UNDERSTANDING THE MECHANISMS OF FAST HYPERPARAMETER TRANSFER

Anonymous authors
Paper under double-blind review

ABSTRACT

The growing scale of deep learning models has rendered exhaustive hyperparameter (HP) optimization prohibitively expensive. A promising solution is the use of scale-aware HPs, which can enable direct transfer of optimal settings from small-scale grid searches to large models with minimal performance loss. Such approaches are useful when the optimal settings converge "fast" enough with scale. While approaches like the Maximal Update Parameterization (μP) have empirically displayed *fast transfer* when scaling model width, a deeper conceptual understanding of the mechanisms that enable this is still missing. Our work establishes a systematic conceptual framework for analyzing fast HP transfer across different synthetic and practical scenarios. In synthetic settings, we present various quantitative examples where transfer either offers a provable computational advantage or fails even under μP. We then propose a key property that enables the fast transfer often observed in practice: through a novel decomposition of the optimization trajectory, we identify one component that rapidly converges with model width and determines the optimal HPs, and the other that continues to improve the loss with increased width but has negligible impact on HP choice. We conjecture that this decomposition elucidates the key mechanisms behind *fast transfer* and empirically validate it in settings such as large language model (LLM) training.

1 INTRODUCTION

Scaling-aware hyperparameters. A central dogma in empirical deep learning is that performance will steadily improve (Kaplan et al., 2020; Hoffmann et al., 2022) as training data and parameter count increase. This paradigm incentivizes practitioners to develop increasingly larger trained models while first conducting experiments at smaller, more economical scales. For such small-scale experimentation to effectively inform larger training runs, it becomes crucial to reason about hyperparameters (HPs) in a scaling-aware framework and to analyze the behavior of the *sequence* of progressively scaled-up training runs. For example, when scaling the width n of a neural network, we should explicitly conceptualize the learning rate as the product of a scale-independent HP η and a scaling factor n^{-a}. This perspective was initially formalized in the Tensor Program series (Yang & Hu, 2021; Yang et al., 2022; Yang & Littwin, 2023) for the width-scaling of neural networks. It was shown theoretically that the correct scaling for ensuring "optimal" training in the limit is the Maximal Update Parameterization (μP, Yang & Hu, 2021) which ensures that asymptotically the optimal η is scale-independent.

Hyperparameter transfer. Empirical evidence (Yang et al., 2022; Lingle, 2024) has demonstrated that in fact, μP enables a much stronger property which we refer to as *fast hyperparameter transfer* (see Section 3) that is not apparent from its theoretical derivation. At a high level, fast HP transfer occurs when optimal HPs converge significantly faster with respect to width than the performance measure of interest. This phenomenon allows practitioners to perform HP selection on smaller proxy models and subsequently apply these optimal values to larger-scale training runs, drastically reducing the cost of HP tuning. Despite its practical utility, this phenomenon remains primarily an empirical success, with limited theoretical understanding of its underlying mechanisms.

Our contributions. We provide insight into the puzzle of fast HP transfer by first developing a framework for reasoning about HP transfer in Section 2. Then in Section 3 we formally define fast HP transfer in terms of convergence rates of optimal HPs and the loss, and provide a direct connection to the "usefulness" of transfer when performing compute-optimal grid search. We quantify these convergence rates in synthetic settings, and demonstrate that while valid in certain linear model, fast transfer is not guaranteed in neural networks *a priori* even when using μP. Such examples illustrate

that the benefits of transfer heavily depend on structural properties of the training process emerging from the data, optimizer, and architecture.

To illuminate this structure, in Section 4 we introduce a novel *loss decomposition* obtained by decomposing the step-wise linearized loss change along the training trajectory. We empirically demonstrate that the linearization is an effective proxy for the true loss change when considering the *exponential moving average* (EMA) of the optimization trajectory. The faithfulness of this approximation arises due to the smoothness of the resulting EMA trajectory. Using the linearization, we track the loss change arising from the projection of the update in the top-k directions that maximize this change in the loss at each matrix layer. This decomposes the total loss change into two components: the *top-k loss* arising from the dominant k components and the remaining *residual loss*.

Intuitively, we might expect that the training behavior on the top-k subspaces concentrates quickly with width while the remaining directions provide performance gains as the width increases. In Figure 1 we validate this intuition by applying the loss decomposition introduced in Section 4 to a sequence of Llama-style transformers of increasing width trained using μP. In this setting we see that the leading top-k loss remains approximately invariant across widths, whereas the residual loss consistently improves with width, especially later in training. Moreover, since the top-k loss provides the majority of the loss decrease it is reasonable to expect that the optimal learning rate for the total loss will not deviate much from the optimal learning rate for the top-k loss. Therefore, for an appropriately chosen $k < n$ the top-k loss decomposition provides the following potential explanation for the phenomenon of fast transfer.

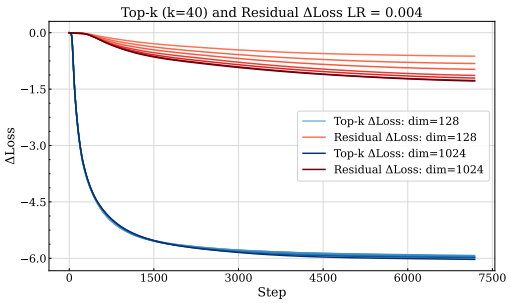


Figure 1: Loss decomposition into top-k and residual components in transformers with varying widths.

Fast transfer via loss decomposition

1. The top-k loss converges rapidly with n, hence so do the optimal top-k loss HPs.
2. The optimal HPs of the top-k loss *determine* the optimal HPs for the total loss.

In the rest of this paper, we provide a formalization for this intuitive explanation and use it to provide conceptual insights into the fast transfer phenomenon with support from experiments across synthetic and realistic settings, including large language model (LLM) training.

1.1 Related Work

Hyperparameter transfer. The concept of hyperparameter transfer was introduced by Yang et al. (2022), who showed that HPs tuned on small proxy models can transfer reliably to much larger ones under μP scaling. Subsequent empirical studies confirmed the effectiveness of learning rate transfer in transformers, while also highlighting certain limitations (Lingle, 2024; Vlassis et al., 2025). As Yang et al. (2022) noted, the success of hyperparameter transfer is not fully explained from first principles. Addressing this gap empirically, Noci et al. (2024) showed that top Hessian eigenvalues converge rapidly under μP across widths, suggesting a width-stable curvature that could underpin transfer. However, the connection between these spectral statistics and the optimal learning rate remain unclear. More recently, the concurrent work of Hong & Wang (2025) established a scale separation between certain macro- and micro-variables, thereby suggesting HPs can be effectively tuned in early training stages. In contrast to these results, our work explicitly defines (via a trajectory-level loss-decomposition) and connects the fast convergence of certain statistics of the optimization path to the fast convergence of optimal HPs.

Optimization trajectories. Our fast transfer hypothesis rests on the existence of low-dimensional structure in optimization trajectories. Gur-Ari et al. (2018) observed that gradients rapidly align with top eigenvectors of the Hessian, suggesting that GD operates within a "tiny subspace." Song et al. (2024) refined this view, showing that while gradient variance concentrates in the top eigenspace, motion in these directions primarily drive oscillations rather than loss reduction. This behavior is consistent with edge-of-stability (EoS) (Cohen et al., 2021; Damian et al., 2022a) and motivates our

use of EMA smoothing: by averaging out oscillatory components tied to the top eigenspace, the smoothed trajectory reveals the low-dimensional subspace where genuine loss decrease occurs.

Similar low-dimensional structure also appeared in various theoretical settings on gradient-based feature learning, such as the learning of multi-index models (Abbe et al., 2022; Damian et al., 2022b; Bietti et al., 2023; Mousavi-Hosseini et al., 2023; Lee et al., 2024), where SGD "localizes" the parameters into low-dimensional subspaces. Recent works have also shown that gradient updates induce a spiked (signal-plus-noise) eigenstructure in the conjugate kernel (Ba et al., 2022; Moniri et al., 2023; Wang et al., 2024; Dandi et al., 2024) or the Hessian (Arous et al., 2023; 2025) of the neural network, and the top eigenvectors contain information of the low-dimensional target function.

2 Preliminaries and Formal Framework

We now formalize our framework for HP transfer. We will let n denote the scaling dimension. The scaled HPs used during training at scale n are:

$$\mathcal{H}_n(\boldsymbol{\nu}, \boldsymbol{\gamma}) = (\nu_i n^{-\gamma_1}, \dots, \nu_h n^{-\gamma_h}),$$

where we refer to $\boldsymbol{\nu} = (\nu_1, \dots, \nu_h)$ as the HPs and to $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_h)$ as the scaling exponents. Conceptually, $\boldsymbol{\nu}$ are a set of n-independent constants which are tuned for a specific problem and $\boldsymbol{\gamma}$ are a set of exponents which specify how to scale the HPs with n so that training can be performed at any scale n using $\mathcal{H}_n(\boldsymbol{\nu}, \boldsymbol{\gamma})$. In this paper, we focus on the "optimization hyperparameters" of the abcd-parameterizations of Yang & Littwin (2023) (see Appendix A). This setting encompasses the width scaling of hyperparameters needed for training standard neural network architectures using common optimizers such as SGD and Adam. Thus we will often simply refer to the scale n as the width; however our framework can be used more broadly for reasoning about scale-aware HPs.

