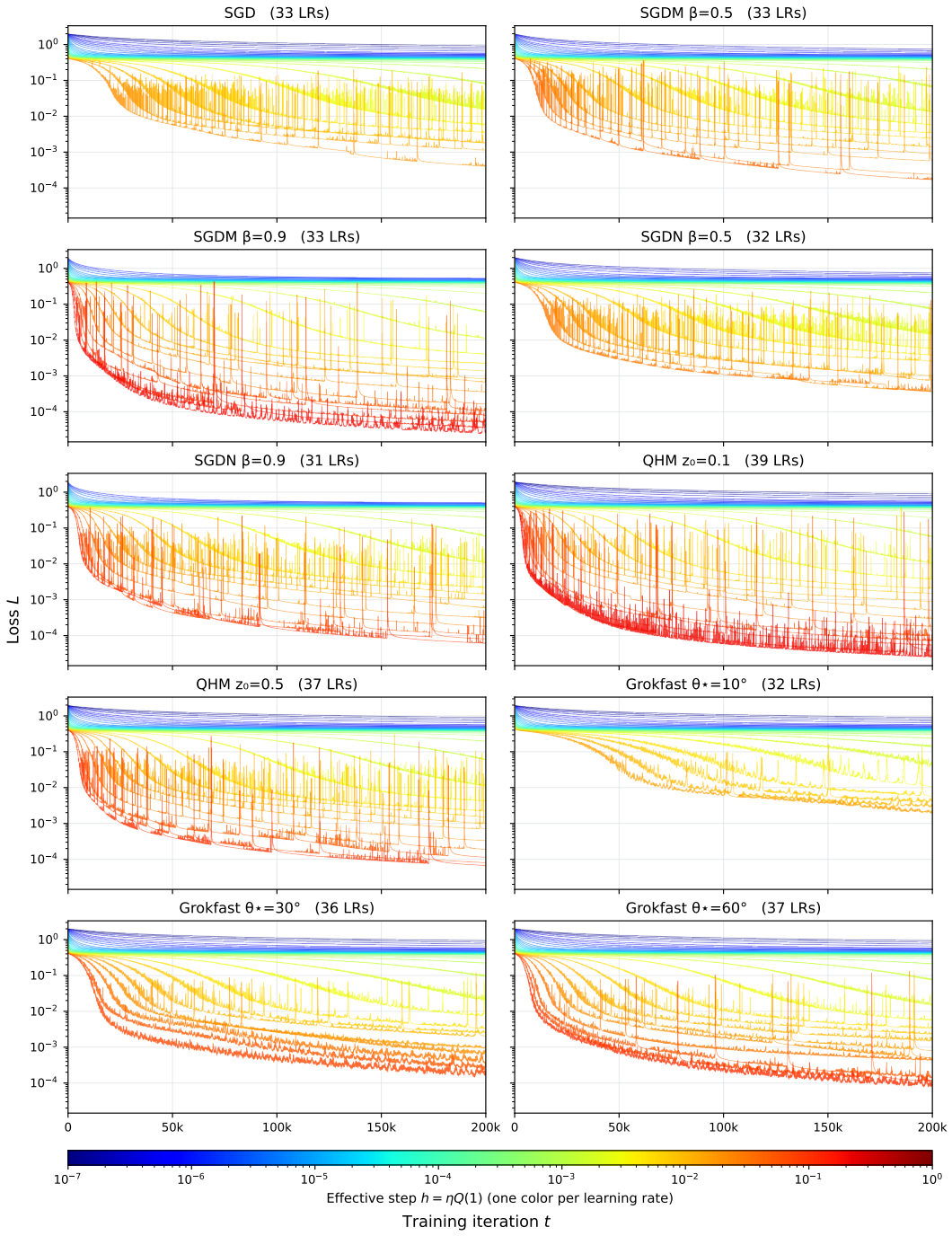


Figure 18: **Individual plots of ViT loss L.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

