How Does Local Landscape Geometry Evolve in Language Model Pre-Training?

Anonymous authors
Paper under double-blind review

Abstract

The scale and expense of pre-training language models make efficient hyperparameter tuning essential, yet a principled guidance is still missing. Recent work shows that the geometry of loss landscape shapes training dynamics of neural networks and further informs hyperparameter choices. In this work, we analyze language model pre-training dynamics from a local landscape geometry perspective. Our study reveals two distinct phases. In the *early* phase, sharpness of the local landscape is initially high, leading to instability and loss plateaus under large learning rates (LRs). Later, the landscape shifts from sharp to flatter regions. This dynamic explains the necessity of LR warmup and further suggests that larger peak LRs require proportionally longer warmup periods. In the *late* phase, the local landscape is governed by the gradient noise scale. Through diffusion-limit analysis, we prove a *depth–flatness trade-off*: high noise from smaller batches widens the loss basin, whereas reduced noise from larger batches deepens it. This theory motivates a dynamic batch-size (BS) scheduler that begins with a small BS and increases it late in training. Together, we provide a unified account of loss landscape evolution, which translates into actionable tuning strategies for large-scale pre-training.

1 Introduction

Training language models efficiently requires carefully tuned hyperparameters, yet a principled guidance for tuning remains unclear. While practitioners often rely on grid search or trial-and-error, these approaches are costly and unreliable at scale. Recent research (Foret et al., 2021; Cohen et al., 2021; Gilmer et al., 2022) has highlighted that the geometry of the local loss landscape offers fundamental insights into optimization, revealing how factors such as sharpness[1] (Keskar et al., 2017; Zhang et al., 2017; Jiang et al., 2020) interact with hyperparameters to shape training dynamics. Consequently, leveraging insights from the local landscape geometry presents a promising path toward principled hyperparameter tuning for language model pre-training.

Several pioneering works have already attempted to study language models from the local landscape geometry perspective. Zhang et al. (2024a); Wang et al. (2025) identified blockwise sharpness patterns in language models through Hessian-based analyses. Wen et al. (2024) introduced the "river-valley" landscape to explain the effectiveness of Warmup-Stable-Decay (WSD) schedules (Hu et al., 2024). Peng et al. (2024); Chen et al. (2025) further visualized the loss landscapes of finetuned language models, offering geometric insights into the safety alignment. However, few studies have investigated the *dynamics* of local landscape geometry during language model pre-training.

To this end, we pose the central research questions of this paper:

1. *How does the local landscape geometry evolve in language model pre-training?*
2. *What implications does this evolution have for principled hyperparameter tuning?*

Our contributions. In this work, we present the first systematic study of the evolution of local landscape geometry during language model pre-training. As illustrated in Figure 1, our analysis reveals two distinct phases, each with significant implications for hyperparameter tuning.

• *Early in Training: From Sharp to Flat Landscapes.* In the early phase, we observe that the model shifts from sharper regions of the loss landscape toward flatter ones, contrary to the progressive

[1] To avoid misunderstanding, we clarify the terminology in Table 1.

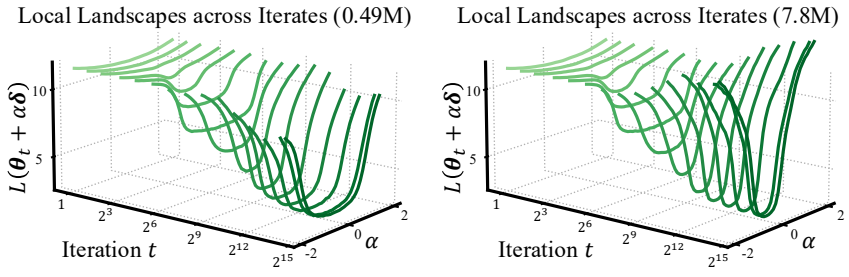

Figure 1: **The evolution of local loss landscape throughout pre-training.** We train `LLaMA-2` models with 170M parameters using different BSs (0.49M and 7.8M), and visualize the one-dimensional loss landscape at iterate $\boldsymbol{\theta}_t$ along a random direction $\boldsymbol{\delta}$, i.e., plot $L(\boldsymbol{\theta}_t + \alpha\boldsymbol{\delta})$ vs. the perturbation coefficient α. The landscapes are shown across different training iterations t. **Early phase.** The landscapes gradually widen/flatten for both training runs. **Late phase.** Training with smaller BS produces wider landscapes than training with larger BS.

sharpening phenomenon in prior works (Cohen et al., 2021; Song & Yun, 2023; Cohen et al., 2025). Lyapunov stability analysis in Section 4 shows that the maximum stable learning rate (LR) is inversely proportional to sharpness. Since sharpness is extremely high early in pre-training, using large peak LRs without sufficient warmup leads to instabilities, such as loss spikes and plateaus (see Figure 2).

Implications. The sharp-to-flat transition explains the necessity of LR warmup: LR should remain small until sharpness has sufficiently decayed, preventing training instabilities. This further provides a practical tuning recipe: within a reasonable range, larger peak LRs require proportionally longer warmup, to safely navigate the sharpest stage of training.

• *Late in Training: Basin Selection Governed by Noise Scale.* In the late phase, the local landscape geometry is largely governed by the noise scale during training, with batch size (BS) B serving as its primary controller. Our analysis shows that smaller BS widens the loss basin, while larger BS deepens it. Theoretically, we analyze the diffusion limit of preconditioned SGD, which uncovers a depth–flatness trade-off: reduced gradient noise tends to minimize the loss, leading to deeper minima; whereas increased noise tends to regularize the sharpness of landscape, moving toward wider ones.

Implications. The trade-off, together with the extensive ramping-time experiment in Figure 6, motivates a principled BS scheduling strategy: begins with a small BS and ramps it until the late phase of training. Our scheduling ensures steady loss reduction with minimal token consumption, ultimately achieving lower terminal loss than constant-BS training. Moreover, since the noise scale is proportional to η/B in our theory, we predict that BS ramping and LR decay reduce the noise scale in similar ways and thus yield comparable performance (see Figure 8).

In summary, our work provides a two-phase picture of landscape evolution in pre-training: an early sharp-to-flat transition that necessitates LR warmup, and a late noise-driven regime that motivates BS scheduling. This unified view advances our understanding of pre-training dynamics and underscores the importance of landscape geometry in offering principled guidance for hyperparameter tuning.

2 Related Works

Local Landscape Geometry (Sharpness) Evolution. Understanding how local landscape geometry, particularly sharpness, evolves during training has drawn significant attention before the success of large language models. Wu et al. (2018); Cohen et al. (2022); Song & Yun (2023); Cohen et al. (2025) showed that initially gradient descent (GD) tends to move from flatter to sharper regions of the landscape. In addition, Jastrzębski et al. (2019); Jastrzebski et al. (2020) argued that in SGD, sharpness also changes monotonically but either increase or decrease depending on the setting. In the later phase, however, sharpness is largely governed by the properties of the optimizer (Zhou et al., 2025). One notable example is that the stochastic noise introduced by SGD and its variants implicitly biases training toward flat minima (Wu et al., 2018; Zhu et al., 2019; Xie et al., 2021; Wu et al., 2022). Yet, these findings are largely restricted to small-scale networks; *In comparison*, our work presents the *first systematic study* of how local landscape geometry evolves in large-scale language model pre-training, offering new insights into LR warmup and the design of BS schedules.

Large-Scale Pre-training: Learning Rate Warmup. Learning rate warmup, first introduced in large-batch ResNet (He et al., 2016; Goyal et al., 2017) and Transformer training (Vaswani et al., 2017), is now standard in large-scale pre-training (Shoeybi et al., 2019; Zhang et al., 2022; Hu et al., 2024). Its mechanism, however, remains only partly understood. Gotmare et al. (2019) showed that warmup prevents excessively large early parameter updates; Bergsma et al. (2025) attributed the early updates to bias reduction, rather than curvature. Gilmer et al. (2022) argued that warmup guides optimization into flatter regions where large LRs are stable; and Kosson et al. (2024) showed in language model pre-training that warmup mitigates momentum bias correction and correlated gradients that otherwise drive unstable representation shifts. Yet no unified explanation exists. *In comparison*, our work views warmup from a *unified* geometric perspective, suggesting that larger peak LRs demand proportionally longer warmup.

Large-Scale Pre-training: Batch Size Schedules. Batch size is another critical hyperparameter in large-scale pre-training, shaping the trade-off between step efficiency and data efficiency. Most prior work (McCandlish et al., 2018; Kaplan et al., 2020; Gray et al., 2023; 2024; Zhang et al., 2025) has focused on the critical batch size (CBS), the point where further increasing BS yields diminishing returns. However, CBS is typically treated as a constant, and much less attention has been given to *BS scheduling*. Early works on adaptive sampling proposed gradually increasing BS to balance efficiency and noise reduction (De et al., 2017; Lau et al., 2024b;a; 2025; Ostroukhov et al., 2024). However, these studies remain mostly theoretical. Advanced language models (Brown et al., 2020; Touvron et al., 2023; Liu et al., 2024; Li et al., 2025) employed stage-wise BS schedules, but without systematic analysis. *In contrast*, our work connects BS scheduling to the evolving local landscape geometry, providing a principled foundation for when and how to expand BS during pre-training.

3 Preliminaries

Basic Notations. We use bold lowercase letters (e.g., $\boldsymbol{x} = (x_i)$) to denote vectors and bold uppercase letters (e.g., $\mathbf{A} = (a_{ij})$) to denote matrices. For a matrix $\mathbf{A}$, let $\|\mathbf{A}\|_2$, $\|\mathbf{A}\|_F$, and $\mathrm{Tr}(\mathbf{A})$ denote its spectral norm, Frobenius norm and trace, respectively. The Hadamard product is denoted by $\odot$.

Theoretical Setup. Our theory focuses on the preconditioned stochastic gradient descent (PSGD). We consider a model with parameters $\boldsymbol{\theta} \in \mathbb{R}^p$ and a training set of n examples. Let $L_i(\boldsymbol{\theta})$ be the fitting error evaluated at the i-th example and $L(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^n L_i(\boldsymbol{\theta})$ be the empirical risk. We analyze the preconditioned SGD with a fixed positive-definite[2] preconditioner $\mathbf{M} \succ 0$. At iteration k, the update rule gives:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta \mathbf{M}(\nabla L(\boldsymbol{\theta}_k) + \boldsymbol{\xi}_k), \tag{1}$$

where $\eta > 0$ is the LR and $\{\boldsymbol{\xi}_k\}$ are i.i.d. random noise vectors with

$$\mathbb{E}[\boldsymbol{\xi}_k] = \mathbf{0}, \quad \mathbb{E}[\boldsymbol{\xi}_k \boldsymbol{\xi}_k^\top] = \boldsymbol{\Sigma}(\boldsymbol{\theta}_k)/B. \tag{2}$$

Note that $\boldsymbol{\Sigma}(\boldsymbol{\theta}_k) = \frac{1}{n}\sum_{i=1}^n \nabla L_i(\boldsymbol{\theta}_k)\nabla L_i(\boldsymbol{\theta}_k)^\top - \nabla L(\boldsymbol{\theta}_k)\nabla L(\boldsymbol{\theta}_k)^\top$ is the gradient covariance at $\boldsymbol{\theta}_k$, and B denotes the BS. During the late phase of training, the model remains close to some global minimum $\boldsymbol{\theta}^\star$ and the loss can be approximated quadratically:

$$L(\boldsymbol{\theta}) = L(\boldsymbol{\theta}^\star) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}^\star)^\top \mathbf{H}(\boldsymbol{\theta}^\star)(\boldsymbol{\theta} - \boldsymbol{\theta}^\star), \quad \mathbf{H}(\boldsymbol{\theta}^\star) := \nabla^2 L(\boldsymbol{\theta}^\star) \succ 0. \tag{3}$$

Similar formulations have been widely used in dynamical stability analyses (Wu et al., 2018; Cohen et al., 2021; Zhou et al., 2025) and theoretical advances on BS scaling (McCandlish et al., 2018).

Experimental Setup. Our experiments are mainly conducted on `LLaMA-2` architecture (Touvron et al., 2023) models with 93M and 170M parameters. Training is performed on the `FineWeb-Edu` dataset (Penedo et al., 2024), with sufficient training budgets ranging from 50 to 1000 tokens-per-parameter (TPP)[3] and a context length of 1024. We adopt AdamW (Kingma & Ba, 2014) with hyperparameters $\beta_1 = 0.95$, $\beta_2 = 0.95$, and weight decay 0.1, together with gradient clipping at 1.0 for stability. The evaluation is conducted on a held-out validation split of approximately 50M tokens. More experiments on larger scales, other architectures, and optimizers are deferred to Section D.

[2] Most practical preconditioners are positive-definite: $\mathbf{M} = I$ for SGD, diagonal $\mathbf{M}$ for AdaGrad (Duchi et al., 2011), RMSProp (Tieleman & Hinton, 2012), Adam, etc.