In our setting the *training procedure* $\mathcal{A}$ is held fixed and we only vary n, the HPs $\boldsymbol{\nu}$, and the scaling $\boldsymbol{\gamma}$. The training procedure returns an output $\mathcal{A}(n, \boldsymbol{\nu}, \boldsymbol{\gamma})$. For neural network training this means that the architecture, optimizer, dataset, etc. are all fixed, and we will let $\mathcal{A}(n, \boldsymbol{\nu}, \boldsymbol{\gamma})$ be the resulting optimization trajectory. For a given configuration, we can measure a scalar metric $\phi_n(\boldsymbol{\nu}; \boldsymbol{\gamma})$ obtained from the output of $\mathcal{A}$. For example, this can be the final validation loss of the model.

The HP search takes place over a *search space* $\mathcal{X}$ which we take to be a h dimensional box. We define the *optimal HPs* and the corresponding *optimal value* as

$$\boldsymbol{\nu}^\star(n; \boldsymbol{\gamma}) = \arg\min_{\boldsymbol{\nu} \in \mathcal{X}} \phi_n(\boldsymbol{\nu}; \boldsymbol{\gamma}), \quad \phi_n^\star(\boldsymbol{\gamma}) := \phi_n(\boldsymbol{\nu}^\star(n; \boldsymbol{\gamma}); \boldsymbol{\gamma})$$

For convenience, we omit the scaling exponents $\boldsymbol{\gamma}$ from the notation when they are clear from context, and abbreviate as $\phi_n(\cdot)$, $\boldsymbol{\nu}^\star(n)$, and $\phi_n^\star$. To formalize when a scaling exponent $\boldsymbol{\gamma}$ admits stable HP transfer across widths, we will need to impose regularity assumptions on the family $\{\phi_n\}$ so that a well-defined limit ϕ_∞ exists and the optimal HPs $\boldsymbol{\nu}^\star(n)$ converge to the minimizer $\boldsymbol{\nu}^\star(\infty)$ of ϕ_∞. Additionally, to quantify a notion of suboptimality due to transfer, a key condition we will impose is the **local strong convexity** of ϕ_∞. This condition links the suboptimality of a transferred HP to suboptimality in terms of loss and ensures that performance meaningfully degrades away from the optimum. Without such a condition, ϕ_∞ can be flat near its minimizer, making accurate HP selection irrelevant in the large-n limit. We will say that the scaling $\boldsymbol{\gamma}$ admits HP transfer (or simply that HP transfer holds) when the above conditions are satisfied. The rigorous formulation is given in Definition 3 (Appendix A). We regard this as the weakest notion of HP transfer, since it concerns only the asymptotic convergence of certain quantities rather than their convergence rates.

3 The Puzzle of Useful Transfer

In this section, we relate the convergence rate of optimal HPs to that of the evaluation metric (e.g., validation loss). An important aspect of our definition of HP transfer in Definition 3 is that $\boldsymbol{\nu}^\star(n) \to \boldsymbol{\nu}^\star(\infty)$ as shown in Proposition 8. As we will see in Proposition 1, the convergence rate of $\phi_n \to \phi_\infty$ already implies an upper bound on the convergence rate of the minimizers $\boldsymbol{\nu}^\star(n)$ which comes from the local strong convexity condition — it turns out that this convergence also informs whether it is computationally more efficient to use HP transfer than to tune the model at a single width n, as we show in Theorems 13 and 14. Without further assumptions this bound in Proposition 1 is tight. In practice, however, we often observe "fast transfer": the convergence rate of the minimizers is (much) faster than what is implied by this agnostic bound. Clearly, the fast transfer phenomenon must arise from additional structure in practical settings, which we investigate in the ensuing sections.

3.1 Tracking Scale-dependent Quantities

We make use of the following asymptotic notation. Given sequences $x_n \in \mathbb{R}$ and $y_n \in \mathbb{R}^+$, we write $x_n = O(y_n)$ to mean there exists $c > 0$ such that $|x_n| \leq c y_n$ for all large n. We write $x_n = o(y_n)$ if $x_n/y_n \to 0$ as $n \to \infty$. Lastly, we write $x_n = \Theta(y_n)$ if there exist $c_1, c_2 > 0$ such that $c_1 y_n \leq |x_n| \leq c_2 y_n$ for large n. As shorthand, we write $x_n \sim y_n$ to mean $x_n = \Theta(y_n)$.

Definition 1 (Convergent Quantities)**.** *We define the following width n-dependent quantities.*

- ***Loss gap:*** $a_n = |\phi_n^\star - \phi_\infty^\star|$. *The quantity measures the discrepancy between the optimal value of the metric ϕ at the finite-width model (over all HPs) and that of the infinite-width model.*
- ***Hyperparameter gap:*** $b_n = \|\boldsymbol{\nu}^\star(n) - \boldsymbol{\nu}^\star(\infty)\|$. *The quantity measures the discrepancy between the optimal HPs at width n and the infinite-width counterpart.*
- ***Suboptimality gap:*** $c_n = |\phi_\infty(\boldsymbol{\nu}^\star(n)) - \phi_\infty^\star|$. *The quantity measures the performance gap between two infinite-width models: one with the optimal HP tuned at infinite-width, and the other with the optimal HP at a smaller width. Intuitively speaking, this quantity reflects the performance gap in transferring the optimal HP in a small model to a large model.*

Fast Transfer. In the following, in addition to assuming HP transfer holds (Def. 3 in Appendix B.1) we will make the mild technical assumption that the "uniform loss gap is locally tight" which implies that $a_n \sim \|\phi_n - \phi_\infty\|_{\sup}$ (see Def. 4 and Lemma 12 in Appendix B.2). The following proposition uses strong convexity around the optimal infinite-width HP to directly connect the convergence rates of the optimal HP and the evaluation metric (see Appendix B.2).

Proposition 1. *$b_n = O\big(a_n^{1/2}\big)$ and $c_n = \Theta(b_n^2) = O(a_n)$.*

Since HP transfer holds, we have by definition $a_n, b_n, c_n \to 0$ as $n \to \infty$. This being said, *a priori* we may have $b_n = \Theta\big(a_n^{1/2}\big)$ and hence $c_n = \Theta(a_n)$. This would indicate that the suboptimality gap converges at the same rate as the loss gap. In practice however we tend to observe a much faster rate for c_n relative to a_n. We refer to this phenomenon formally as *fast transfer*.

Definition 2. *We say that HP transfer is fast if $c_n = o(a_n)$ which occurs iff $b_n = o\big(a_n^{1/2}\big)$.*

For instance, if $a_n \sim n^{-\alpha}$ and $b_n \sim n^{-\beta}$, then fast transfer occurs if and only if $\beta > \alpha/2$.

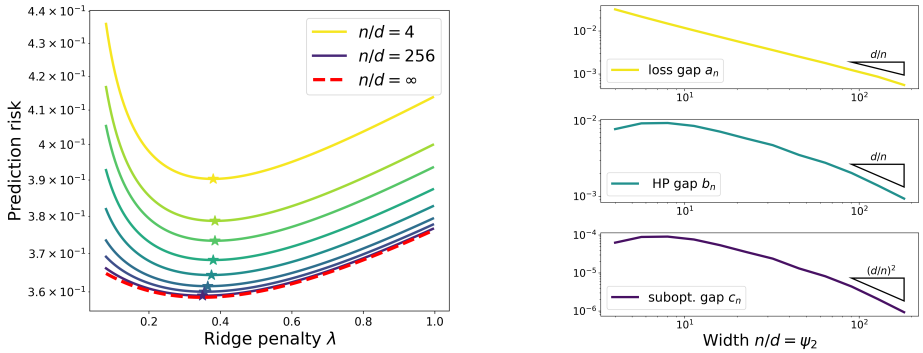
Useful Transfer. The notion of fast transfer also coincides with the computational usefulness of HP transfer when performing a grid search. More precisely, consider two HP tuning strategies: **direct** performs grid search on models of a single width, and **transfer** performs grid search on models of a smaller width and then trains a final model of larger width using the obtained optimal HPs. We say that transfer is "useful" if the transfer strategy yields better performance than the direct strategy for a given compute budget. The following result characterizes when this occurs. We provide the formal results with explicit rates in Theorems 13 and 14 found in Appendix B.3.