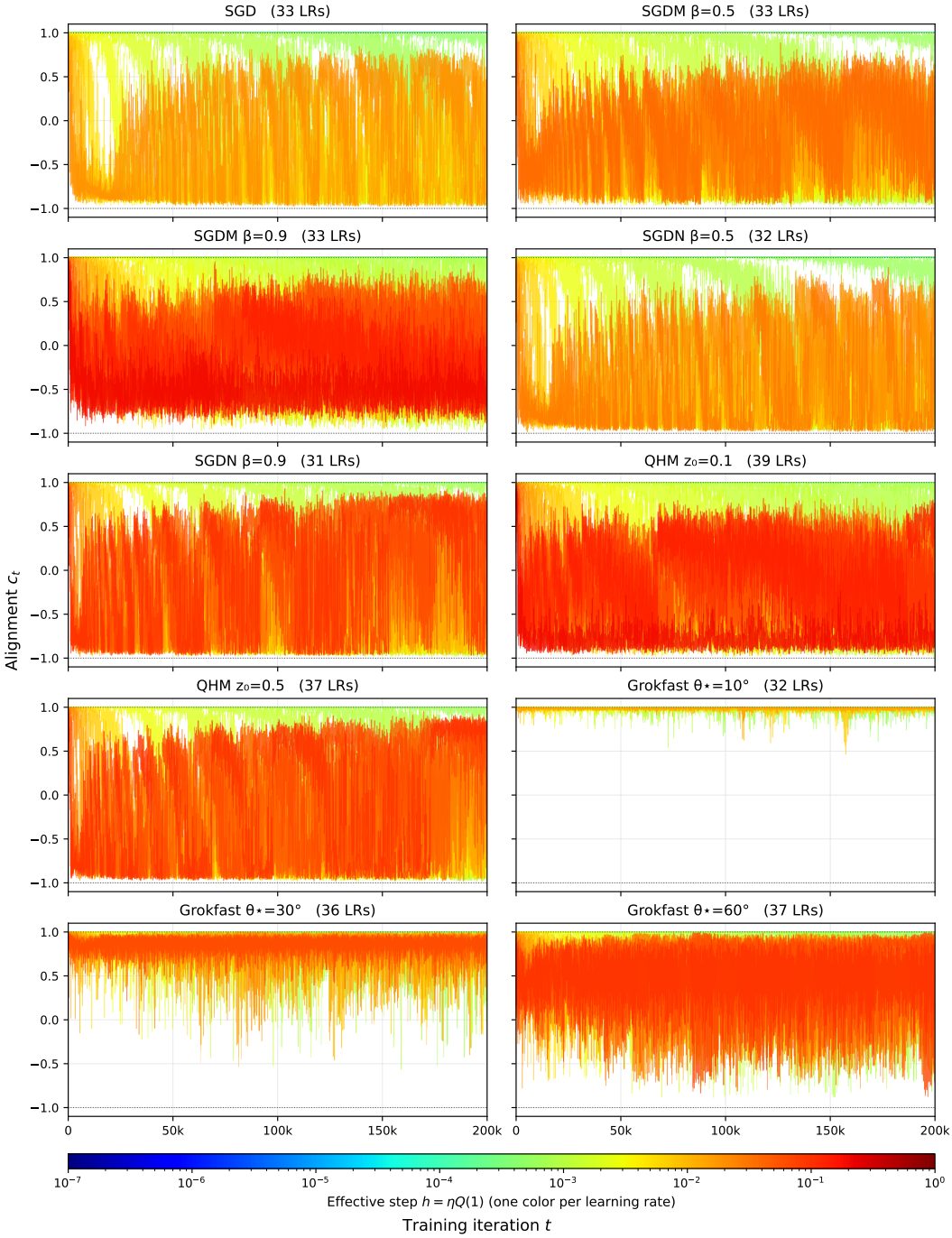


Figure 19: **Individual plots of ViT alignment c_t.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