[3] At least $10\times$ over Chinchilla-optimal tokens (Hoffmann et al., 2022).

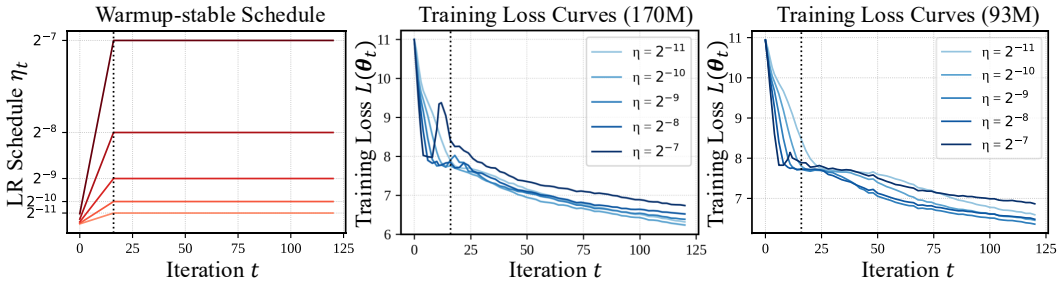

Figure 2: **Loss spikes and plateaus early in training.** We train a series of `LLaMA-2` models with 93M and 170M parameters. We adopt a warmup-stable schedule, where the warmup length is shortened to 16 iterations and the peak LR is varied, $\eta \in \{2^{-11}, 2^{-10}, 2^{-9}, 2^{-8}, 2^{-7}\}$. **(Left).** LR schedule: η_t vs. training iteration t. **(Middle, Right).** Training loss curves for different model sizes: $L(\boldsymbol{\theta}_t)$ vs. training iteration t. The vertical dashed line marks the end of the warmup phase.

Our experiments vary the LRs and BSs. In Section 4, we primarily study the role of LR and warmup length, fixing BS at 7.8M. In Section 5, we focus on the effect of BS, with LR fixed at 2^{-10}. To decouple BS ramping from LR decay, we adopt a *warmup-stable* schedule: after linear warmup to the peak value, the LR remains constant (similar to WSD (Hu et al., 2024), but without decay phase).

4 Early in Pre-Training: From Sharp to Flat Landscapes

In this section, we provide evidence that, during the early phase of pre-training, the local landscape of language models evolves from sharp regions toward flatter ones. We first observe that training with large LRs and insufficient warmup often leads to instability and early loss plateaus. By Lyapunov stability analysis, we then attribute these behaviors to sharp-to-flat dynamics occurring in the initial phase of training. This finding explains why pre-training needs LR warmup and suggests that larger peak LRs require proportionally longer warmup periods.

Motivating Observations: Instability and Loss Plateaus Early in Training. The loss curves for pre-training are typically smooth initially; the model escapes from random initialization and the loss decreases rapidly. Yet, surprisingly, when the warmup length is extremely shortened, we *consistently* observe loss spikes and plateaus near the end of the warmup phase.

To demonstrate this, we train models of different sizes with a fixed warmup length of 16 iterations while varying the peak LR. As shown in Figure 2, a loss plateau reliably appears around the end of the warmup phase across all settings. Additionally, larger LRs produce higher spikes, which mark a characteristic feature of early training instability. Given these results, two natural questions arise:

Q1. *Why does shortened warmup induce training instability?*

Q2. *Why do spikes and plateaus occur only at the very beginning of training?*

To shed light on these questions, we analyze the dynamics of PSGD via Lyapunov stability analysis.

Lyapunov Stability Analysis: Sharpness Matters. Let $\boldsymbol{\theta}_k$, $\tilde{\boldsymbol{\theta}}_k$ be two nearby trajectories, and define their difference as $\boldsymbol{e}_k := \tilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k$. When the noise term $\boldsymbol{\xi}$ is set to zero, the evolution of $\boldsymbol{e}_k$ satisfies:

$$\boldsymbol{e}_{k+1} = \boldsymbol{e}_k - \eta \mathbf{M}(\nabla L(\boldsymbol{\theta}_k + \boldsymbol{e}_k) - \nabla L(\boldsymbol{\theta}_k)) \overset{\text{(Linearization)}}{=} (\mathbf{I} - \eta \mathbf{M}\mathbf{H}(\boldsymbol{\theta}_k))\boldsymbol{e}_k, \tag{4}$$

The dynamics in Equation (4) describe the local sensitivity of the iteration: if matrix $(\mathbf{I} - \eta \mathbf{M}\mathbf{H}(\boldsymbol{\theta}_k))$ repeatedly expands $\boldsymbol{e}_k$, small perturbations grow exponentially and the iterates are linearly unstable. Intuitively, the LR η interacts directly with the curvature of the landscape: if η is too large relative to the sharpest direction, the update rule amplifies perturbations and leads to loss spikes. The following lemma formalizes this stability condition for preconditioned GD.

Lemma 4.1 (Stability Condition for Preconditioned GD)**.** *Define the preconditioned curvature matrix* $\mathbf{S}(\boldsymbol{\theta}_k) := \mathbf{M}^{1/2}\mathbf{H}(\boldsymbol{\theta}_k)\mathbf{M}^{1/2}$*, and let* $\{\lambda\}_{i=1}^{p}$ *be the eigenvalues of* $\mathbf{S}(\boldsymbol{\theta}_k)$*. The linear system in Equation (4) is asymptotically stable (i.e.,* $\lim_{k\to\infty} \boldsymbol{e}_k = \mathbf{0}$*) if* η *satisfies* $0 < \eta < \frac{2}{\lambda_{\max}(\mathbf{S}(\boldsymbol{\theta}_k))}, \forall k \geq 0$.

Lemma 4.1 shows that the Lyapunov stability is governed by the largest eigenvalue of $\mathbf{S}$. If the curvature along the sharpest direction is too large, only a sufficiently small LR can prevent divergence.

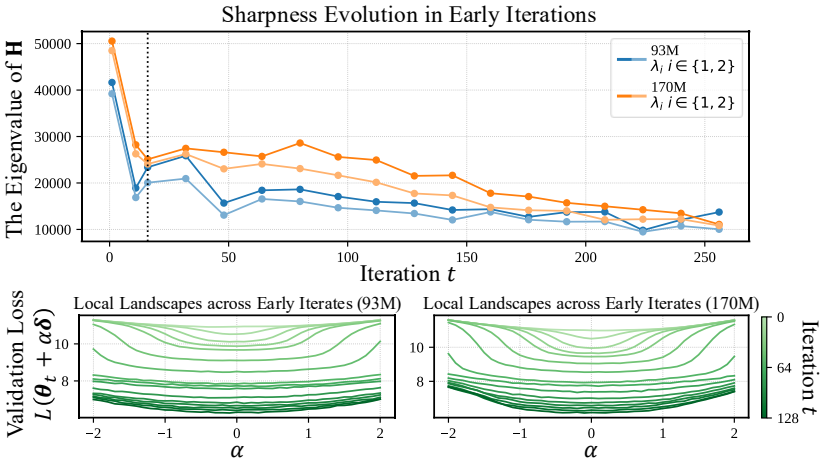

Figure 3: **Early pre-training shifts iterates from sharp to flat regions.** We visualize the local landscape geometry evolution of training runs in Figure 2. For each model size, we select the training run with LR 2^{-10}. **(Top).** Evolution of the top eigenvalues of the Hessian across iterations: $\lambda_i(\mathbf{H}(\boldsymbol{\theta}_t))$ vs. iteration t. **(Bottom).** One-dimensional loss landscape along a random perturbation direction: the perturbed loss $L(\boldsymbol{\theta}_t + \alpha\boldsymbol{\delta})$ vs. perturbation coefficient α, shown across early training iterations t.

We next characterize the one-step loss change as η approaches the stability boundary $2/\lambda_{\max}(\mathbf{S}_k)$.

Lemma 4.2 (One-step Loss Change). *Let $\boldsymbol{\delta}_k := \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k$. Suppose that along the segment $\{\boldsymbol{\theta}_k + \alpha\boldsymbol{\delta}_k : \alpha \in [0, 1]\}$, we have $0 \le \lambda_{\min}(\mathbf{S}(\boldsymbol{\theta}_k + \alpha\boldsymbol{\delta}_k)) \le \lambda_{\max}(\mathbf{S}(\boldsymbol{\theta}_k + \alpha\boldsymbol{\delta}_k)) \le \Lambda_k$. Then,*

$$L(\boldsymbol{\theta}_{k+1}) - L(\boldsymbol{\theta}_k) \le -\eta(1 - \frac{1}{2}\eta\Lambda_k)(\nabla L(\boldsymbol{\theta}_k))^\top \mathbf{M} \nabla L(\boldsymbol{\theta}_k).$$

In particular, if $\eta \uparrow 2/\Lambda_k$, the guaranteed decrease per step $(L(\boldsymbol{\theta}_k) - L(\boldsymbol{\theta}_{k+1}))/\eta \to 0$.

Lemma 4.2 states that when η is close to $2/\Lambda_k$, each update yields only a marginal decrease in loss. Together with Lemma 4.1, it is clear that training near the stability boundary naturally leads to characteristic loss spikes and plateaus.

Importantly, the stability boundary is determined by the sharpness of the loss landscape. To further address **Q1-2**, we analyze how sharpness evolves during the early phase of pre-training.

The Early Dynamics: From Sharp to Flat Landscapes. We study how the local landscape geometry, particularly the sharpness, evolves for training runs in Figure 2. Specifically, we track the evolution of the top eigenvalues of the Hessian[4] $\mathbf{H}(\boldsymbol{\theta}_t)$ during early pre-training. For the early checkpoints $\boldsymbol{\theta}_t$, we also visualize the one-dimensional loss landscape along a random direction by plotting the function $\mathcal{L}(\alpha) := L(\boldsymbol{\theta}_t + \alpha\boldsymbol{\delta})$ with $\boldsymbol{\delta} \sim \mathcal{N}(0, \mathbf{I})$. Li et al. (2018) showed that such random-direction visualizations reliably capture intrinsic properties of the loss landscape properties, such as sharpness. To ensure fair comparison across iterations, we fix the same random vector $\boldsymbol{\delta}$ for all $\boldsymbol{\theta}_t$.

In Figure 3 (top), the largest eigenvalues of the Hessian $\mathbf{H}(\boldsymbol{\theta}_t)$ start at high values[5] and then decrease sharply, indicating a substantial reduction in curvature along the sharpest direction. Furthermore, in Figure 3 (bottom), the loss landscape along a random direction progressively widens as training proceeds, confirming that the model shifts from sharp to flat regions even in the most directions.

A Tuning Recipe: Larger Peak LR, Longer Warmup. We have seen that training stability depends on sharpness: when the landscape is steep, only a sufficiently small LR can keep updates stable; and pre-training initially traverses from sharp landscapes to flatter ones. Now let us return to **Q1** and **Q2**:

A1. *If the warmup phase is shortened, the LR rises too quickly while the model is still in sharp regions, leading to loss spikes and plateaus.*

[4]Following Cohen et al. (2021), we use the `Lanczos` algorithm to calculate top eigenvalues of Hessian.

[5]In fact, at initialization, sharpness is extremely low but rises sharply after the first update. The sharpness curves reported in Figure 3 therefore start from the first iteration.

A2. *As training progresses, the landscape becomes flatter and the same LR no longer threatens stability, which explains why instability is confined to the very beginning.*

Therefore, in practice, we need a sufficiently long warmup phase to keep the LR small until sharpness has decayed, thereby preventing loss spikes and plateaus. This rationale further suggests a practical tuning recipe: *the larger the peak LR, the longer the warmup should be*, ensuring iterates safely transition into flatter landscapes before reaching full step size.

To validate this, we train models with varied peak LRs η and warmup lengths T_w (in iterations). In Figure 4, within a LR range of 2^{-8} to 2^{-11}, larger peak LRs require *proportionally* longer warmup to achieve the optimal validation loss $L(\boldsymbol{\theta}_{\text{best}})$. However, this proportionality does not hold universally. When $\eta = 2^{-7}$, the optimal warmup length remains 2^{10} iterations, the same as for $\eta = 2^{-8}$. Thus, the relationship applies within a reasonable range, when both the peak LR and warmup length are neither too small nor too large.