Theorem 2 (Informal)**.** *Suppose $a_n \sim n^{-\alpha}$ and $b_n \sim n^{-\beta}$. Given a compute budget of $\mathcal{F}$ flops, let $p^{\mathrm{dir}}(\mathcal{F})$ and $p^{\mathrm{tr}}(\mathcal{F})$ be the compute-optimal performance under the direct and transfer strategies, respectively. Then as $\mathcal{F} \to \infty$, $p^{\mathrm{tr}}(\mathcal{F}) \sim p^{\mathrm{dir}}(\mathcal{F})$ iff $\beta = \alpha/2$ and $p^{\mathrm{tr}}(\mathcal{F}) \ll p^{\mathrm{dir}}(\mathcal{F})$ iff $\beta > \alpha/2$.*

Observe that the requirement $\beta > \alpha/2$ is the same condition as *fast transfer* (Definition 2). Consequently, under local strong convexity of ϕ_∞ with respect to $\boldsymbol{\nu}(n)$, the naive loss convergence rate already implies that transfer never underperforms the direct tuning strategy asymptotically, provided that the HPs are parameterized to be n-independent. While this supports the effectiveness of μ-transfer (Yang et al., 2022), it does not address the question of when the optimal HPs converge faster than what is implied naively by the loss convergence. The question turns out to be subtle, and the following subsection presents simple examples where fast transfer may be present or absent.

3.2 Examples of Fast and Slow Hyperparameter Transfer

Note that the precise scaling of quantities given in Definition 1 are computationally prohibitive to measure in large-scale scenarios: since the infinite-scale model ($n \to \infty$) in inaccessible, we must resort to power-law fit from finite-n data, which requires a refined grid search of HPs at each scale. Consequently, in this section we present synthetic settings, where the convergence rate of (a_n, b_n, c_n) can be either analytically derived or reliably estimated from data.



(a) Prediction risk vs. ridge penalty. (b) Scaling of optimal loss and ridge penalty.

Figure 2: Optimal ridge penalty (generalization error) for RF regression to learn a single-index model.

Fast HP Transfer: Random Features Regression. First we consider tuning the ridge penalty λ in a high-dimensional random features (RF) model with nonlinearity σ, where the target function is a single-index model with link function σ_* on isotropic Gaussian input in $\mathbb{R}^d$. We aim to select the optimal regularization parameter $\lambda \in \mathbb{R}$ that minimizes the prediction risk $\mathcal{R}(f) = \mathbb{E}_{\boldsymbol{x}}[(y - f(\boldsymbol{x}))^2]$.

In the proportional limit where the number of training data N, dimension of input features d, and model width n all diverge $N, d, n \to \infty, N/d \to \psi_1, n/d \to \psi_2$ where $\psi_1, \psi_2 \in (0, \infty)$, Mei & Montanari (2022); Gerace et al. (2020) derived precise asymptotics of prediction risk under standard assumptions (see Assumption 2). In this model, the number of trainable parameters is controlled by the ratio $\psi_2 = n/d$, and the infinite-width model is obtained by sending $\psi_2 \to \infty$. The following proposition quantifies the convergence of prediction risk and hyperparameter as a function of ψ_2.

Proposition 3. *Define $\lambda^*(\psi_2) := \arg\min_{\lambda \in \mathbb{R}} \mathcal{R}_{\psi_2}(\lambda)$, where $\mathcal{R}_{\psi_2}(\lambda)$ is the asymptotic prediction risk of the RF model with width $n/d = \psi_2$ and ridge penalty λ. Then under Assumption 2, we have*

- ***Loss gap:*** $|\mathcal{R}_{\psi_2}(\lambda^*(\psi_2)) - \mathcal{R}_\infty(\lambda^*(\infty))| = \Theta(\psi_2^{-1})$.
- ***Hyperparameter gap:*** $|\lambda^*(\psi_2) - \lambda^*(\infty)| = O(\psi_2^{-1})$.
- ***Suboptimality gap:*** $|\mathcal{R}_\infty(\lambda^*(\psi_2)) - \mathcal{R}_\infty(\lambda^*(\infty))| = O(\psi_2^{-2})$.

This proposition states that both the loss gap a_n and the HP gap b_n scale as $d/n = \psi_2^{-1}$, whereas the suboptimality gap scales as $c_n \sim \psi^{-2} \ll a_n$; we conclude that the ridge regularization parameter λ exhibits fast transfer per Definition 2. Consequently, Theorem 13 implies that tuning the ridge penalty on small (narrower) RF model and then transfer is more compute-efficient than directly tuning the large model. To our knowledge, this gives the first concrete setting where the hyperparameter transfer strategy in Yang et al. (2023) provably offers computational advantage.

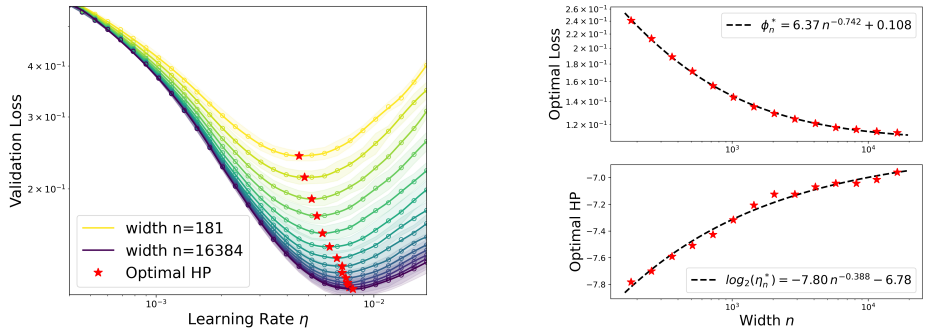
Figure 2 presents the prediction risk of the RF ridge estimator across varying widths $n/d = \psi_2$; we set $\sigma = \tanh, \sigma_* = \mathrm{ReLU}, \psi_1 = 4, \sigma_\varepsilon = 1/4$. The analytical curves are obtained by solving the coupled Stieltjes transforms (6)(7). Figure 2(b) confirms the scaling of a_n, b_n, c_n in Proposition 3.

Slow HP Transfer: Two-layer ReLU Network. Next we consider a classification setting with shallow ReLU neural network: $f(\boldsymbol{x}) = \sum_{i=1}^n a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i)$, where we aim to tune the learning rate η that minimizes the validation loss. We set the target to be the norm indicator function, which is a well-studied function that requires a wide two-layer network to approximate (Safran & Lee, 2022),

$$y = \mathbb{1}\{\|\boldsymbol{x}\|_2^2 > F^{-1}_{\chi^2_d}(0.5)\}, \quad \text{where } \boldsymbol{x} \sim \mathcal{N}(0, \boldsymbol{I}_d),$$

where $F^{-1}_{\chi^2_d}(0.5)$ represents the median of a chi-square distribution with d degrees of freedom – this threshold ensures that the classes are exactly balanced. We set $d = 2^6, n = 2^{14}$, and run the Adam optimizer (Kingma & Ba, 2014) for $T = 2^{14}$ steps with batch size 2^8 to minimize the binary cross-entropy loss. The initialization and learning rate are set according to μP (Yang & Hu, 2021).

In Figure 3(a) we observe that even under μP, the optimal learning rate still drifts towards the right. Moreover, Figure 3(b) illustrates that under a power-law fit, the HP of interest η converges slower than the validation loss, and the estimated scaling $\beta \approx \alpha/2$ suggests that the HP convergence rate does not beat the agnostic rate from strong convexity. An interesting future direction is to rigorously derive the scaling exponents and demonstrate slow transfer in this μP example.



(a) Validation loss vs. learning rate.

(b) Scaling of optimal loss and learning rate.

Figure 3: Optimal learning rate (validation loss) for two-layer ReLU network to learn the ball indicator function.

4 Fast Transfer via Trajectory Decomposition

At a high-level, the only way for fast transfer to occur is if somehow the optimal HP is "dependent" on some statistics of the training trajectory which converge much faster than the statistics of the trajectory on which the loss depends. To provide a clearer understanding we will try to explicitly "extract" the fast converging statistics and connect these with the optimal HP. To do so we will leverage prior intuition about the "low-dimensional" nature of optimization trajectories (see Section 1.1). Intuitively, we may expect that the movement in a small number of directions might simultaneously "dominate" the determination of the optimal hyperparameters since these correspond to a majority of the loss decrease and is sensitive to the choice of hyperparameters. Furthermore, the associated low-dimensional statistics converge fast with width and are nearly "width-invariant".