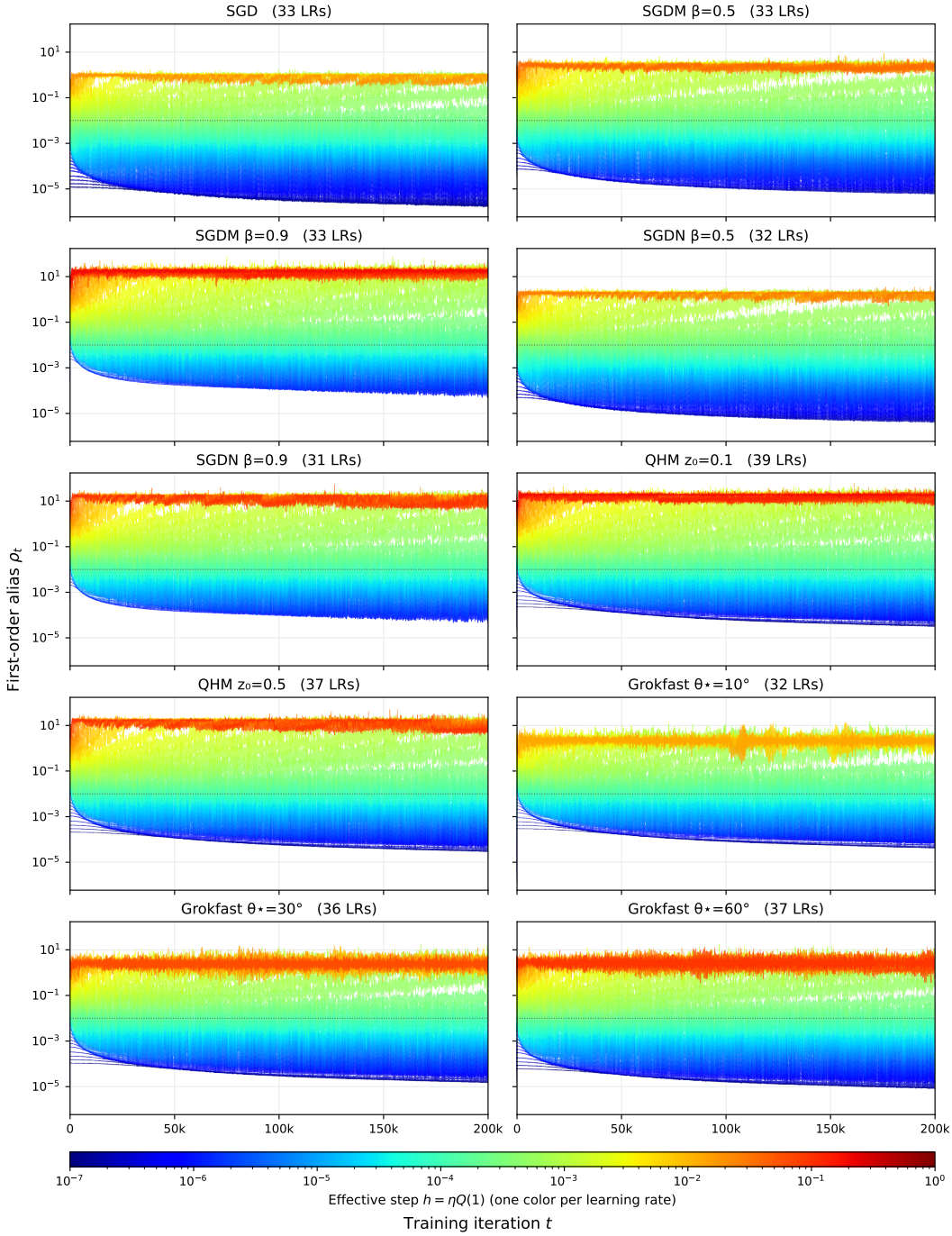


Figure 20: **Individual plots of ViT aliasing ρ_t.** Plots are drawn with respect to the training iteration t coordinate. Different learning rates are colored in different colors.

Table 1: **Characteristic quantities of the first order optimizers used in the experiments.** The parameters of the filters $Q(z)$ and the corresponding edge gains. The edge gain Γ_Q is twice the reciprocal of the first positive gain $k_\star$ for which a root of $z - 1 + k\bar{Q}(z) = 0$ reaches the unit circle.

Optimizer	$Q(1)$	$Q'(1)$	$\alpha_Q = \frac{1}{2} - \frac{Q'(1)}{Q(1)}$	$\theta_\star$	$k_\star$	Γ_Q
SGD	1	0	0.5	180°	2	1
SGDM50	2	-2	1.5	180°	6	0.33333333
SGDM90	10	-90	9.5	180°	38	0.052631579
SGDN50	2	-1	1	180°	3	0.66666667
SGDN90	10	-81	8.6	180°	13.571429	0.14736842
QHM01	1	-8.8888889	9.3888889	180°	31.090909	0.064327485
QHM05	1	-8	8.5	180°	12.666667	0.15789474
GF10	1	-11.473713	11.973713	10°	0.37900746	5.2769409
GF30	1	-3.8637033	4.3637033	30°	1.3032254	1.534654
GF60	1	-2	2.5	60°	3	0.66666667

Table 2: **Regime boundaries of MLP experiments.** All optimizers have the same low-LR/mid-LR boundaries $h_{\text{low|mid}}$, since their curves coincide into a single curve. However, their mid-LR/high-LR boundaries $h_{\text{mid|high}}$ differ. We also provide the edge position in the raw learning rate coordinate $\eta_{\text{edge}} = h_{\text{mid|high}}/Q(1)$. Finally, we also provide the number of runs fall into each category of the regimes N_{low}, N_{mid}, and N_{high} for each optimizer.

Optimizer	$h_{\text{low\|mid}}$	$h_{\text{mid\|high}}$	η_{edge}	N_{low}	N_{mid}	N_{high}
SGD	5e-05	0.003	0.003	16	11	6
SGDM50	5e-05	0.01	0.005	15	13	5
SGDM90	5e-05	0.05	0.005	10	18	5
SGDN50	5e-05	0.004	0.002	15	11	6
SGDN90	5e-05	0.02	0.002	10	16	5
QHM01	5e-05	0.05	0.05	16	18	4
QHM05	5e-05	0.014	0.014	16	15	6
GF10	5e-05	0.0007	0.0007	16	7	8
GF30	5e-05	0.002	0.002	16	10	8
GF60	5e-05	0.005	0.005	16	12	8