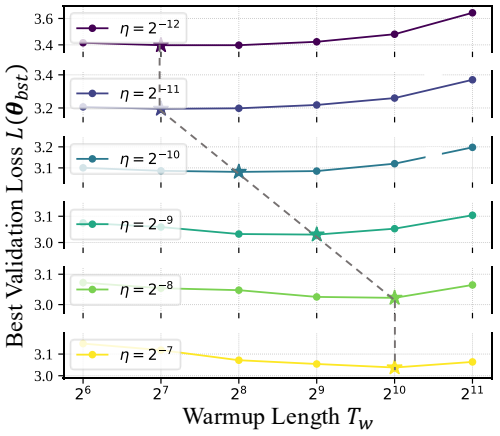

Figure 4: **Larger Peak LR, Longer Warmup.** We train a series of `LLaMA-2` models with 170M parameters and 100 TPP. We vary the peak LRs η and warmup lengths T_w. We plot the best validation loss $L(\boldsymbol{\theta}_{\text{bst}})$ vs. T_w for different η. For each η, the optimal T_w is highlighted with a star.

Comparison with Gilmer et al. (2022); Kalra & Barkeshli (2024). These works also studied warmup from a sharpness perspective but focused mainly on standard image classification tasks (e.g., ResNet on CIFAR-10) and full-batch gradient descent. In contrast, our work investigates warmup in the context of *large-scale* language model pre-training, visualizing the sharpness evolution under a general and practical training setup.

5 Late in Pre-Training: Local Landscape Governed by Noise Scale

In this section, we turn to the local landscape geometry in the late phase. We observe that BS plays a central role: training with a large BS tends to find a *deeper* basin of the landscape, whereas a small BS favors a *wider* basin. Theoretically, we prove that this trade-off between *widen* or *deepen* is governed by the *noise scale*. Building on this, we propose a BS scheduler for the data-limited regime: *use small BS early and ramp the BS late*, which consumes fewer tokens to achieve the same loss.

The Effect of BS: Local Landscapes Late in Training. We conduct experiments to systematically investigate the role of BS in shaping the local landscape geometry during the late phase of pre-training. Specifically, we train models with different BSs for $T = 20{,}480$ iterations. Figure 5 (top left) shows the validation loss curves for each run. Evidently, larger BS consistently leads to lower terminal loss and faster convergence in term of iterations[6]. We then visualize the loss landscape around the final iterate $\boldsymbol{\theta}_T$. In Figure 5 (top right), it is clear that small BS produces flatter basins, whereas large BS yields deeper ones. To further demonstrate, Figure 5 (bottom) compares the landscape evolution of runs with $B = 0.49$M and $B = 7.8$M, indicating that in the late training phase, larger BS tends to deepen the basin, while smaller BS shifts toward wider basins.

Despite these results, two key questions remain:

Q3. *Why is there a trade-off between widening and deepening the basin?*

Q4. *Which factor underlying the hyperparameter BS governs this trade-off?*

To delve into **Q3-4**, we revisit the stochastic differential equation (SDE) in Jastrzębski et al. (2017).

Widen or Deepen: Noise Scale Governs Basin Selection. Following Jastrzębski et al. (2017), we take the continuous-time limit of Equation (1). Suppose that the noise covariance satisfies[7] $\frac{\eta}{B}\mathbf{M}\boldsymbol{\Sigma}(\boldsymbol{\theta}^\star)\mathbf{M}^\top = 2\tau\mathbf{M} + \mathcal{O}(\eta)$ for some temperature $\tau > 0$. As $\eta \to 0$, the scaled discrete process

[6] In terms of processed tokens, small BS training converges faster.

[7] The assumption of noise covariance is justified in Section C.2.

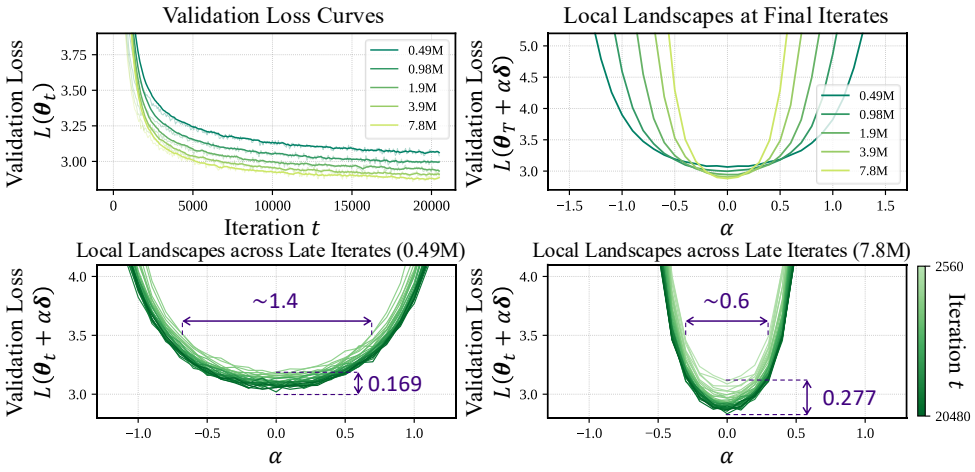

Figure 5: **Large BS deepens the basin, small BS widens the basin.** We train a series of `LLaMA-2` models (170M) for $T = 20{,}480$ iterations, using BSs $B \in \{0.49\text{M}, 0.98\text{M}, 1.9\text{M}, 3.9\text{M}, 7.8\text{M}\}$. **(Top left).** Validation loss curves for different BSs: $L(\boldsymbol{\theta}_t)$ vs. training iteration t. **(Top right).** One-dimensional loss landscapes at the final iterates $\boldsymbol{\theta}_T$ along a random perturbation direction: perturbed loss $L(\boldsymbol{\theta}_T + \alpha\boldsymbol{\delta})$ vs. perturbation coefficient α, visualized across different BSs. **(Bottom).** One-dimensional loss landscape: the perturbed loss $L(\boldsymbol{\theta}_t + \alpha\boldsymbol{\delta})$ vs. perturbation coefficient α, shown across late training iterations t for $B = 0.49\text{M}$ and $B = 7.8\text{M}$.

$\boldsymbol{\theta}_{\lfloor t/\eta \rfloor}$ converges weakly to the Itô SDE:

$$d\boldsymbol{\theta}_t = -\mathbf{M}\nabla L(\boldsymbol{\theta}_t)dt + \sqrt{2\tau}\mathbf{M}^{1/2}dW_t \tag{5}$$

where W_t is standard Brownian motion and the noise scale τ is proportional to η/B.

Building on Equation (5) and the local quadratic model in Equation (3), we establish the trade-off between deepening and widening the loss basin.

Theorem 5.1 (Depth-Flatness Trade-off)**.** *Let the empirical risk $L(\boldsymbol{\theta})$ admit multiple local minima $\{\boldsymbol{\theta}_i^\star\}_{i=1}^m$ with Hessians $\mathbf{H}(\boldsymbol{\theta}_i^\star) \succ 0$. Under the SDE in Equation (5) with temperature τ, the stationary probability that training resides in basin i is given by:*

$$P_\tau(\textit{basin } i) = \frac{\exp(-F_i(\tau)/\tau)}{\sum_j \exp(-F_j(\tau)/\tau)}, \quad F_i(\tau) := L(\boldsymbol{\theta}_i^\star) + \frac{\tau}{2}\log\det \mathbf{H}(\boldsymbol{\theta}_i^\star).$$

Theorem 5.1 states that the basin selection is controlled by the free energy function $F(\tau) = L(\boldsymbol{\theta}^\star) + \frac{\tau}{2}\log\det\mathbf{H}(\boldsymbol{\theta}^\star)$. In early training, the loss term $L(\boldsymbol{\theta}^\star)$ dominates, so the model primarily seeks regions of lower loss. In later training, $L(\boldsymbol{\theta}^\star)$ is comparable to the flatness penalty $\log\det\mathbf{H}(\boldsymbol{\theta}^\star)$, and basin selection becomes increasingly sensitive to the noise scale $\tau \propto \eta/B$.

Efficient Pre-Training: A BS Scheduler in Data-Limited Regime. Turning back to **Q3** and **Q4**, the trade-off arises because basin selection balances loss minimization against curvature regularization (**A3**), with the governing factor being the noise scale τ (**A4**). Since the primary objective of pre-training is to minimize the training loss[8], this balance naturally favors largest BS available (small τ). In practice, however, data availability is limited, and excessively large BS substantially increase data consumption[9]. Thus, scheduling BS in pre-training is crucial, particularly in the data-limited regime.

• **BS Scheduler: Design Principle I.** Inspired by our theory, loss reduction dominates early in training, during which large BS yields limited benefit. This suggests the first design principle.

Design Principle I. *Start the training process with a small BS before increasing it later.*

Related ideas were noted by Li et al. (2025); Merrill et al. (2025), often referred to as *BS warmup*. However, a key difference in our design lies in when the BS should be increased. Surprisingly, we find that ramping BS *later in training* yields consistently *greater* performance.

[8]Note that our analysis focuses on how reduced noise (e.g., via larger BS) helps the optimizer move into deeper minima, conceptually different from the flat-minima perspective in fully interpolating regimes.

[9]For example, in Figure 5 (top right), when $B = 7.8\text{M}$, the run consumes approximately 160B tokens.

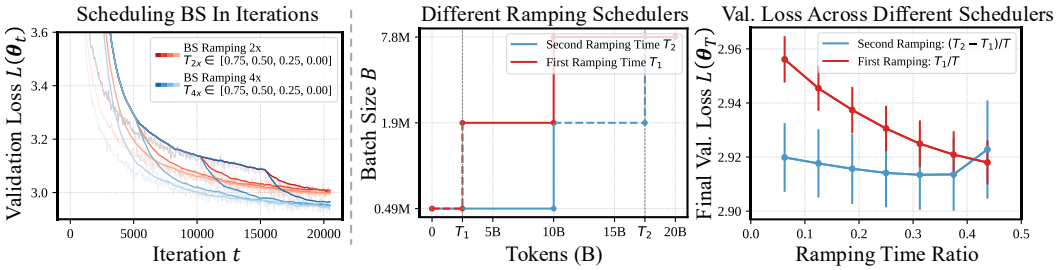

Figure 6: **(Left) Collapse of loss curves under different BS schedules.** Validation loss curves for training with different BS scheduling. In all runs, BS starts at 0.49M. For blue curves, BS is ramped up to $4\times$ its initial value; for red curves, BS is ramped to $2\times$. The ramping times, $T_{2\times}$ or $T_{4\times}$, are varied across different positions. **(Middle, Right) Ramping BS is more efficient late in training.** We evaluate a two-stage BS-ramping schedule with ramp times T_1 and T_2. For the red curves, we fix $T_2 = 10$B and vary T_1; for the blue curves, we fix $T_1 = 10$B and vary T_2. **(Middle).** Illustration of BS schedulers. **(Right).** Final validation loss vs. the relative ramping time, i.e., $(T_1)/T, (T_2 - T_1)/T \in [0, 0.5]$, where T denotes the total training tokens.

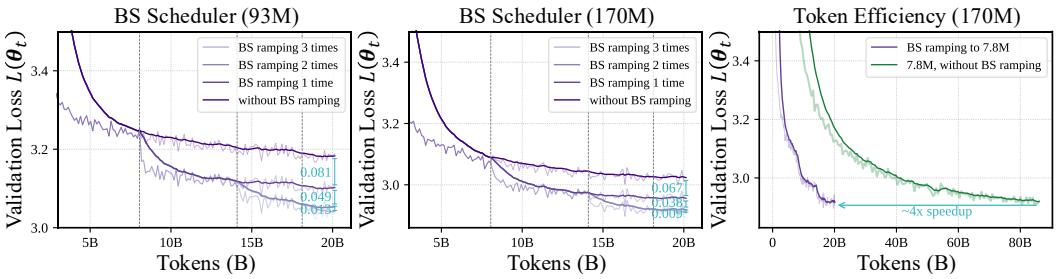

Figure 7: **BS scheduling improves data efficiency.** We train `LLaMA-2` models with 93M and 170M parameters, using a BS schedule that starts at 0.49M and increases by $4\times$ at each ramp. Models are trained with $1, 2$, or 3 ramping steps, while models without ramping serve as the baseline. Vertical gray dashed lines indicate ramping positions. **(Left, Middle).** The validation curves for each run. **(Right).** Comparison between training with BS ramping to 7.8M and training with a fixed 7.8M BS.

• **BS Scheduler: Design Principle II.** To study this, we train models with different BS schedulers while keeping total training iterations fixed. In Figure 6 (left), all runs begin with an initial BS of 0.49M and ramp up to either $4\times$ or $2\times$ that value at different training iterations. Remarkably, all loss curves eventually collapse onto the same trajectory, regardless of when the BS ramping occurs. Note that when measured at the same training iteration, ramping the BS earlier results in higher data consumption. This indicates that early BS ramping offers no efficiency advantage, achieving the same loss but consuming more data.

We next evaluate BS schedules under a fixed token budget. Specifically, we consider a two-stage BS-ramping scheduler characterized by ramp times T_1 and T_2. To isolate the effect of each stage, we vary either T_1 or T_2 while keeping the other fixed. See Figure 6 (middle) for an illustration. In Figure 6 (right), a clear trend emerges: BS ramping is most effective when applied late in training (i.e., with T_1 and T_2 large), whereas ramping too early consistently harms final performance.