4.1 Top-k Loss Decomposition

Let us consider an optimization trajectory $\boldsymbol{\omega} = (\boldsymbol{w}_0, \ldots, \boldsymbol{w}_T)$ of a neural network. Define the one-step loss change $\delta\mathcal{L}(\boldsymbol{w}_t) := \mathcal{L}(\boldsymbol{w}_{t+1}) - \mathcal{L}(\boldsymbol{w}_t)$, so that the overall loss change is the sum

$$\Delta\mathcal{L}(\boldsymbol{\omega}) := \mathcal{L}(\boldsymbol{w}_T) - \mathcal{L}(\boldsymbol{w}_0) = \sum_{t=0}^{T-1} \delta\mathcal{L}(\boldsymbol{w}_t).$$

Let $\boldsymbol{g}_t := \boldsymbol{\nabla}\mathcal{L}(\boldsymbol{w}_t)$ and $\delta\boldsymbol{w}_t := \boldsymbol{w}_{t+1} - \boldsymbol{w}_t$. Let us define $\delta\phi(\boldsymbol{w}_t) := \langle \boldsymbol{g}_t, \delta\boldsymbol{w}_t \rangle$ to be the linearization of $\delta\mathcal{L}(\boldsymbol{w}_t)$. Under appropriate smoothness conditions on the trajectory,

$$\delta\phi(\boldsymbol{w}_t) \approx \delta\mathcal{L}(\boldsymbol{w}_t) \text{ and } \phi(\boldsymbol{\omega}) := \sum_{t=0}^{T-1} \delta\phi(\boldsymbol{w}_t) \approx \Delta\mathcal{L}(\boldsymbol{\omega}).$$

To ensure that such smoothness conditions hold and $\phi(\boldsymbol{\omega})$ is a useful proxy for $\Delta\mathcal{L}(\boldsymbol{\omega})$, we take $\boldsymbol{\omega}$ to be an exponential moving average (EMA) of the base optimization trajectory. Empirically, on realistic problems, we obtain excellent agreement between $\phi(\boldsymbol{\omega})$ and $\Delta\mathcal{L}(\boldsymbol{\omega})$ for EMA trajectories which perform at least as well as the corresponding base trajectories (see Fig. 5).

The utility of considering the linearization $\phi(\boldsymbol{\omega})$ in our setting is that we can decompose $\delta\phi(\boldsymbol{w}_t)$ based on the structure of $(\boldsymbol{g}_t, \delta\boldsymbol{w}_t)$ and recover a resulting decomposition of $\phi(\boldsymbol{\omega})$. Let us consider a fixed time index t. If $\boldsymbol{w} = (\boldsymbol{W}^{(1)}, \ldots, \boldsymbol{W}^{(L)})$ are the model parameters with corresponding gradients $\boldsymbol{g} = (\boldsymbol{G}^{(1)}, \ldots, \boldsymbol{G}^{(L)})$ such that $\boldsymbol{W}^{(\ell)}$ and $\boldsymbol{G}^{(\ell)}$ are tensors of the same shape, then

$$\langle \boldsymbol{g}, \delta\boldsymbol{w} \rangle = \sum_{\ell \in [L]} \left\langle \boldsymbol{G}^{(\ell)}, \delta\boldsymbol{W}^{(\ell)} \right\rangle.$$

Consider a single summand coming from $\boldsymbol{W} \in \mathbb{R}^{m \times n}$ with corresponding gradient $\boldsymbol{G} \in \mathbb{R}^{m \times n}$. Define the matrix $\mathcal{S}(\boldsymbol{G}, \delta\boldsymbol{W})$ which symmetrizes the product $\boldsymbol{G}^\top \cdot \delta W$ as

$$\mathcal{S}(\boldsymbol{G}, \delta\boldsymbol{W}) := \frac{1}{2}(\boldsymbol{G}^\top \cdot \delta\boldsymbol{W} + \delta\boldsymbol{W}^\top \cdot \boldsymbol{G}).$$

If $\mathcal{S}(\boldsymbol{G}, \delta\boldsymbol{W})$ has eigenvalues $\lambda_1, \ldots, \lambda_n$ such that $|\lambda_1| \geq \cdots \geq |\lambda_n|$, then we define the top-k step-wise linear loss change for parameter $\boldsymbol{W}$ to be the sum of its first k eigenvalues.

$$\delta\phi^k(\boldsymbol{W}) := \sum_{i=1}^{k} \lambda_i. \tag{1}$$

The top-k loss change captures an intuitive notion of the loss change in the top-k directions of maximum change in loss. Note that $\delta\phi^n(\boldsymbol{W}) = \delta\phi(\boldsymbol{W})$. If the update $\delta\boldsymbol{W}$ is aligned with the gradient, that is $\boldsymbol{G} = c \cdot \delta\boldsymbol{W}$ for a constant $c \neq 0$, then $\delta\phi^k(\boldsymbol{W})$ is the sum of the top-k singular values of $\boldsymbol{G}$. We can add the step-wise changes over time and over all parameters[1] to compute the top-k loss change,

$$\phi^k(\boldsymbol{\omega}) := \sum_{\ell \in [L]} \sum_{t=0}^{T-1} \delta\phi^k\left(\boldsymbol{W}_t^{(\ell)}\right). \tag{2}$$

For some parameters we will use a "row version" of $\delta\phi^k$ which uses $\mathcal{S}(\boldsymbol{G}^\top, \delta\boldsymbol{W}^\top)$ in place of $\mathcal{S}(\boldsymbol{G}, \delta\boldsymbol{W})$. Then we can make the index k span $[n]$ across all layers so that using the same k for each layer in Eq. (2) becomes reasonable.

Decomposition-aware HP transfer. We can now make our earlier intuitions more precise using this step-wise decomposition. Assume that we fix a training procedure $\mathcal{A}$ (see Definition 3). Then we can define the width-n trajectory trained with HPs $\boldsymbol{\nu}$ and scaling $\boldsymbol{\gamma}$ to be the trajectory $\boldsymbol{\omega}_n(\boldsymbol{\nu}, \boldsymbol{\gamma})$ obtained from executing the training procedure $\mathcal{A}$ with hyperparameters $\mathcal{H}_n(\boldsymbol{\nu}, \boldsymbol{\gamma})$. If we consider a fixed scaling $\boldsymbol{\gamma}$ and an HP dependent *truncation function* κ which outputs an index $\kappa(\boldsymbol{\nu}) \in [n]$ given HPs $\boldsymbol{\nu}$, then we can abbreviate

$$\begin{aligned} &\phi_n(\boldsymbol{\nu}) := \phi(\boldsymbol{\omega}_n(\boldsymbol{\nu}, \boldsymbol{\gamma})),\ \phi_n^\kappa(\boldsymbol{\nu}) := \phi^{\kappa(\boldsymbol{\nu})}(\boldsymbol{\omega}_n(\boldsymbol{\nu}, \boldsymbol{\gamma})) \\ &\boldsymbol{\nu}^\star(n) := \arg\min_{\boldsymbol{\nu}} \phi_n(\boldsymbol{\nu}),\ \boldsymbol{\nu}_\kappa^\star(n) := \arg\min_{\boldsymbol{\nu}} \phi_n^{\kappa(\boldsymbol{\nu})}(\boldsymbol{\nu}). \end{aligned} \tag{3}$$

We will refer to ϕ_n^κ as the **top-κ loss curve** and $\phi_n^{-\kappa} := \phi_n - \phi_n^\kappa$ as the **residual loss curve**.

Recall our initial intuition for fast transfer based on a loss decomposition. Conceptually, in our framework, this intuition says that fast transfer occurs if for an appropriately chosen sequence κ_n the following are simultaneously true for large enough n

- Top-κ strong convexity: The top-κ_n losses $\phi_n^{\kappa_n}$ and $\phi_n^{\kappa_n}$ are strongly-convex.
- Top-κ invariance: The top-κ_n loss converges rapidly so $\phi_n^{\kappa_n} \approx \phi_\infty^{\kappa_n}$ and $\boldsymbol{\nu}_{\kappa_n}^\star(n) \approx \boldsymbol{\nu}_{\kappa_n}^\star(\infty)$.
- Residual Flatness: The residuals $\phi_n^{-\kappa_n}$ and $\phi_\infty^{-\kappa_n}$ are both "flat" as functions of $\boldsymbol{\nu}$, and so $\boldsymbol{\nu}_{\kappa_n}^\star(n) \approx \boldsymbol{\nu}^\star(n)$ and $\boldsymbol{\nu}_{\kappa_n}^\star(\infty) \approx \boldsymbol{\nu}^\star(\infty)$.

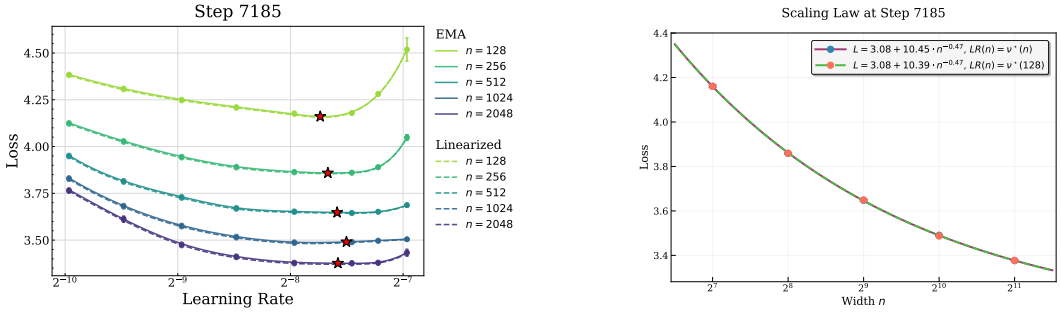
It follows that when these conditions hold that $\boldsymbol{\nu}^\star(n) \approx \boldsymbol{\nu}^\star(\infty)$. In Appendix B.5 we make these intuitions more formal by defining a quantity $\mathcal{J}_n(\kappa)$ which given a truncation function κ upper bounds the HP gap b_n using quantitative measures of top-κ invariance and residual flatness. By choosing a truncation function $\kappa_n^\star$ which minimizes $\mathcal{J}_n(\kappa)$ by optimally balancing the top-κ invariance and residual flatness we obtain the best such upper bound on b_n. In practice, it is intractable to optimize this objective so instead we optimize a proxy objective $\mathcal{J}_{\text{proxy}}(\kappa)$ to obtain a truncation function $\kappa_n := \hat{\kappa}(n)$ using a procedure outlined in Algorithm 1 (see Appendix C). We will then validate empirically that top-κ invariance and residual flatness holds qualitatively with our choice of $\hat{\kappa}(n)$.[2]

4.2 Experimental Results

Experimental Setup We train a Llama-style transformer architecture (Touvron et al., 2023) using the Adam (Kingma & Ba, 2014) optimizer with a warmup stable (WSD) learning rate schedule (Hu et al., 2024) on WikiText-103 (Merity et al., 2016). Further experimental details are given in Appendix D. There we also show that similar observations hold for the Adam β_1 and β_2 hyperparameters (App. D.4) and for the learning rate when training a 2 layer MLP using SGD on CIFAR-10 (App. D.6). In Appendix D.5, we investigate the recently popularized Muon optimizer (Jordan et al., 2024).

[1] We will only decompose matrix parameters and use the full inner-product for vector parameters.

[2] It suffices to just exhibit one such $\hat{\kappa}(n)$, but there can be many ways of producing qualitatively similar κ_n.



(a) EMA loss (solid) and linearized loss (dashed).

(b) Scaling law for the EMA loss.

Figure 4: Training a 4-layer Llama transformer on WikiText-103. **Left:** EMA and linearized losses nearly coincide. **Right:** We plot the EMA loss as a function of width n using (1): the optimal learning rates $\nu^\star(n)$ (2): the optimal width-128 learning rate $\nu^\star(128)$. For Adam these two curves overlap indicating perfect transfer.

In this setting we find that transfer is slightly worse than for Adam and that the decomposition yields qualitatively different results. We conjecture that this results from the whitening of the gradient which prevents the updates from being low-rank. More careful investigation of the transfer properties of Muon is an exciting area for future work.

Fast Transfer and Linearization Faithfulness. The Adam learning rate sweeps are shown in Figure 4. As we can see from Figures 5 and 4, the EMA loss and the linearized loss ϕ are nearly indistinguishable indicating that the EMA trajectory is sufficiently smooth. From Figure 5 we see that the smoothing does not degrade the final loss. Our setting also clearly exhibits fast transfer since the optimal learning rate is converging rapidly with the width n (see Fig. 4a) while the reducible loss improves more slowly with the model width, converging to zero with rate $n^{-0.49}$ (see Fig. 4b). Using the optimal learning rate obtained at width $n = 128$ for larger widths widths is essentially optimal as indicated by the fact that the two curves in Figure 4b overlap. We now further probe the optimization and scaling dynamics in this fast transfer setting using the lens of our decomposition.

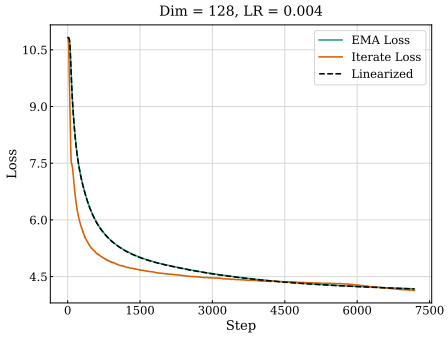


Figure 5: Training a transformer on WikiText-103 (see Section 4.2). The linearized loss and EMA loss are identical through training and close to the iterate loss at the final step.

Decomposition Over Time. In Figure 1 we visualize the top-k loss with $k = 40$ and the residual loss across training time and different widths for a fixed learning rate when training with Adam. We see that throughout training the top-k loss is nearly width invariant and accounts for the majority of the loss decrease. This indicates that the bulk of the improvement due to optimization comes from a low-dimensional subspace. As a result, the majority of the loss improvement due to width comes from improving the residual loss. This suggests that the additional learning is mostly occurring in the bottom "modes" of the trajectory. These modes become more dominant later in training.

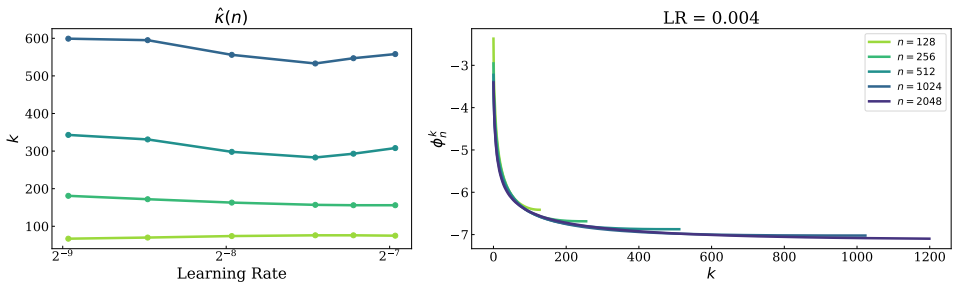


Figure 6: **Left:** The computed values of $\hat{\kappa}(n)$ using Algorithm 1. **Right:** The top-k loss ϕ_n^k for LR $= 0.004$. the ϕ_n^k descend rapidly with k and overlap over different n for an intermediate range of k where top-k invariance holds.

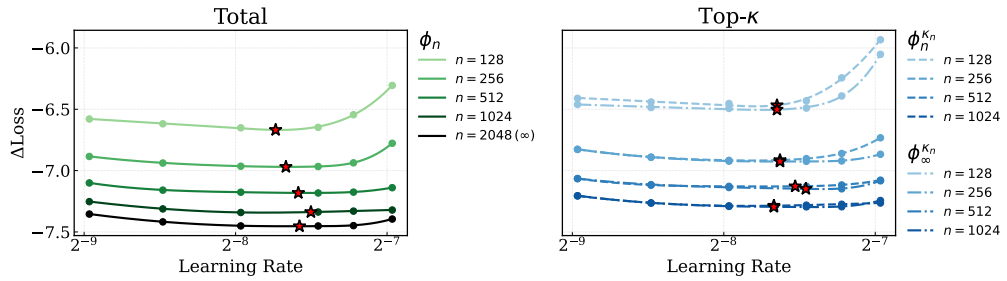


Figure 7: **Left:** Total loss curves ϕ_n. **Right:** Top-κ loss curve pairs $\phi_n^{\kappa_n}$ (dashed) and $\phi_\infty^{\kappa_n}$ (dash-dotted). The top-κ pairs are nearly overlapping with minimizers close to those of the corresponding total losses.

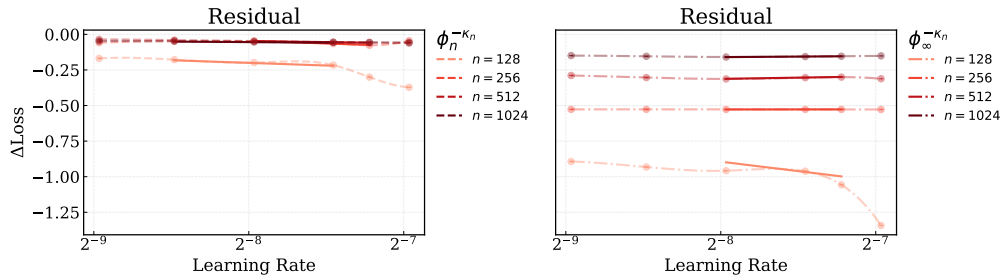


Figure 8: Residuals are nearly flat around the top-κ minimizers (solid lines).