Together, since BS ramping ultimately leads all runs onto the same trajectory, delaying it allows maximal progress (lower loss) along that trajectory under a data-limited budget. This behavior also aligns with our theory. In the late phase of training, the flatness penalty becomes comparable to the loss term, and BS ramping sharply reduces the noise scale, driving rapid convergence toward deeper minima. This consistency between theory and practice leads to our second principle.

Design Principle II. *Ramp the batch size late in training—when loss reduction becomes marginal.*

To further validate our design principle, we train models using a BS schedule that starts at 0.49M and ramps by $4\times$ whenever loss minimization slows. In Figure 7 (left, middle), models with $1, 2$ or 3 BS ramping steps achieves significant lower validation loss. While additional ramping steps provide diminishing returns, each step still offers a measurable improvement. Moreover, Figure 7 (right)

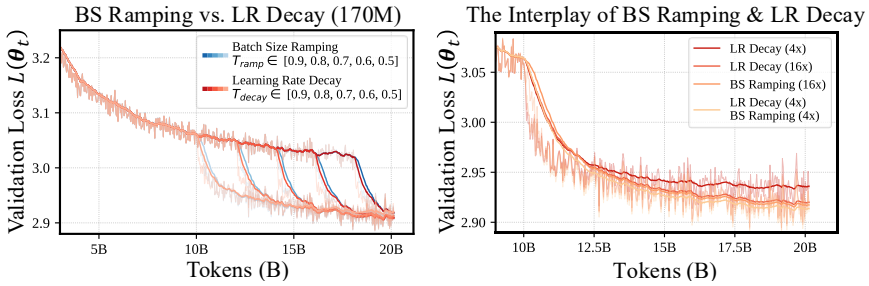

Figure 8: **(Left) BS ramping performs similarly to LR decay.** Validation loss curves for training with either BS ramping or LR decay. For BS ramping, BS increases to $16\times$ its initial value; for LR decay, the LR drops to $1/16$ of its initial value. Each method applies a single step at varying positions. **(Right) Interplay between BS ramping and LR decay.** We evaluate four different scheduling strategies. At the 10B tokens, (a) LR drops to $1/4$ of its initial value; (b) LR drops to $1/16$; (c) BS ramps to $16\times$. (d) LR drops to $1/4$ and BS ramps to $4\times$. In all runs (both left and right), BS starts at 0.49M and LR begins at 2^{-10} (after linear warmup).

highlights the data-efficiency of the BS scheduling: ramping the BS up to 7.8M achieves nearly the same final validation loss as training with a fixed 7.8M BS, but requires only about $\frac{1}{4}$ of the tokens (i.e., a $\sim 4\times$ speedup). These results confirm that our BS scheduling design preserves the benefits of large BS while substantially reducing data consumption.

Comparison with McCandlish et al. (2018); Merrill et al. (2025). McCandlish et al. (2018) linked BS scaling to the gradient noise and introduced the notion of CBS. Merrill et al. (2025) explored the BS scheduling (BS warmup), doubling BS once the CBS exceeds the current BS. We extend these works by showing that the noise scale governs the depth–flatness trade-off in basin selection, and by proposing a BS scheduling design principle that ramps the BS late in training.

6 More Discussions: LR Decay and BS Ramping

So far, we have excluded LR decay in our experiments to isolate the effect of BS ramping. Yet, recall that the noise scale τ is proportional to η/B. Our theory suggests that decaying the LR and ramping the BS both reduce the noise scale, and thus may have similar effects on basin selection.

Comparing BS Ramping with LR Decay. To study this, we train models using either BS ramping or LR decay. Both methods apply a one-time step change: BS ramping multiplies the BS by 16 at T_{ramp}, while LR decay divides the LR by 16 at T_{decay}. We align T_{ramp} and T_{decay} so that the changes occur at the same positions, enabling a direct comparison of their effects. In Figure 8 (left), BS ramping and LR decay produce remarkably similar validation loss curves across all change positions, consistent with the idea that both reduce the noise scale in comparable ways.

Interacting BS Ramping with LR Decay. Furthermore, we study the combined effect of using both BS ramping and LR decay. Specifically, we decay the LR by $4\times$ and simultaneously ramp the BS by $4\times$ at 10B tokens. We compare this hybrid schedule with three baselines: at the same point, we (a) drops the LR by $4\times$, (b) drops the LR by $16\times$ and (c) ramps the BS by $16\times$. We denote the hybrid schedule by (d). In Figure 8 (right), three of the schedules (b c d) produce nearly identical loss curves. Crucially, these three configurations yield the same noise scale, since they preserve the ratio η/B. In contrast, schedule (a) results in a noticeably different trajectory.

In summary, our results reinforce our theoretical prediction: training dynamics in the late phase are governed primarily by the noise scale $\tau \propto \eta/B$. LR decay reduce the noise scale in the same manner as BS ramping, and any hybrid scheduler that preserves η/B will exhibit nearly identical behavior.

7 Conclusion and Limitations

In conclusion, we present a unified theoretical and empirical view of how local landscape geometry evolves during language model pre-training. Our analysis reveals two phases: an early sharp-to-flat transition and a late noise-governed regime. The early dynamics explain the necessity of LR warmup, suggesting that larger peak LRs require proportionally longer warmup lengths. The late regime shows that noise scale controls a trade-off between widening and deepening loss basin, motivating a BS scheduling that starts with small BS and increases the BS late in training.

Limitations. The current theory primarily relies on strong assumptions, such as infinite-small LR in SDE. A natural future direction is to generalize the theory to more realistic settings. Additionally, the current theory cannot fully explain the collapse of loss curves under different BS schedules. Understanding the learning dynamics under different BS schedules remains an open question.

REFERENCES

Shane Bergsma, Nolan Simran Dey, Gurpreet Gosal, Gavia Gray, Daria Soboleva, and Joel Hestness. Straight to zero: Why linearly decaying the learning rate to zero works best for LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025. 3

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 3

Huanran Chen, Yinpeng Dong, Zeming Wei, Yao Huang, Yichi Zhang, Hang Su, and Jun Zhu. Understanding pre-training and fine-tuning from loss landscape perspectives. *arXiv preprint arXiv:2505.17646*, 2025. 1

Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, et al. Symbolic discovery of optimization algorithms. *Advances in Neural Information Processing Systems*, 36, 2024. 20

Jeremy Cohen, Alex Damian, Ameet Talwalkar, J Zico Kolter, and Jason D. Lee. Understanding optimization in deep learning with central flows. In *The Thirteenth International Conference on Learning Representations*, 2025. 2

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. *International Conference on Learning Representations*, 2021. 1, 2, 3, 5

Jeremy M Cohen, Behrooz Ghorbani, Shankar Krishnan, Naman Agarwal, Sourabh Medapati, Michal Badura, Daniel Suo, David Cardoze, Zachary Nado, George E Dahl, et al. Adaptive gradient methods at the edge of stability. *arXiv preprint arXiv:2207.14484*, 2022. 2

Soham De, Abhay Yadav, David Jacobs, and Tom Goldstein. Automated inference with adaptive batches. In *Artificial Intelligence and Statistics*, pp. 1504–1513. PMLR, 2017. 3

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011. 3

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021. 1

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 *International Conference on Learning Representations*, 2022. 1, 3, 6

Akhilesh Gotmare, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation. In *International Conference on Learning Representations*, 2019. 3

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. 3

Gavia Gray, Anshul Samar, and Joel Hestness. Efficient and approximate per-example gradient norms for gradient noise scale. In *Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ NeurIPS 2023)*, 2023. 3

Gavia Gray, Shane Bergsma, Joel Hestness, et al. Normalization layer per-example gradients are sufficient to predict gradient noise scale in transformers. *Advances in Neural Information Processing Systems*, 37:93510–93539, 2024. 3

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. 3

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022. 3

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024. 1, 3, 4

Stanisław Jastrzębski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Three factors influencing minima in sgd. *arXiv preprint arXiv:1711.04623*, 2017. 6

Stanislaw Jastrzebski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho*, and Krzysztof Geras*. The break-even point on optimization trajectories of deep neural networks. In *International Conference on Learning Representations*, 2020. 2

Stanisław Jastrzębski, Zachary Kenton, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amost Storkey. On the relation between the sharpest directions of DNN loss and the SGD step length. In *International Conference on Learning Representations*, 2019. 2

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020. 1

Dayal Singh Kalra and Maissam Barkeshli. Why warmup the learning rate? underlying mechanisms and improvements. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 6

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. 3

Andrej Karpathy. NanoGPT. https://github.com/karpathy/nanoGPT, 2022. 20

Jordan Keller et al. Muon optimizer. https://github.com/KellerJordan/Muon?tab=readme-ov-file, 2024. 20

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017. 1

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3, 20

Atli Kosson, Bettina Messmer, and Martin Jaggi. Analyzing & reducing the need for learning rate warmup in GPT training. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 3

Tim Tsz-Kit Lau, Weijian Li, Chenwei Xu, Han Liu, and Mladen Kolar. Communication-efficient adaptive batch size strategies for distributed local gradient methods. *arXiv preprint arXiv:2406.13936*, 2024a. 3

Tim Tsz-Kit Lau, Han Liu, and Mladen Kolar. Adadagrad: Adaptive batch size schemes for adaptive gradient methods. *arXiv preprint arXiv:2402.11215*, 2024b. 3

Tim Tsz-Kit Lau, Weijian Li, Chenwei Xu, Han Liu, and Mladen Kolar. Adaptive batch size schedules for distributed training of language models with data and model parallelism. In *Proceedings of Conference on Parsimony and Learning*, 2025. 3

Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, et al. Minimax-01: Scaling foundation models with lightning attention. *arXiv preprint arXiv:2501.08313*, 2025. 3, 7

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 5

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024. 3

Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018. 3, 9

William Merrill, Shane Arora, Dirk Groeneveld, and Hannaneh Hajishirzi. Critical batch size revisited: A simple empirical approach to large-batch language model training. *arXiv preprint arXiv:2505.23971*, 2025. 7, 9

Petr Ostroukhov, Aigerim Zhumabayeva, Chulu Xiang, Alexander Gasnikov, Martin Takáč, and Dmitry Kamzolov. Adabatchgrad: Combining adaptive batch size and adaptive step size. *arXiv preprint arXiv:2402.05264*, 2024. 3

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. 3, 20

ShengYun Peng, Pin-Yu Chen, Matthew Daniel Hull, and Duen Horng Chau. Navigating the safety landscape: Measuring risks in finetuning large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 1

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 20

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019. 3

Minhak Song and Chulhee Yun. Trajectory alignment: understanding the edge of stability phenomenon via bifurcation theory. *arXiv preprint arXiv:2307.04204*, 2023. 2

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. 20

Tijmen Tieleman and Geoffrey Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. *COURSERA Neural Networks Mach. Learn*, 17:6, 2012. 3

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 3, 20

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, volume 30, 2017. 3

Jinbo Wang, Mingze Wang, Zhanpeng Zhou, Junchi Yan, Weinan E, and Lei Wu. The sharpness disparity principle in transformers for accelerating language model pre-training. In *Forty-second International Conference on Machine Learning*, 2025. 1, 23

Mingze Wang, Jinbo Wang, Haotian He, Zilin Wang, Guanhua Huang, Feiyu Xiong, Zhiyu li, Weinan E, and Lei Wu. Improving generalization and convergence by enhancing implicit regularization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 23

Kaiyue Wen, Zhiyuan Li, Jason Wang, David Hall, Percy Liang, and Tengyu Ma. Understanding warmup-stable-decay learning rates: A river valley loss landscape perspective. *arXiv preprint arXiv:2410.05192*, 2024. 1

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. 20

Lei Wu, Chao Ma, and Weinan E. How SGD selects the global minima in over-parameterized learning: A dynamical stability perspective. *Advances in Neural Information Processing Systems*, 31:8279–8288, 2018. 2, 3

Lei Wu, Mingze Wang, and Weijie J Su. The alignment property of SGD noise and how it helps select flat minima: A stability analysis. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. 2

Zeke Xie, Issei Sato, and Masashi Sugiyama. A diffusion theory for deep learning dynamics: Stochastic gradient descent exponentially favors flat minima. In *International Conference on Learning Representations*, 2021. 2

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017. 1

Hanlin Zhang, Depen Morwani, Nikhil Vyas, Jingfeng Wu, Difan Zou, Udaya Ghai, Dean Foster, and Sham Kakade. How does critical batch size scale in pre-training? *International Conference on Learning Representations*, 2025. 3

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022. 3

Yushun Zhang, Congliang Chen, Tian Ding, Ziniu Li, Ruoyu Sun, and Zhi-Quan Luo. Why transformers need adam: A hessian perspective. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024a. 1, 23

Yushun Zhang, Congliang Chen, Ziniu Li, Tian Ding, Chenwei Wu, Yinyu Ye, Zhi-Quan Luo, and Ruoyu Sun. Adam-mini: Use fewer learning rates to gain more. *arXiv preprint arXiv:2406.16793*, 2024b. 20

Zhanpeng Zhou, Mingze Wang, Yuchen Mao, Bingrui Li, and Junchi Yan. Sharpness-aware minimization efficiently selects flatter minima late in training. In *The Thirteenth International Conference on Learning Representations*, 2025. 2, 3

Zhanxing Zhu, Jingfeng Wu, Bing Yu, Lei Wu, and Jinwen Ma. The anisotropic noise in stochastic gradient descent: Its behavior of escaping from sharp minima and regularization effects. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7654–7663. PMLR, 09–15 Jun 2019. 2

A TERMINOLOGIES

Terminology	General Meaning	Usage in This Paper
Sharpness	A measure of curvature in the loss landscape, often characterized via the Hessian. Different works may define it differently.	We define sharpness as the curvature along the sharpest direction of the loss landscape. Mathematically, it is presented as the largest eigenvalue of the Hessian $\lambda_{\max}(\mathbf{H}(\boldsymbol{\theta}_t))$ or of the preconditioned curvature matrix $\lambda_{\max}(\mathbf{S}(\boldsymbol{\theta}_t))$.
Flat/sharp minimum	A minimum is a point where the gradient vanishes $\nabla L(\boldsymbol{\theta}) = 0$ and the loss does not decrease in a small neighborhood. A sharp minimum has large curvature; a flat minimum has small curvature.	We use these terms sparingly and follow the standard definitions from the sharpness/flat-minima literature.
Wide/deep basin	A loss basin is a region of the landscape surrounding a minimum. A wide basin rises loss slowly in most directions, whereas a deep basin has a significantly lower minimum value compared to its surroundings.	We use these terms to establish the depth–flatness trade-off: large noise scales tend to find wide basins, while small noise scales tend to find deeper regions with lower loss.