Decomposition Across Widths. In Figure 7, we apply our loss decomposition across different n and compute $\kappa_n = \hat{\kappa}(n)$. We use the largest width $n_{\max} = 2048$ as an infinite-width proxy and consider transfer from widths $n < n_{\max}$. In the right panel, we see that $\phi_n^{\kappa_n} \approx \phi_\infty^{\kappa_n}$, that is the top-$\kappa_n$ loss is approximately the same for the width n model and the infinite-width proxy across different learning rates, despite the large gap in the total losses ϕ_n and ϕ_∞ shown in the left panel. Furthermore, the minimizers of the total loss are essentially dictated by the minimizers of the respective top-κ_n loss, that is $\boldsymbol{\nu}^\star_{\kappa_n}(n) \approx \boldsymbol{\nu}^\star(n)$ and $\boldsymbol{\nu}^\star_{\kappa_n}(\infty) \approx \boldsymbol{\nu}^\star(\infty)$ (see Eq. (3)). In Figure 8 we plot the corresponding residuals and see these are much more locally "flat" around the corresponding top-κ_n minimizer. As a result, the residuals contribute less to the determination of the overall optimal learning rate.

In the left panel of Figure 12 we show the computed values of $\hat{\kappa}(n)$ which appear to be approximately constant across learning rates and grow sublinearly with n. In Figure 6 we can see the value of ϕ_n^k for a fixed learning rate as we vary the value of k. The top-k loss is smooth with k and starts to flatten out once k is not too small which shows that our results are not highly sensitive to the choice of $\hat{\kappa}(n)$. The top-k invariance can also be seen to hold cleanly for intermediate values of k. Overall, we can qualitatively see how our decomposition can account for fast transfer in the sense described in Section 4, even when the convergence of the loss itself is much slower. The above provides concrete evidence for our central hypothesis that there is a low-dimensional projection of the trajectory which remains nearly invariant across width and is responsible for dictating the learning rate.

5 Conclusion

This work introduces a novel conceptual framework to reason about hyperparameter transfer and its underlying driving forces. We posit that a basic form of HP transfer can hold generically due to asymptotic considerations, but we show that this asymptotic condition can fail to be useful in practice. We conjecture that the utility of HP transfer is in fact dependent on the existence of non-trivial low-dimensional structure in the optimization dynamics. Through a novel decomposition of the dynamics we propose concrete measurements of such structure and argue that these provide an informative sufficient condition for useful HP transfer. Our experiments explicitly show that this structure exists in practice and may underlie the practical success of HP transfer. We hope that this will motivate further investigation into what settings allow for efficient HP transfer and a deeper understanding of optimization dynamics across scales.

LLM Usage. Large language models are used to polish the abstract, find relevant references in the related work section, and symbolically verify a computation in Appendix B.4.

REFERENCES

Emmanuel Abbe, Enric Boix Adsera, and Theodor Misiakiewicz. The merged-staircase property: a necessary and nearly sufficient condition for sgd learning of sparse functions on two-layer neural networks. In *Conference on Learning Theory*, pp. 4782–4887. PMLR, 2022.

Gerard Ben Arous, Reza Gheissari, Jiaoyang Huang, and Aukosh Jagannath. High-dimensional sgd aligns with emerging outlier eigenspaces. *arXiv preprint arXiv:2310.03010*, 2023.

Gerard Ben Arous, Reza Gheissari, Jiaoyang Huang, and Aukosh Jagannath. Local geometry of high-dimensional mixture models: Effective spectral theory and dynamical transitions. *arXiv preprint arXiv:2502.15655*, 2025.

Jimmy Ba, Murat A Erdogdu, Taiji Suzuki, Zhichao Wang, Denny Wu, and Greg Yang. High-dimensional asymptotics of feature learning: How one gradient step improves the representation. *Advances in Neural Information Processing Systems*, 35:37932–37946, 2022.

Peter L Bartlett, Andrea Montanari, and Alexander Rakhlin. Deep learning: a statistical viewpoint. *Acta numerica*, 30:87–201, 2021.

Alberto Bietti, Joan Bruna, and Loucas Pillaud-Vivien. On learning Gaussian multi-index models with gradient flow. *arXiv preprint arXiv:2310.19793*, 2023.

Lénaïc Chizat and Praneeth Netrapalli. The feature speed formula: a flexible approach to scale hyper-parameters of deep neural networks. *Advances in Neural Information Processing Systems*, 37:62362–62383, 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 *International Conference on Learning Representations (ICLR)*, 2021.

Alex Damian, Eshaan Nichani, and Jason D Lee. Self-stabilization: The implicit bias of gradient descent at the edge of stability. *arXiv preprint arXiv:2209.15594*, 2022a.

Alexandru Damian, Jason Lee, and Mahdi Soltanolkotabi. Neural networks can learn representations with gradient descent. In *Conference on Learning Theory*, pp. 5413–5452. PMLR, 2022b.

Yatin Dandi, Luca Pesce, Hugo Cui, Florent Krzakala, Yue M Lu, and Bruno Loureiro. A random matrix theory perspective on the spectrum of learned features and asymptotic generalization capabilities. *arXiv preprint arXiv:2410.18938*, 2024.

Edgar Dobriban and Stefan Wager. High-dimensional asymptotics of prediction: Ridge regression and classification. *The Annals of Statistics*, 46(1):247–279, 2018.

Mario Geiger, Stefano Spigler, Arthur Jacot, and Matthieu Wyart. Disentangling feature and lazy training in deep neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2020 (11):113301, 2020.

Federica Gerace, Bruno Loureiro, Florent Krzakala, Marc Mézard, and Lenka Zdeborová. Generalisation error in learning with random features and the hidden manifold model. In *International Conference on Machine Learning*, pp. 3452–3462. PMLR, 2020.

Guy Gur-Ari, Daniel A Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

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.

Letong Hong and Zhangyang Wang. On the provable separation of scales in maximal update parameterization. In *Forty-second International Conference on Machine Learning*, 2025.

Hong Hu and Yue M Lu. Universality laws for high-dimensional learning with random features. *IEEE Transactions on Information Theory*, 69(3):1932–1964, 2022.

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. In *Conference on Language Models (COLM)5*, 2024.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. URL https://kellerjordan.github.io/posts/muon/.

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.

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

Jason D Lee, Kazusato Oko, Taiji Suzuki, and Denny Wu. Neural network learns low-dimensional polynomials with sgd near the information-theoretic limit. *arXiv preprint arXiv:2406.01581*, 2024.

Lucas Lingle. An empirical study of mup learning rate transfer. *arXiv preprint arXiv:2404.05728*, 2024.

Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and the double descent curve. *Communications on Pure and Applied Mathematics*, 75 (4):667–766, 2022.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Behrad Moniri, Donghwan Lee, Hamed Hassani, and Edgar Dobriban. A theory of non-linear feature learning with one gradient step in two-layer neural networks. *arXiv preprint arXiv:2310.07891*, 2023.

Alireza Mousavi-Hosseini, Sejun Park, Manuela Girotti, Ioannis Mitliagkas, and Murat A Erdogdu. Neural networks efficiently learn low-dimensional representations with SGD. In *The Eleventh International Conference on Learning Representations*, 2023.

Lorenzo Noci, Alexandru Meterez, Thomas Hofmann, and Antonio Orvieto. Super consistency of neural network landscapes and learning rate transfer. *Advances in Neural Information Processing Systems*, 37:102696–102743, 2024.

Itay Safran and Jason Lee. Optimization-based separations for neural networks. In *Conference on Learning Theory*, pp. 3–64. PMLR, 2022.

Minhak Song, Kwangjun Ahn, and Chulhee Yun. Does sgd really happen in tiny subspaces? *arXiv preprint arXiv:2405.16002*, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Georgios Vlassis, David Belius, and Volodymyr Fomichov. A thorough reproduction and evaluation of mup. *Transactions on Machine Learning Research*, 2025.

Zhichao Wang, Denny Wu, and Zhou Fan. Nonlinear spiked covariance matrices and signal propagation in deep neural networks. In *The Thirty Seventh Annual Conference on Learning Theory*, pp. 4891–4957. PMLR, 2024.

Denny Wu and Ji Xu. On the optimal weighted ℓ_2 regularization in overparameterized linear regression. *Advances in Neural Information Processing Systems*, 33:10112–10123, 2020.

Greg Yang and Edward J Hu. Tensor programs iv: Feature learning in infinite-width neural networks. In *International Conference on Machine Learning*, pp. 11727–11737. PMLR, 2021.

Greg Yang and Etai Littwin. Tensor programs ivb: Adaptive optimization in the infinite-width limit. *arXiv preprint arXiv:2308.01814*, 2023.

Greg Yang, Edward J Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. *arXiv preprint arXiv:2203.03466*, 2022.

Greg Yang, Dingli Yu, Chen Zhu, and Soufiane Hayou. Tensor programs vi: Feature learning in infinite-depth neural networks. *arXiv preprint arXiv:2310.02244*, 2023.