Table 1: Terminology and usage in this paper.

B BROADER LIMITATIONS

While our work provides a unified geometric view of pre-training dynamics, it is subject to several broader limitations.

Scale and data diversity. Our experiments use decoder-only Transformers up to 530M parameters, trained on subsets of FineWeb-Edu with at most 1000 tokens-per-parameter. While this range is realistic for many pre-training settings, it is still much smaller than the multi-billion-parameter, multi-trillion-token regimes used in frontier models. The behavior of the early sharp-to-flat transition and the later noise-dominated regime may look different at those larger scales due to factors such as stronger path-dependence, data-mixture effects, or very long context windows. In addition, all of our experiments use English web text. Other domains—such as code, multilingual data, speech, or vision–language corpora, may exhibit different curvature patterns or gradient-noise characteristics. Further work is needed to understand how our observations generalize across scales and data types.

Architectural coverage. Our experiments focus mainly on LLaMA-style models that use RMSNorm and RoPE, along with a smaller set of GPT-2–like models that use LayerNorm, GELU, and different depth/width configurations. We observe the same qualitative early sharp-to-flat behavior across these families, but we do not systematically vary architectural components such as attention mechanisms, normalization placement (pre-norm vs. post-norm), activation functions, parameter sharing, or mixture-of-experts routing. We also do not yet study how architectural changes, such as replacing RMSNorm with LayerNorm or altering normalization statistics, affect the depth–flatness trade-off or the recommended learning-rate and batch-size schedules. A more comprehensive architectural study is left for future work.

Optimizer and hyperparameter dependence. Most of our experiments fix a particular optimizer configuration with standard hyperparameters, and gradient clipping. However, adaptive methods maintain evolving state (e.g., moving averages of first and second moments) whose transient behavior interacts non-trivially with curvature. Therefore, verifying our findings in a more general hyperparameter space is an important direction for future research. Additionally, we only considered the WSD-like schedulers. Other schedulers, such as `cosine` scheduler should be considered as well.

C Missing Proof

C.1 Early in Pre-Training: Lyapunov Stability Analysis

Lemma C.1 (Stability Condition for Preconditioned GD). *Define the preconditioned curvature matrix $\mathbf{S}(\boldsymbol{\theta}_k) := \mathbf{M}^{1/2}\mathbf{H}(\boldsymbol{\theta}_k)\mathbf{M}^{1/2}$, and let $\{\lambda\}_{i=1}^{p}$ be the eigenvalues of $\mathbf{S}(\boldsymbol{\theta}_k)$. The linear system in Equation (4) is asymptotically stable (i.e., $\lim_{k\to\infty} \boldsymbol{e}_k = \mathbf{0}$) if η satisfies*

$$0 < \eta < \frac{2}{\lambda_{\max}(\mathbf{S}(\boldsymbol{\theta}_k)}, \forall k \geq 0. \tag{6}$$

Proof. Since $\boldsymbol{e}_{k+1} = (\mathbf{I} - \eta\mathbf{M}\mathbf{H}_k)\boldsymbol{e}_k$, the linear system is asymptotically stable if all eigenvalues of $\mathbf{I} - \eta\mathbf{M}\mathbf{H}_k$ have magnitude less than 1. Note that:

$$\mathbf{I} - \eta\mathbf{M}\mathbf{H}_k = \mathbf{M}^{1/2}(\mathbf{I} - \eta\mathbf{S}_k)\mathbf{M}^{-1/2}, \tag{7}$$

so the eigenvalues are $1 - \eta\lambda_j(\mathbf{S}_k)$. The stability condition $|1 - \eta\lambda_j| < 1$ for all j is equivalent to:

$$0 < \eta < \frac{2}{\lambda_{\max}(\mathbf{S}_k)}. \tag{8}$$

□

Lemma C.2 (Exact one-step loss change). *Define:*

$$\mathbf{S}(\boldsymbol{\theta}) := \mathbf{M}^{1/2}\mathbf{H}(\boldsymbol{\theta})\mathbf{M}^{1/2},$$
$$\boldsymbol{g}_k := \mathbf{M}^{1/2}\nabla L(\boldsymbol{\theta}_k),$$
$$\boldsymbol{\delta}_k := \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k = -\eta\mathbf{M}\nabla L(\boldsymbol{\theta}_k).$$

Then the true loss change can be written exactly as

$$L(\boldsymbol{\theta}_{k+1}) - L(\boldsymbol{\theta}_k) = -\eta\|\nabla L(\boldsymbol{\theta}_k)\|_{\mathbf{M}}^2 + \eta^2 \int_0^1 (1-t)\, \boldsymbol{g}_k^\top \mathbf{S}(\boldsymbol{\theta}_k + t\boldsymbol{\delta}_k)\boldsymbol{g}_k \, dt, \tag{9}$$

where $\|\nabla L(\boldsymbol{\theta}_k)\|_{\mathbf{M}}^2 := (\nabla L(\boldsymbol{\theta}_k))^\top \mathbf{M}\nabla L(\boldsymbol{\theta}_k)$.

Proof. Let $\boldsymbol{\delta}_k := \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k = -\eta\mathbf{M}\nabla L(\boldsymbol{\theta}_k)$ and define the scalar function

$$\phi(t) := L(\boldsymbol{\theta}_k + t\boldsymbol{\delta}_k), \quad t \in [0, 1].$$

Then

$$L(\boldsymbol{\theta}_{k+1}) - L(\boldsymbol{\theta}_k) = \phi(1) - \phi(0).$$

Compute the derivatives:

$$\phi'(t) = (\nabla L(\boldsymbol{\theta}_k + t\boldsymbol{\delta}_k))^\top \boldsymbol{\delta}_k,$$
$$\phi''(t) = \boldsymbol{\delta}_k^\top \mathbf{H}(\boldsymbol{\theta}_k + t\boldsymbol{\delta}_k)\boldsymbol{\delta}_k.$$

By Taylor's theorem with integral remainder:

$$\phi(1) - \phi(0) = \phi'(0) + \int_0^1 (1-t)\phi''(t)\, dt.$$

Now evaluate at $t = 0$:

$$\phi'(0) = (\nabla L(\boldsymbol{\theta}_k))^\top \boldsymbol{\delta}_k = -\eta(\nabla L(\boldsymbol{\theta}_k))^\top \mathbf{M}\nabla L(\boldsymbol{\theta}_k) = -\eta\|\nabla L(\boldsymbol{\theta}_k)\|_{\mathbf{M}}^2.$$

For the second derivative term:

$$\phi''(t) = \boldsymbol{\delta}_k^\top \mathbf{H}(\boldsymbol{\theta}_k + t\boldsymbol{\delta}_k)\boldsymbol{\delta}_k = \eta^2 \boldsymbol{g}_k^\top \mathbf{S}(\boldsymbol{\theta}_k + t\boldsymbol{\delta}_k)\boldsymbol{g}_k,$$

since $\boldsymbol{\delta}_k = -\eta\mathbf{M}\nabla L(\boldsymbol{\theta}_k)$ and $\boldsymbol{g}_k = \mathbf{M}^{1/2}\nabla L(\boldsymbol{\theta}_k)$, and thus

$$\boldsymbol{\delta}_k^\top \mathbf{H}(\cdot)\boldsymbol{\delta}_k = \eta^2 \boldsymbol{g}_k^\top \mathbf{S}(\cdot)\boldsymbol{g}_k.$$

Substituting both terms yields the result.

□

Lemma C.3 (One-step Loss Change). *Let $\boldsymbol{\delta}_k := \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k$. Suppose that along the segment $\{\boldsymbol{\theta}_k + \alpha\boldsymbol{\delta}_k : \alpha \in [0,1]\}$, we have $0 \le \lambda_{\min}(\mathbf{S}(\boldsymbol{\theta}_k + \alpha\boldsymbol{\delta}_k)) \le \lambda_{\max}(\mathbf{S}(\boldsymbol{\theta}_k + \alpha\boldsymbol{\delta}_k)) \le \Lambda_k$. Then,*

$$L(\boldsymbol{\theta}_{k+1}) - L(\boldsymbol{\theta}_k) \le -\eta(1 - \frac{1}{2}\eta\Lambda_k)(\nabla L(\boldsymbol{\theta}_k))^\top \mathbf{M} \nabla L(\boldsymbol{\theta}_k).$$

In particular, if $\eta \le 2/\Lambda_k$, each update is guaranteed to non-increasing in loss, i.e., $L(\boldsymbol{\theta}_{k+1}) \le L(\boldsymbol{\theta}_k)$. Instead, if $\eta \uparrow 2/\Lambda_k$, the guaranteed decrease per step $\boxed{(L(\boldsymbol{\theta}_k) - L(\boldsymbol{\theta}_{k+1}))/\eta \to 0}$.

Proof. From Lemma C.2, we have

$$\boldsymbol{g}_k^\top \mathbf{S}(\boldsymbol{\theta}_k + t\boldsymbol{\delta}_k)\boldsymbol{g}_k \le \Lambda_k \|\boldsymbol{g}_k\|^2$$

for all t, since $\mathbf{S}(\cdot)$ is symmetric. Therefore,

$$\begin{aligned} L(\boldsymbol{\theta}_{k+1}) - L(\boldsymbol{\theta}_k) &\le -\eta\|\nabla L(\boldsymbol{\theta}_k)\|_{\mathbf{M}}^2 + \eta^2\Lambda_k\|\boldsymbol{g}_k\|^2 \int_0^1 (1-t)\,dt \\ &= -\eta\|\nabla L(\boldsymbol{\theta}_k)\|_{\mathbf{M}}^2 + \tfrac{1}{2}\eta^2\Lambda_k\|\boldsymbol{g}_k\|^2. \end{aligned}$$

Note that $\|\boldsymbol{g}_k\|^2 = \|\nabla L(\boldsymbol{\theta}_k)\|_{\mathbf{M}}^2$, yielding the result.

□

C.2 Late in Pre-Training: SDE Analysis

C.2.1 Discrete-Time Solution

Lemma C.4 (Eigenbasis Decomposition). *Let $\mathbf{S} := \mathbf{M}^{1/2}\mathbf{H}(\boldsymbol{\theta}^\star)\mathbf{M}^{1/2}$ with eigendecomposition $\mathbf{S} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top$, $\boldsymbol{\Lambda} = diag(\lambda_1, \dots, \lambda_d)$. Define $\mathbf{G} := \mathbf{Q}^\top\mathbf{M}^{1/2}\boldsymbol{\Sigma}(\boldsymbol{\theta}^\star)\mathbf{M}^{1/2}\mathbf{Q}/B$. In coordinates $\boldsymbol{w}_k := \mathbf{Q}^\top\mathbf{M}^{-1/2}\boldsymbol{e}_k$, the recursion gives:*

$$\boldsymbol{w}_{k+1} = (\mathbf{I} - \eta\boldsymbol{\Lambda})\boldsymbol{w}_k + \eta\boldsymbol{\zeta}_k, \quad \mathbb{E}[\boldsymbol{\zeta}_k\boldsymbol{\zeta}_k^\top] = \mathbf{G}$$

The stationary covariance $\boldsymbol{\Sigma}_w$ has diagonal elements:

$$(\boldsymbol{\Sigma}_w)_{jj} = \frac{\eta^2\mathbf{G}_{jj}}{1 - (1-\eta\lambda_j)^2} = \frac{\eta\mathbf{G}_{jj}}{2\lambda_j - \eta\lambda_j^2} \tag{10}$$

Proof. First, we verify that $\mathbf{S} = \mathbf{M}^{1/2}\mathbf{H}(\boldsymbol{\theta}^\star)\mathbf{M}^{1/2}$ can be eigendecomposed. Since both $\mathbf{M}$ and $\mathbf{H}(\boldsymbol{\theta}^\star)$ are positive definite matrices, $\mathbf{S}$ is also positive definite matrix. By the spectral theorem, $\mathbf{S}$ admits the eigendecomposition $\mathbf{S} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top$, where $\mathbf{Q}$ is orthogonal and $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \dots, \lambda_d)$ with $\lambda_i > 0$.

Starting from $\boldsymbol{e}_{k+1} = \mathbf{A}\boldsymbol{e}_k + \eta\mathbf{M}\boldsymbol{\xi}_k$ with $\mathbf{A} = \mathbf{I} - \eta\mathbf{M}\mathbf{H}(\boldsymbol{\theta}^\star)$, we change variables to $\boldsymbol{w}_k = \mathbf{Q}^\top\mathbf{M}^{-1/2}\boldsymbol{e}_k$.

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \mathbf{Q}^\top\mathbf{M}^{-1/2}\boldsymbol{e}_{k+1} = \mathbf{Q}^\top\mathbf{M}^{-1/2}(\mathbf{A}\boldsymbol{e}_k + \eta\mathbf{M}\boldsymbol{\xi}_k) \\ &= \mathbf{Q}^\top\mathbf{M}^{-1/2}(\mathbf{I} - \eta\mathbf{M}\mathbf{H}(\boldsymbol{\theta}^\star))\boldsymbol{e}_k + \eta\mathbf{Q}^\top\mathbf{M}^{1/2}\boldsymbol{\xi}_k \\ &= \mathbf{Q}^\top\mathbf{M}^{-1/2}\boldsymbol{e}_k - \eta\mathbf{Q}^\top\mathbf{M}^{1/2}\mathbf{H}(\boldsymbol{\theta}^\star)\boldsymbol{e}_k + \eta\mathbf{Q}^\top\mathbf{M}^{1/2}\boldsymbol{\xi}_k \\ &= \boldsymbol{w}_k - \eta\mathbf{Q}^\top\mathbf{M}^{1/2}\mathbf{H}(\boldsymbol{\theta}^\star)\mathbf{M}^{1/2}\mathbf{Q}\boldsymbol{w}_k + \eta\mathbf{Q}^\top\mathbf{M}^{1/2}\boldsymbol{\xi}_k \\ &= \boldsymbol{w}_k - \eta\mathbf{Q}^\top\mathbf{S}\mathbf{Q}\boldsymbol{w}_k + \eta\mathbf{Q}^\top\mathbf{M}^{1/2}\boldsymbol{\xi}_k \\ &= (\mathbf{I} - \eta\boldsymbol{\Lambda})\boldsymbol{w}_k + \eta\mathbf{Q}^\top\mathbf{M}^{1/2}\boldsymbol{\xi}_k \end{aligned}$$

Defining $\boldsymbol{\zeta}_k := \mathbf{Q}^\top\mathbf{M}^{1/2}\boldsymbol{\xi}_k$, we get:

$$\mathbb{E}[\boldsymbol{\zeta}_k\boldsymbol{\zeta}_k^\top] = \mathbf{Q}^\top\mathbf{M}^{1/2}\mathbb{E}[\boldsymbol{\xi}_k\boldsymbol{\xi}_k^\top]\mathbf{M}^{1/2}\mathbf{Q} = \mathbf{Q}^\top\mathbf{M}^{1/2}\boldsymbol{\Sigma}(\boldsymbol{\theta}^\star)\mathbf{M}^{1/2}\mathbf{Q}/B =: \mathbf{G}$$

As matrix $(\mathbf{I} - \eta\mathbf{\Lambda})$ is diagonal, the recursion now decouples into independent scalar equations for each component j:

$$(\boldsymbol{w}_{k+1})_j = (1 - \eta\lambda_j)(\boldsymbol{w}_k)_j + \eta(\boldsymbol{\zeta}_k)_j.$$

For each component j, the stationary variance satisfies:

$$(\mathbf{\Sigma}_w)_{jj} = (1 - \eta\lambda_j)^2(\mathbf{\Sigma}_w)_{jj} + \eta^2\mathbf{G}_{jj} \tag{11}$$

Solving for $(\mathbf{\Sigma}_w)_{jj}$:

$$\begin{aligned}(\mathbf{\Sigma}_w)_{jj} &= \frac{\eta^2\mathbf{G}_{jj}}{1 - (1 - \eta\lambda_j)^2} = \frac{\eta^2\mathbf{G}_{jj}}{1 - (1 - 2\eta\lambda_j + \eta^2\lambda_j^2)} \\ &= \frac{\eta^2\mathbf{G}_{jj}}{2\eta\lambda_j - \eta^2\lambda_j^2} = \frac{\eta\mathbf{G}_{jj}}{2\lambda_j - \eta\lambda_j^2}\end{aligned}$$

□

C.2.2 Continuous-Time Limit

We now take the continuous-time limit ($\eta \to 0$) to derive a simpler universal theory. The exact solution for the variance in the eigenbasis from Lemma C.4, i.e., $(\mathbf{\Sigma}_w)_{jj} = \eta\mathbf{G}_{jj}/(2\lambda_j - \eta\lambda_j^2)$, guides the necessary scaling for the continuous-time limit. Because $(\mathbf{\Sigma}_w)_{jj}$ converges to a finite non-zero value as $\eta \to 0$, the numerator $\eta\mathbf{G}_{jj}$ must remain finite. This suggests defining a quantity τ such that for each mode j:

$$\eta\,\mathbf{G}_{jj} \to 2\tau \quad \text{as} \quad \eta \to 0.$$

We strengthen this to :

$$\eta\,\mathbf{G} \to 2\tau\mathbf{I} \quad \text{as} \quad \eta \to 0.$$

Recalling that $\mathbf{G} = \mathbf{Q}^\top\mathbf{M}^{1/2}\mathbf{\Sigma}(\boldsymbol{\theta}^\star)\mathbf{M}^{1/2}\mathbf{Q}/B$, this condition in the original coordinate system translates to the required scaling for the noise covariance:

$$\frac{\eta}{B}\mathbf{M}\mathbf{\Sigma}(\boldsymbol{\theta}^\star)\mathbf{M}^\top \to 2\tau\mathbf{M}.$$

Proposition C.1 (Convergence to SDE)**.** *Consider the scaled discrete process $\boldsymbol{\theta}_{\lfloor t/\eta \rfloor}$ as $\eta \to 0$. Suppose the noise covariance satisfies*

$$\frac{\eta}{B}\mathbf{M}\mathbf{\Sigma}(\boldsymbol{\theta}^\star)\mathbf{M}^\top = 2\tau\mathbf{M} + O(\eta), \tag{12}$$

for some temperature $\tau > 0$. Then the process converges weakly to the Itô SDE:

$$d\boldsymbol{\theta}_t = -\mathbf{M}\nabla L(\boldsymbol{\theta}_t)dt + \sqrt{2\tau}\mathbf{M}^{1/2}dW_t \tag{13}$$

where W_t is standard Brownian motion.

Proof. Consider the discrete preconditioned SGD update:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta\mathbf{M}(\nabla L(\boldsymbol{\theta}_k) + \boldsymbol{\xi}_k),$$

Define the scaled process $\boldsymbol{\theta}^{(\eta)}(t) = \boldsymbol{\theta}_{\lfloor t/\eta \rfloor}$. The increment $\Delta\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k$ satisfies:

$$\mathbb{E}[\Delta\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_k = \boldsymbol{\theta}] = -\eta\mathbf{M}\nabla L(\boldsymbol{\theta}),$$

$$\mathrm{Cov}(\Delta\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_k = \boldsymbol{\theta}) = \frac{\eta^2}{B}\mathbf{M}\mathbf{\Sigma}(\boldsymbol{\theta}^\star)\mathbf{M}^\top.$$

Given the scaling condition Equation (12), the covariance is $O(\eta)$.

The generator $\mathcal{L}^{(\eta)}$ of the discrete process for a smooth function $\boldsymbol{f}$ is:

$$\mathcal{L}^{(\eta)}\boldsymbol{f}(\boldsymbol{\theta}) = \frac{1}{\eta}\mathbb{E}[\boldsymbol{f}(\boldsymbol{\theta}_{k+1}) - \boldsymbol{f}(\boldsymbol{\theta}_k) \mid \boldsymbol{\theta}_k = \boldsymbol{\theta}].$$

Using a Taylor expansion and taking conditional expectation:

$$\mathbb{E}[\boldsymbol{f}(\boldsymbol{\theta}_{k+1}) - \boldsymbol{f}(\boldsymbol{\theta}_k) \mid \boldsymbol{\theta}] = -\eta \nabla \boldsymbol{f}(\boldsymbol{\theta})^\top \mathbf{M} \nabla L(\boldsymbol{\theta}) + \frac{1}{2}\mathbb{E}[(\Delta\boldsymbol{\theta})^\top \nabla^2 \boldsymbol{f}(\boldsymbol{\theta}) \Delta\boldsymbol{\theta}] + O(\eta^{3/2}).$$

For the second term, with $\Delta\boldsymbol{\theta} = -\eta M(\nabla L(\boldsymbol{\theta}) + \boldsymbol{\xi}_k)$:

$$\begin{aligned}
\mathbb{E}[(\Delta\boldsymbol{\theta})^\top \nabla^2 \boldsymbol{f}(\boldsymbol{\theta}) \Delta\boldsymbol{\theta}] &= \eta^2 \mathbb{E}[(\nabla L(\boldsymbol{\theta}) + \boldsymbol{\xi}_k)^\top \mathbf{M}^\top \nabla^2 \boldsymbol{f}(\boldsymbol{\theta}) \mathbf{M} (\nabla L(\boldsymbol{\theta}) + \boldsymbol{\xi}_k)] \\
&= \eta^2 \mathbb{E}[\boldsymbol{\xi}_k^\top \mathbf{M}^\top \nabla^2 \boldsymbol{f}(\boldsymbol{\theta}) \mathbf{M} \boldsymbol{\xi}_k] + O(\eta^2) \\
&= \eta^2 \mathrm{Tr}(\mathbf{M}^\top \nabla^2 \boldsymbol{f}(\boldsymbol{\theta}) \mathbf{M} \mathbb{E}[\boldsymbol{\xi}_k \boldsymbol{\xi}_k^\top]) + O(\eta^2) \\
&= \frac{\eta^2}{B} \mathrm{Tr}(\mathbf{M}^\top \nabla^2 \boldsymbol{f}(\boldsymbol{\theta}) \mathbf{M} \boldsymbol{\Sigma}(\boldsymbol{\theta}^\star)) + O(\eta^2) \\
&= \frac{\eta^2}{B} \mathrm{Tr}(\mathbf{M} \boldsymbol{\Sigma}(\boldsymbol{\theta}^\star) \mathbf{M}^\top \nabla^2 \boldsymbol{f}(\boldsymbol{\theta})) + O(\eta^2)
\end{aligned}$$

where we used $\mathbb{E}[\boldsymbol{\xi}^\top \mathbf{A} \boldsymbol{\xi}] = \mathrm{Tr}(A\mathbb{E}[\boldsymbol{\xi}\boldsymbol{\xi}^\top])$ and trace cyclicity $\mathrm{Tr}(\mathbf{ABC}) = \mathrm{Tr}(\mathbf{CAB})$.