CONTENTS

A BACKGROUND

Function Class Regularity Let $\mathcal{X}$ be a compact metric space, and let $C(\mathcal{X})$ denote the space of real-valued continuous functions on $\mathcal{X}$, equipped with the uniform norm:

$$\|f\|_{\sup} := \sup_{\boldsymbol{\nu}\in\mathcal{X}} |f(\boldsymbol{\nu})|.$$

We say a collection $\mathscr{F} \subset C(\mathcal{X})$ is:

- *uniformly bounded* if $\sup_{f\in\mathscr{F}} \|f\|_{\sup} \le K$ for some $K < \infty$,
- *uniformly equicontinuous* if for every $\varepsilon > 0$ there exists $\delta > 0$ such that

$$\|\boldsymbol{\nu} - \boldsymbol{\nu}'\| < \delta \implies |f(\boldsymbol{\nu}) - f(\boldsymbol{\nu}')| < \varepsilon \quad \text{for all } f \in \mathscr{F}.$$

We denote by $C^k(\mathcal{X})$ the space of k-times continuously differentiable functions. For $f \in C^k(\mathcal{X})$, the k-th derivative is written $f^{(k)}$. We define $f^{(0)} \equiv f$. In multivariate settings, this refers to the k-th total derivative.

Theorem 4 (Arzelà–Ascoli)**.** *Any uniformly bounded and uniformly equicontinuous collection $\mathscr{F} \subset C(\mathcal{X})$ is relatively compact in the uniform norm topology.*

Proposition 5. *Let $\{f_n\} \subset C^1(\mathcal{X})$ such that $f_n' \to g$ uniformly and $f_n(\boldsymbol{\nu}_0) \to L$ for some $\boldsymbol{\nu}_0 \in \mathcal{X}$ and $L \in \mathbb{R}$. Then $f_n \to f$ uniformly for some $f \in C^1(\mathcal{X})$, and $f' = g$.*

Proposition 6. *If $\{f_n\} \subset C^1(\mathcal{X})$ and the derivatives f_n' are uniformly bounded, then $\{f_n\}$ is uniformly equicontinuous.*

A.1 SCALING LIMITS AND TENSOR PROGRAMS

In this section we will recall some simplified background from (Yang & Hu, 2021; Yang & Littwin, 2023). For concreteness, we will fix the architecture to a L-hidden layer MLP, but all statements can be extended to a much more generic architectures (see Section 2.9.1 in (Yang & Littwin, 2023)). An L-hidden layer MLP of width n with nonlinearity $\phi : \mathbb{R} \to \mathbb{R}$ and no biases is parameterized by weight matrices $\boldsymbol{W}^1 \in \mathbb{R}^{n\times d}$, $\boldsymbol{W}^2, \dots, \boldsymbol{W}^L \in \mathbb{R}^{n\times n}$, and $\boldsymbol{W}^{L+1} \in \mathbb{R}^{1\times n}$. On an input $\boldsymbol{x} \in \mathbb{R}^d$, the network computes

$$\boldsymbol{h}^\ell(\boldsymbol{x}) = \boldsymbol{W}^\ell \boldsymbol{z}^\ell(\boldsymbol{x}) \in \mathbb{R}^n, \;\; \boldsymbol{z}^\ell(\boldsymbol{x}) = \phi(\boldsymbol{h}^\ell(\boldsymbol{x})) \in \mathbb{R}^n, \;\; \text{for } \ell = 1, \dots, L, \tag{4}$$

and the output is $f(\boldsymbol{x}) = \boldsymbol{W}^{L+1}\boldsymbol{z}^L(\boldsymbol{x}) \in \mathbb{R}$. Given N inputs $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$, we will abbreviate

$$\begin{aligned}
\boldsymbol{h}^\ell &:= [\boldsymbol{h}^\ell(\boldsymbol{x}_1) \mid \cdots \mid \boldsymbol{h}^\ell(\boldsymbol{x}_N)] \in \mathbb{R}^{n\times N}, \\
\boldsymbol{z}^\ell &:= [\boldsymbol{z}^\ell(\boldsymbol{x}_1) \mid \cdots \mid \boldsymbol{z}^\ell(\boldsymbol{x}_N)] \in \mathbb{R}^{n\times N}, \\
\boldsymbol{f} &:= (f(\boldsymbol{x}_1), \dots, f(\boldsymbol{x}_N)) \in \mathbb{R}^N.
\end{aligned}$$

abc-parameterization Assume that we train the network using SGD. We recall the definition of abc-parameterization from (Yang & Hu, 2021) (see Geiger et al. (2020); Chizat & Netrapalli (2024) for similar derivations). An abc-parameterization is a width-aware HP scaling specified by a set of HPs $\boldsymbol{\nu} = \{\alpha_\ell, \sigma_\ell, \eta_\ell\}_{\ell\in[L+1]}$ and HP scaling exponents $\boldsymbol{\gamma} = \{a_\ell, b_\ell, c_\ell\}_{\ell\in[L+1]}$ such that

(a) The weights $\boldsymbol{W}^\ell$ receive a multiplier $\alpha_\ell n^{-a_\ell}$,

(b) We initialize each $W^\ell_{\alpha\beta} \sim \mathcal{N}(0, \sigma_\ell^2 n^{-2b_\ell})$, and

(c) The SGD learning rate in layer ℓ is $\eta_\ell n^{-c_\ell}$.

Asymptotic Notation Given a sequence $\boldsymbol{x} = \{\boldsymbol{x}(n)\}_{n=1}^\infty$ of random tensors we write $\boldsymbol{x} = \Theta(n^{-a})$ and say that $\boldsymbol{x}$ has coordinates of size $\Theta(n^{-a})$ if there exist constants $A, B > 0$ such that, almost surely for sufficiently large n,

$$A \le \frac{1}{\#\boldsymbol{x}(n)} \sum_\alpha \boldsymbol{x}(n)_\alpha^2 \le B,$$

where $\#\boldsymbol{x}(n)$ denotes the number of entries in $\boldsymbol{x}(n)$. We use $O(n^{-a})$ and $\Omega(n^{-a})$ similarly.

Dynamical Dichotomy Theorem. We recall some definitions from (Yang & Littwin, 2023). To reflect the network after t steps of training we add a subscript t to the quantities in Eq. (4). We use Δ to denote a one-step difference of a time-dependent quantity. We say an abc-parameterization is

1. *stable at initialization* if
$$\boldsymbol{h}_0^\ell, \boldsymbol{z}_0^\ell = \Theta(1), \forall \ell \in [L], \text{ and } \boldsymbol{f}_0 = O(1).$$
2. *stable during training* if for any time $t \geq 0$ we have for any training routine
$$\Delta \boldsymbol{h}_t^\ell, \Delta \boldsymbol{z}_t^\ell = O(1), \forall \ell \in [L], \text{ and } \Delta \boldsymbol{f}_t = O(1).$$
3. *trivial* if for any time $t \geq 1$ and training routine, $\boldsymbol{f}_t - \boldsymbol{f}_0 \to 0$ almost surely as $n \to \infty$. We say the parameterization is *non-trivial* otherwise.
4. is in the *kernel regime* if there exists $\mathcal{K} : \mathbb{R}^N \to \mathbb{R}^N$ such that for every $t \geq 0$ and training routine, as $n \to \infty$,
$$\boldsymbol{f}_{t+1} - \boldsymbol{f}_t - \eta \mathcal{K}(\boldsymbol{f}_t) \to 0.$$
5. is *feature learning* if $\Delta \boldsymbol{z}_t^L = \Omega(1)$ for some training routine and $t \geq 0$.

Theorem 7 (Dynamical Dichotomy (Yang & Hu, 2021))**.** *A nontrivial and stable abc-parametrization either admits feature learning or is in the kernel regime but not both. The kernel regime does not admit feature learning, that is, for any training routine $\Delta \boldsymbol{z}_t^L \to 0$ for all $t \geq 0$.*

The μP and NTK parameterization are the maximal feature learning and kernel parameterizations respectively. All other such parameterizations can be obtained from one of these by setting some initialization or learning rate to zero (see Section 5.3 in (Yang & Hu, 2021) for more discussion). For adaptive optimizers such as Adam it is possible to extend the definitions to abcd-parameterizations (see Section 2.2 in (Yang & Littwin, 2023) for more details), for which a similar Dynamical Dichotomy theorem exists.

B Theoretical Results

B.1 Weak Transfer

In the following we provide a minimal set of technical conditions for the concept of HP transfer to be well-defined and align with empirical observations. We say that a function is *locally strongly convex* with parameters $\tau, \delta > 0$, if for every minimizer $\boldsymbol{\nu}^\star \in \arg\min f$,
$$|f(\boldsymbol{\nu}) - f(\boldsymbol{\nu}^\star)| \geq \frac{\tau}{2} \|\boldsymbol{\nu} - \boldsymbol{\nu}^\star\|^2, \text{ for all } \boldsymbol{\nu} \text{ such that } \|\boldsymbol{\nu} - \boldsymbol{\nu}^\star\| \leq \delta.$$

Recall that we assume that the HP search takes place over a *search space* $\mathcal{X}$:
$$\mathcal{X} = \prod_{i=1}^h [\ell_i, u_i], \quad \text{int}\,(\mathcal{X}) := \prod_{i=1}^h (\ell_i, u_i),$$
which is a h-dimensional box with bounds $\ell_i < u_i$ and interior $\text{int}\,(\mathcal{X})$.