Therefore:

$$\frac{1}{2}\mathbb{E}[(\Delta\boldsymbol{\theta})^\top \nabla^2 \boldsymbol{f}(\boldsymbol{\theta}) \Delta\boldsymbol{\theta}] = \frac{\eta^2}{2B} \mathrm{Tr}(\mathbf{M} \boldsymbol{\Sigma}(\boldsymbol{\theta}^\star) \mathbf{M}^\top \nabla^2 \boldsymbol{f}(\boldsymbol{\theta})) + O(\eta^2).$$

Using the scaling condition Equation (12), we have :

$$\frac{\eta^2}{2B} \mathrm{Tr}(\mathbf{M} \boldsymbol{\Sigma}(\boldsymbol{\theta}^\star) \mathbf{M}^\top \nabla^2 \boldsymbol{f}(\boldsymbol{\theta})) = \frac{\eta}{2} \mathrm{Tr}\left(2\tau \mathbf{M} \nabla^2 \boldsymbol{f}(\boldsymbol{\theta})\right) + O(\eta^2) = \eta\tau \mathrm{Tr}(\mathbf{M} \nabla^2 \boldsymbol{f}(\boldsymbol{\theta})) + O(\eta^2).$$

Thus,

$$\mathcal{L}^{(\eta)} \boldsymbol{f}(\boldsymbol{\theta}) = -\nabla \boldsymbol{f}(\boldsymbol{\theta})^\top \mathbf{M} \nabla L(\boldsymbol{\theta}) + \tau \mathrm{Tr}(\mathbf{M} \nabla^2 \boldsymbol{f}(\boldsymbol{\theta})) + O(\eta).$$

As $\eta \to 0$, $\mathcal{L}^{(\eta)} \boldsymbol{f}(\boldsymbol{\theta})$ converges to:

$$\mathcal{L} \boldsymbol{f}(\boldsymbol{\theta}) = -\nabla \boldsymbol{f}(\boldsymbol{\theta})^\top \mathbf{M} \nabla L(\boldsymbol{\theta}) + \tau \mathrm{Tr}(\mathbf{M} \nabla^2 \boldsymbol{f}(\boldsymbol{\theta})),$$

which is the generator of the Itô SDE:

$$d\boldsymbol{\theta}_t = -\mathbf{M} \nabla L(\boldsymbol{\theta}_t) dt + \sqrt{2\tau} \mathbf{M}^{1/2} dW_t.$$

By the weak convergence theory (e.g., via the martingale problem or generator convergence), the process $\boldsymbol{\theta}^{(\eta)}(t)$ converges weakly to the solution of this SDE. □

Proposition C.2 (Gibbs Stationary Distribution)**.** *The SDE in Equation* (13) *has stationary distribution:*

$$p_\infty(\boldsymbol{\theta}) \propto \exp(-L(\boldsymbol{\theta})/\tau) \tag{14}$$

Proof. The generator of the SDE (5) is $\mathcal{L} f = -\mathbf{M} \nabla L \cdot \nabla f + \tau \mathrm{tr}(\mathbf{M} \nabla^2 f)$. The Fokker-Planck equation for the probability density $p(t, \boldsymbol{\theta})$ is:

$$\partial_t p = \mathcal{L}^* p = \nabla \cdot (\mathbf{M} \nabla L \, p) + \tau \nabla \cdot (\mathbf{M} \nabla p)$$

where $\mathcal{L}^*$ is the adjoint operator. Setting $\partial_t p = 0$ for stationarity:

$$\begin{aligned}
0 &= \nabla \cdot (\mathbf{M} \nabla L \, p_\infty) + \tau \nabla \cdot (\mathbf{M} \nabla p_\infty) \\
&= \nabla \cdot (\mathbf{M} \nabla L \, p_\infty + \tau \mathbf{M} \nabla p_\infty)
\end{aligned}$$

This implies the current $J = \mathbf{M} \nabla L \, p_\infty + \tau \mathbf{M} \nabla p_\infty$ has zero divergence. For a potential-driven system, we require $J = \mathbf{0}$:

$$\begin{aligned}
\mathbf{M} \nabla L \, p_\infty + \tau \mathbf{M} \nabla p_\infty &= \mathbf{0} \\
\nabla L \, p_\infty + \tau \nabla p_\infty &= \mathbf{0} \quad \text{(since } \mathbf{M} \succ 0\text{)} \\
\frac{\nabla p_\infty}{p_\infty} &= -\frac{\nabla L}{\tau}
\end{aligned}$$

Integrating: $\log p_\infty = -L/\tau + \text{const}$, which gives Equation (14). □

Theorem C.1 (Free Energy Minimization)**.** *Let the empirical risk $L(\boldsymbol{\theta})$ admit multiple local minima $\{\boldsymbol{\theta}_i^\star\}_{i=1}^m$ with Hessians $\mathbf{H}(\boldsymbol{\theta}_i^\star) \succ 0$. Under the SDE in Equation (13) with temperature τ, the stationary probability that training resides in basin i is given by:*

$$P_\tau(\textit{basin } i) = \frac{\exp(-F_i(\tau)/\tau)}{\sum_j \exp(-F_j(\tau)/\tau)}, \quad F_i(\tau) := L(\boldsymbol{\theta}_i^\star) + \frac{\tau}{2}\log\det \mathbf{H}(\boldsymbol{\theta}_i^\star). \tag{15}$$

Proof. From the Gibbs distribution Equation (14), the probability mass in basin i is:

$$P_\tau(\text{basin } i) = \frac{\int_{B_i} e^{-L(\boldsymbol{\theta})/\tau} d\boldsymbol{\theta}}{\int_{\mathbb{R}^d} e^{-L(\boldsymbol{\theta})/\tau} d\boldsymbol{\theta}}$$

where B_i is the basin of attraction around minimum $\boldsymbol{\theta}_i^\star$.

For the numerator, using the quadratic approximation $L(\boldsymbol{\theta}) = L(\boldsymbol{\theta}_i^\star) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_i^\star)^\top \mathbf{H}(\boldsymbol{\theta}_i^\star)(\boldsymbol{\theta} - \boldsymbol{\theta}_i^\star)$ in basin i we get:

$$\begin{aligned}\int_{B_i} e^{-L(\boldsymbol{\theta})/\tau} d\boldsymbol{\theta} &= \int_{\mathbb{R}^d} \exp\left(-\frac{L(\boldsymbol{\theta}_i^\star)}{\tau} - \frac{1}{2\tau}(\boldsymbol{\theta} - \boldsymbol{\theta}_i^\star)^\top \mathbf{H}(\boldsymbol{\theta}_i^\star)(\boldsymbol{\theta} - \boldsymbol{\theta}_i^\star)\right) d\boldsymbol{\theta} \\ &= e^{-L(\boldsymbol{\theta}_i^\star)/\tau} \int_{\mathbb{R}^d} \exp\left(-\frac{1}{2\tau}(\boldsymbol{\theta} - \boldsymbol{\theta}_i^\star)^\top \mathbf{H}(\boldsymbol{\theta}_i^\star)(\boldsymbol{\theta} - \boldsymbol{\theta}_i^\star)\right) d\boldsymbol{\theta}\end{aligned}$$

The integral is a multivariate Gaussian with covariance $\tau\mathbf{H}(\boldsymbol{\theta}_i^\star)^{-1}$. Using the standard formula for Gaussian integrals:

$$\int_{\mathbb{R}^d} \exp\left(-\frac{1}{2} y^\top \boldsymbol{\Sigma}^{-1} y\right) dy = (2\pi)^{d/2} (\det \boldsymbol{\Sigma})^{1/2}$$

With $\boldsymbol{\Sigma} = \tau\mathbf{H}(\boldsymbol{\theta}_i^\star)^{-1}$, we have $\det \boldsymbol{\Sigma} = \tau^d (\det \mathbf{H}(\boldsymbol{\theta}_i^\star))^{-1}$ and $\boldsymbol{\Sigma}^{-1} = \tau^{-1}\mathbf{H}(\boldsymbol{\theta}_i^\star)$:

$$\begin{aligned}\int_{\mathbb{R}^d} \exp\left(-\frac{1}{2\tau}(\boldsymbol{\theta} - \boldsymbol{\theta}_i^\star)^\top \mathbf{H}(\boldsymbol{\theta}_i^\star)(\boldsymbol{\theta} - \boldsymbol{\theta}_i^\star)\right) d\boldsymbol{\theta} &= (2\pi)^{d/2} (\tau^d (\det \mathbf{H}(\boldsymbol{\theta}_i^\star))^{-1})^{1/2} \\ &= (2\pi\tau)^{d/2} (\det \mathbf{H}(\boldsymbol{\theta}_i^\star))^{-1/2}\end{aligned}$$

Therefore:

$$\begin{aligned}\int_{B_i} e^{-L(\boldsymbol{\theta})/\tau} d\boldsymbol{\theta} &= e^{-L(\boldsymbol{\theta}_i^\star)/\tau} (2\pi\tau)^{d/2} (\det \mathbf{H}(\boldsymbol{\theta}_i^\star))^{-1/2} \\ &= (2\pi\tau)^{d/2} \exp\left(-L(\boldsymbol{\theta}_i^\star)/\tau - \frac{1}{2}\log\det \mathbf{H}(\boldsymbol{\theta}_i^\star)\right) \\ &= (2\pi\tau)^{d/2} \exp\left(-\frac{1}{\tau}\left(L(\boldsymbol{\theta}_i^\star) + \frac{\tau}{2}\log\det \mathbf{H}(\boldsymbol{\theta}_i^\star)\right)\right) \\ &= (2\pi\tau)^{d/2} \exp(-F_i(\tau)/\tau)\end{aligned}$$

Similarly, the total partition function is:

$$\begin{aligned}Z(\tau) = \int_{\mathbb{R}^d} e^{-L(\boldsymbol{\theta})/\tau} d\boldsymbol{\theta} &= d\sum_{j=1}^m \int_{B_j} e^{-L(\boldsymbol{\theta})/\tau} d\boldsymbol{\theta} \\ &= (2\pi\tau)^{d/2} \sum_{j=1}^m \exp(-F_j(\tau)/\tau)\end{aligned}$$

Therefore:

$$P_\tau(\text{basin } i) = \frac{(2\pi\tau)^{d/2} \exp(-F_i(\tau)/\tau)}{(2\pi\tau)^{d/2} \sum_j \exp(-F_j(\tau)/\tau)} = \frac{\exp(-F_i(\tau)/\tau)}{\sum_j \exp(-F_j(\tau)/\tau)}$$

This completes the proof of the free energy formula Equation (15). □

D EXPERIMENTAL SETUPS AND MORE RESULTS

D.1 EXPERIMENTAL SETUPS

Models. We utilize two popular classes of LLM models for our pre-training experiments:

- **GPT-2.** We use GPT-2 (small) model (Radford et al., 2019), implemented via the nanoGPT code base (Karpathy, 2022). Following nanoGPT, the model employs Gaussian Error Linear Unit (GELU) activations and standard Layer Normalization (LayerNorm). Detailed model configurations are provided in Table 2.
- **LLaMA.** LLaMA (Touvron et al., 2023) is another popular decoder-only Transformer architecture, incorporating Rotary Positional Encoding (RoPE) (Su et al., 2024), Swish-Gated Linear Unit (SwiGLU), and Root mean square layer normalization (RMSNorm). For implementation, we utilize the LLaMA code from HuggingFace Transformers Library (Wolf et al., 2020). Additional model configurations are detailed in Table 2.

Datasets. Training is performed on the `FineWeb-Edu` dataset (Penedo et al., 2024). We adopt the a subset randomly sampled from the whole dataset of around 100B `GPT-2` tokens. The same dataset has been widely used in literature on LLM pre-training.

Optimizers. To generalize our findings across different optimizers, we choose:

- **AdamW.** AdamW (Kingma & Ba, 2014) is adopted with hyperparameters $\beta_1 = 0.95$, $\beta_2 = 0.95$, and weight decay 0.1.
- **Muon.** Muon (Keller et al., 2024) is used with momentum of 0.95 and weight decay 0.1.
- **Adam-mini.** The hyperparameter of Adam-mini (Zhang et al., 2024b) is the same as AdamW.
- **Lion.** Lion (Chen et al., 2024) is used with hyperparameters $\beta_1 = 0.95$, $\beta_2 = 0.98$. The LR of Lion η is divided by $10\times$ compared with the LR of AdamW in the same experiments, and the weight decay λ is ramped up to $10\times$ to keep the effective LR $\lambda\eta = 0.1$.

All these optimizers are used with gradient clipping at 1.0 for stability.

Table 2: Model configurations.

Acronym	Size	d_{model}	d_{FF}	n_head	depth
GPT-2 (small)	124M	768	3072	12	12
LLaMA (93M)	93M	512	2048	16	8
LLaMA (170M)	170M	768	3072	12	8
LLaMA (270M)	270M	1024	4096	16	8
LLaMA (530M)	530M	1536	6144	24	8

D.2 MORE RESULTS UNDER VARIOUS SETUPS.

In this section, we extend our findings to other architectures, optimization algorithms, and larger training scales. Due to computational constraints, we primarily focus on validating the sharp-to-flat early dynamics and the proposed BS scheduling principle across these settings. We also explore the warmup–tuning recipe on additional architectures.

Extension to `GPT-2` Architectures. See Figures 9 and 10 for details.

Extension to Other Optimizers. See Figure 11 for Adam-mini, and see Figure 12 for Lion.

Extension to Larger Models. See Figure 13 for models with 270M and 530M parameters.

D.3 ABLATION STUDIES ON EARLY INSTABILITIES.

In this section, we conduct ablation studies on the root cause of instabilities, such as loss spikes and plateaus, observed in early training.

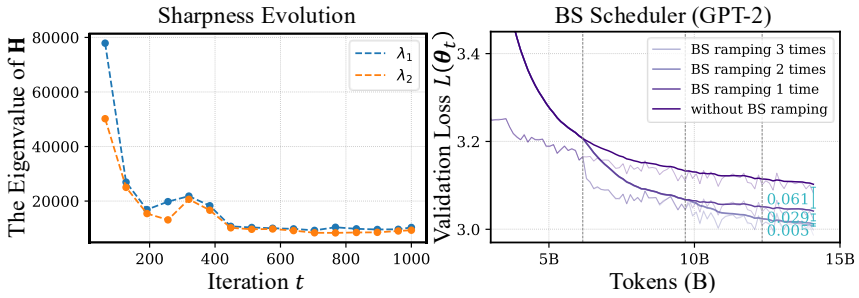

Figure 9: **Extensive to GPT-2 architectures. (Left). Early sharp-to-flat dynamics.** Evolution of the top eigenvalues of the Hessian across iterations: $\lambda_i(\mathbf{H}(\boldsymbol{\theta}_t))$ vs. iteration t. **(Right). BS scheduling improves data efficiency. BS scheduling improves data efficiency.** We use a BS schedule that starts at 0.49M and increases by $4\times$ at each ramp. Models are trained with $1, 2$, or 3 ramping steps, while models without ramping serve as the baseline. Vertical gray dashed lines indicate ramping positions.

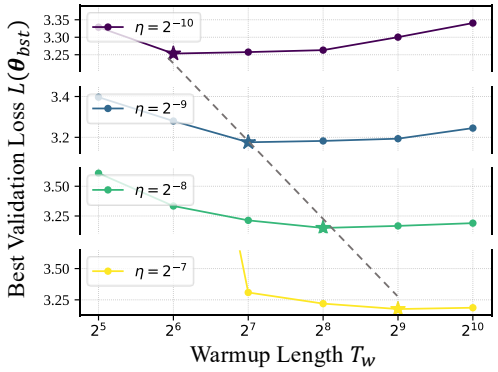

Figure 10: **Extensive to GPT-2 architectures. Larger Peak LR, Longer Warmup.** We train a series of GPT-2 models with 100 TPP. We vary the peak LRs η and warmup lengths T_w. We plot the best validation loss $L(\boldsymbol{\theta}_{\text{bst}})$ vs. T_w for different η. The optimal T_w is highlighted with a star.

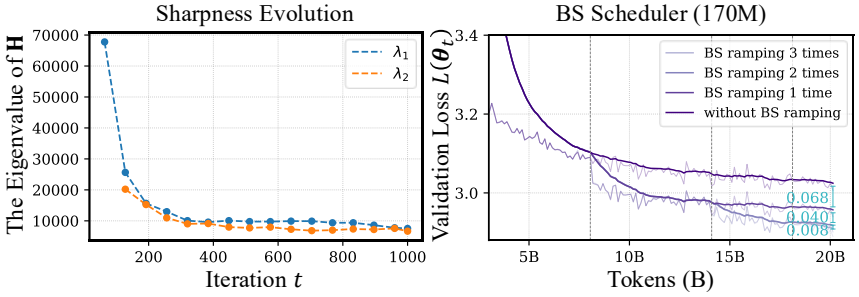

Figure 11: **Extensive to Adam-mini optimizer. (Left). Early sharp-to-flat dynamics.** Evolution of the top eigenvalues of the Hessian across iterations: $\lambda_i(\mathbf{H}(\boldsymbol{\theta}_t))$ vs. iteration t. **(Right). BS scheduling improves data efficiency. BS scheduling improves data efficiency.** We use a BS schedule that starts at 0.49M and increases by $4\times$ at each ramp. Models are trained with $1, 2$, or 3 ramping steps, while models without ramping serve as the baseline. Vertical gray dashed lines indicate ramping positions.

Is it the unstable optimizer? To disentangle optimizer-induced instability from landscape-induced instability, we repeated the experiments using Muon, a substantially more stable optimizer than AdamW. In Figure 14, the loss spikes and plateaus consistently occurs under Muon when warmup is shortened or the peak LR is increased. This rules out the possibility that the behavior stems from AdamW's startup issues.

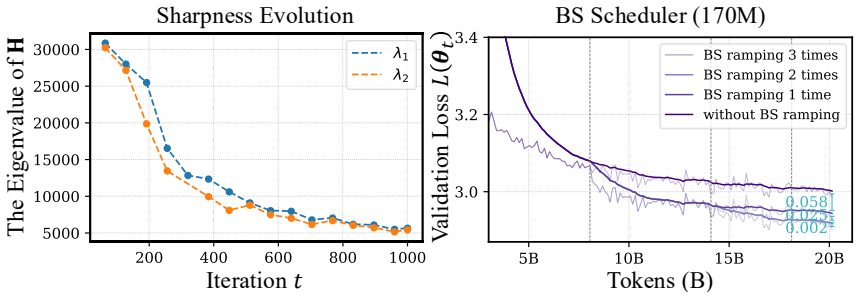

Figure 12: **Extensive to Lion optimizer. (Left). Early sharp-to-flat dynamics.** Evolution of the top eigenvalues of the Hessian across iterations: $\lambda_i(\mathbf{H}(\theta_t))$ vs. iteration t. **(Right). BS scheduling improves data efficiency. BS scheduling improves data efficiency.** We use a BS schedule that starts at 0.49M and increases by $4\times$ at each ramp. Models are trained with $1, 2$, or 3 ramping steps, while models without ramping serve as the baseline. Vertical gray dashed lines indicate ramping positions.

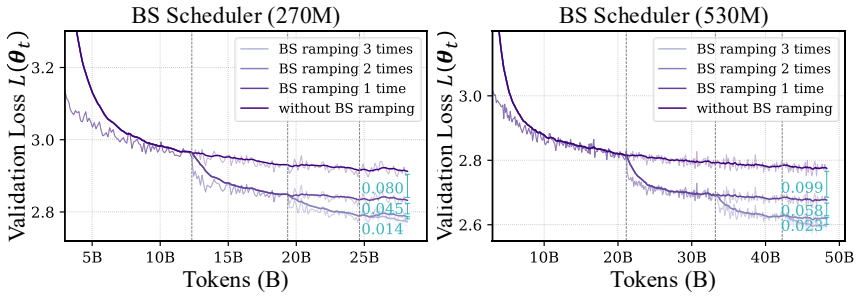

Figure 13: **Extensive to Larger Scale. (Left). Early sharp-to-flat dynamics.** Evolution of the top eigenvalues of the Hessian across iterations: $\lambda_i(\mathbf{H}(\theta_t))$ vs. iteration t. **(Right). BS scheduling improves data efficiency. BS scheduling improves data efficiency.** We use a BS schedule that starts at 0.49M and increases by $4\times$ at each ramp. Models are trained with $1, 2$, or 3 ramping steps, while models without ramping serve as the baseline. Vertical gray dashed lines indicate ramping positions.

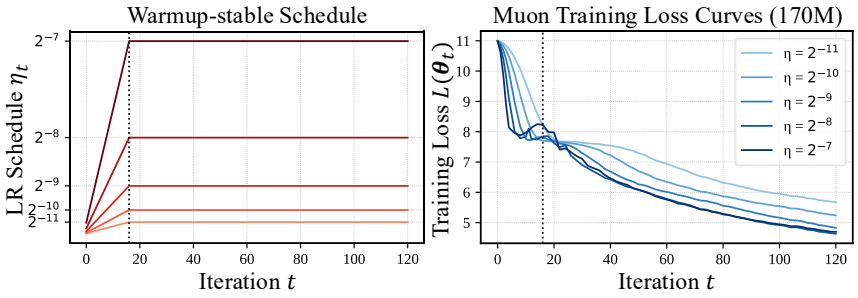

Figure 14: **Muon consistently shows loss spikes and plateaus early in training.** We train a series of `LLaMA-2` models with 170M parameters. We adopt a warmup-stable schedule, where the warmup length is shortened to 16 iterations and the peak LR is varied, $\eta \in \{2^{-11}, 2^{-10}, 2^{-9}, 2^{-8}, 2^{-7}\}$. **(Left).** LR schedule: η_t vs. training iteration t. **(Middle, Right).** Training loss curves for different model sizes: $L(\theta_t)$ vs. training iteration t. The vertical dashed line marks the end of the warmup phase.

What if we use longer warmup? We vary only the warmup length while fixing the peak LR at 2^{-7}. In Figure 15 (left), shorter warmup lengths lead to higher possibilities of loss spikes This behavior is consistent with the sharp-to-flat dynamics. Early in training, the model resides in sharper regions of the landscape, where only sufficiently small LRs ensure stable updates. Therefore, warmup is needed to gradually increase the LR until the trajectory enters flatter regions that can tolerate larger LRs.

What if zero warmup and small BS? We also conduct experiments with no warmup and small batch size. In Figure 15 (right), the loss spikes become even more significant. This aligns with our

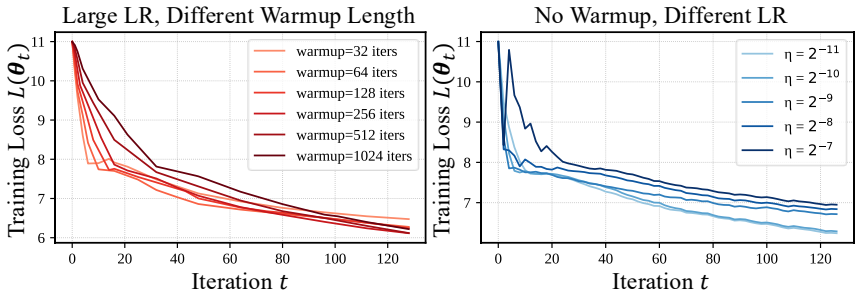

Figure 15: **(Left.) Shorter warmup, more loss spikes.** We train a series of LLaMA-2 models with 170M parameters. We adopt a warmup-stable schedule, where the warmup length varies from $\{2^5, 2^6, 2^7, 2^8, 2^9, 2^{10}\}$ iterations and the peak LR is fixed $\eta = 2^{-7}$. **(Right). Zero warmup and small BS leads to larger loss spikes.** We train a series of LLaMA-2 models with 170M parameters. We adopt a constant LR schedule, where no warmup and the peak LR is varied, $\eta \in \{2^{-11}, 2^{-10}, 2^{-9}, 2^{-8}, 2^{-7}\}$. We also use the 0.49M BS.

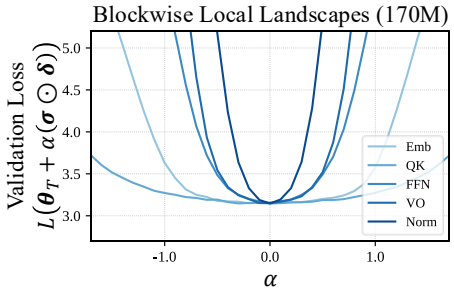

Figure 16: **Local loss landscape exhibits blockwise structure.** One-dimensional loss landscapes at the final iterate $\boldsymbol{\theta}_T$ along a masked random perturbation direction $\boldsymbol{\sigma} \odot \boldsymbol{\delta}$. Here, $\boldsymbol{\theta} \in \mathbb{R}^p$ is the random perturbation vector, and $\boldsymbol{\sigma}$ is a blockwise mask that zeros out perturbations outside the specified block type.

explanation: with no warmup, the LR jumps immediately to a large value while sharpness is still extremely high, causing a spike almost at initialization. More importantly, small BS does not replace warmup, and the instability still appears because of the sharpness.

D.4 Blockwise Landscape Structure

Visualizing the Blockwise Local Loss Landscape. Transformer architecture is composed of different block types, such as query–key (QK) and value–output (VO) projections, feedforward networks (FFN), normalization layers (Norm), and embedding layers (Embed). Prior studies (Zhang et al., 2024a; Wang et al., 2025) found that these block types exhibit heterogeneous levels of sharpness, suggesting that different block types contribute differently to the local loss landscape. To better understand this heterogeneity, we visualize the local loss landscape separately for each block type.

Similar to Figure 5, we perturb the final iterate $\boldsymbol{\theta}_T$ along a random direction $\boldsymbol{\delta}$, but restrict the perturbations to a selected block type using a blockwise mask $\boldsymbol{\sigma}$. In Figure 16 (right), local landscapes differ substantially across block types, and the curvature order we observe is QK < Embed < FFN < VO < Norm. This ordering is slightly different from the results reported by Wang et al. (2025), which found Embed < QK < FFN < VO < Norm. We suggest two possible reasons for this discrepancy. First, our analysis probes the most direction landscape, whereas Wang et al. (2025), following Wang et al. (2024), directly estimate sharpness from the fisher information matrix. Second, embeddings may not appear as the flattest block in terms of loss landscape, but as they are least activated during gradient propagation (many embedding entries receive no gradient), they are effectively flatter in training dynamics.