Definition 3 (HP Transfer)**.** *The scaling $\boldsymbol{\gamma}$ admits HP transfer for a training procedure $\mathcal{A}$ and metric ϕ over a search space $\mathcal{X}$ if the following hold*

1. *$\phi_n \in C^2(\mathcal{X})$ is convex and has a unique minimizer $\boldsymbol{\nu}^\star(n)$.*
2. *$\phi_\infty := \lim_{n\to\infty} \phi_n$ is locally strongly convex and $\arg\min \phi_\infty \subseteq \text{int}\,(\mathcal{X})$.*
3. *The family $\{\phi_n''\}$ is uniformly bounded and uniformly equicontinuous.*

This definition implies the following desirable consequences.

Proposition 8 (HP Transfer Properties)**.** *In the context of Definition 3, the following are true*

1. *$\phi_\infty \in C^2(\mathcal{X})$ is convex and has a unique minimizer $\boldsymbol{\nu}^\star(\infty)$*
2. *$\phi_n \to \phi_\infty$, $\phi_n' \to \phi_\infty'$, $\phi_n'' \to \phi_\infty''$ uniformly and $\boldsymbol{\nu}^\star(n) \to \boldsymbol{\nu}^\star(\infty)$.*

Proof. By Proposition 6, it follows that for all $j \in \{0, 1, 2\}$, $\phi_n^{(j)}$ is uniformly equicontinuous. Since ϕ_n has a limit, it must converge uniformly ϕ_∞. Now repeatedly using Proposition 5 after passing to subsequences and invoking Arzela-Ascoli (Theorem 4), we see that $\phi_n^{(j)} \to \phi_\infty^{(j)}$ uniformly for $j \in \{1, 2\}$. Now it follows that $\phi_\infty \in C^2(\mathcal{X})$ since the uniform limit of continuous functions is continuous and ϕ_∞ is convex since convexity is preserved under pointwise limits. Note that the local strong convexity condition implies that ϕ_∞ has a unique minimizer. Now because $\mathcal{X}$ is compact, every subsequence of $\boldsymbol{\nu}^\star(n)$ has a convergent subsequence. By uniform convergence and the continuity of the ϕ_n and ϕ_∞, it follows that the limit of this subsequence is a minimizer of ϕ_∞. By the uniqueness of this minimizer, all the subsequences converge to $\boldsymbol{\nu}^\star(\infty)$ hence $\boldsymbol{\nu}^\star(n) \to \boldsymbol{\nu}^\star(\infty)$. □

This definition gives a minimal set of conditions under which the concept becomes mathematically coherent and aligns with observed empirical behavior. Requiring ϕ_n to have a unique minimizer removes ambiguity about which configuration should be transferred across scales. The convexity, smoothness, and equicontinuity conditions provide technical regularity that facilitates analysis and generally hold in practice.

The local strong convexity of ϕ_∞ ensures that performance meaningfully degrades away from the optimum. Without this condition, ϕ_∞ may be flat near its minimizer, making accurate hyperparameter selection potentially irrelevant in the large-n limit. The assumption that $\boldsymbol{\nu}^\star(\infty)$ lies in the interior of the search space ensures that this optimum remains unchanged under any enlargement of the domain; this condition rules out the n-dependent drift of the HPs of interest due to a "suboptimal" scaling – see Appendix A for discussions.

"Optimal" scaling limit. In (Yang et al., 2022; 2023), the authors remark that a key principle behind hyperparameter transfer is the "optimality" of the scaling. Heuristically if a scaling yields a suboptimal limit, then it cannot exhibit hyperparameter transfer since the HPs $\boldsymbol{\nu}$ need to undergo a n-dependent rescaling to "convert" the suboptimal scaling into the optimal scaling. The following proposition formalizes this intuition. The proposition requires $\mathbf{0} \in \mathcal{X}$ to avoid uninteresting cases where the optimal HP is zero which will generally not occur for optimization HPs in normal neural network training.

Proposition 9. *Let $\boldsymbol{\gamma}$ be a scaling that exhibits transfer over $\mathcal{X} = [0, u_1] \times \cdots [0, u_k]$ and all $\mathcal{X}'$ containing $\mathcal{X}$. Any other scaling $\boldsymbol{\gamma}' \neq \boldsymbol{\gamma}$ with these properties must satisfy*

$$\min_{\boldsymbol{\nu} \in \mathcal{X}} \phi_\infty(\boldsymbol{\nu}; \boldsymbol{\gamma}) = \min_{\boldsymbol{\nu}' \in \mathcal{X}'} \phi_\infty(\boldsymbol{\nu}'; \boldsymbol{\gamma}').$$

Proof of Proposition 9. For brevity define

$$\boldsymbol{\nu}_\star := \arg\min_{\boldsymbol{\nu} \in \mathcal{X}} \phi_\infty(\boldsymbol{\nu}; \boldsymbol{\gamma}) \text{ and } \boldsymbol{\nu}'_\star := \arg\min_{\boldsymbol{\nu}' \in \mathcal{X}} \phi_\infty(\boldsymbol{\nu}'; \boldsymbol{\gamma}').$$

For the sake of contradiction assume that $\phi_\infty(\boldsymbol{\nu}_\star; \boldsymbol{\gamma}) > \phi_\infty(\boldsymbol{\nu}'_\star; \boldsymbol{\gamma}')$. For large enough n, we will have $\phi_n(\boldsymbol{\nu}_\star; \boldsymbol{\gamma}) > \phi_n(\overline{\boldsymbol{\nu}}_\star; \boldsymbol{\gamma})$ where $\overline{\boldsymbol{\nu}}_\star = \boldsymbol{\nu}'_\star \odot (n^{\gamma_1 - \gamma'_1}, \ldots, n^{\gamma_k - \gamma'_k})$ and $\boldsymbol{\nu}_\star \neq \overline{\boldsymbol{\nu}}_\star$ which is a contradiction. The case $\phi_\infty(\boldsymbol{\nu}_\star; \boldsymbol{\gamma}) < \phi_\infty(\boldsymbol{\nu}'_\star; \boldsymbol{\gamma}')$ follows analogously. □

The dynamical dichotomy theorem states that all scalings induced by abcd-parameterizations except for μP lead to optimization degeneracies or non-feature learning behavior. Therefore the proposition implies that if HP transfer is possible with μP and feature learning is advantageous, then it should be only possible using μP. Of course, it is still a challenging problem to rigorously characterize when μP will exhibit transfer and when feature learning is actually advantageous.

B.2 Asymptotic Rates

Recall the definitions of the quantities a_n, b_n, c_n from Definition 1. We will need to reason about the following quantity which we call the *uniform loss gap*.

$$\bar{a}_n := \|\phi_n - \phi_\infty\|_{\sup}. \tag{5}$$

Using the local strong convexity assumption, we will be able to directly relate the HP gap with the uniform loss gap. We first prove a convenient lemma which bounds the minimizer displacement in terms of the sup-norm of the perturbation.

Lemma 10. *Let $f : \mathcal{X} \to \mathbb{R}$ and $g : \mathcal{X} \to \mathbb{R}$ such that f has a unique minimizer $\boldsymbol{x}_f$ and g has a unique minimizer $\boldsymbol{x}_g$, and g is τ strongly-convex*

$$g(\boldsymbol{x}) - g(\boldsymbol{x}_g) \geq \frac{\tau}{2}\|\boldsymbol{x} - \boldsymbol{x}_g\|^2, \forall \boldsymbol{x} \in \mathcal{X}$$

for some $\tau > 0$. If $\|f - g\|_{\sup} \leq \varepsilon$, then

$$\|\boldsymbol{x}_f - \boldsymbol{x}_g\| \leq 2\left(\frac{\varepsilon}{\tau}\right)^{1/2}.$$

Proof. Note that $g(\boldsymbol{x}_f) - \varepsilon \leq f(\boldsymbol{x}_f) \leq f(\boldsymbol{x}_g) \leq g(\boldsymbol{x}_g) + \varepsilon$, hence

$$g(\boldsymbol{x}_f) - g(\boldsymbol{x}_g) \leq 2\varepsilon.$$

By strong convexity, $2\varepsilon \geq \frac{\tau}{2}\|\boldsymbol{x}_f - \boldsymbol{x}_g\|^2$, which after rearranging gives the desired conclusion. □

The above lemma, along with Propositions 8 and Taylor's theorem immediately yields the following.

Lemma 11. *Assume HP transfer holds (Def. 3), then $b_n = O\left(\bar{a}_n^{1/2}\right)$ and $c_n = \Theta(b_n^2)$.*

To relate the loss gap a_n and the uniform loss gap $\bar{a}_n$ we must make further assumptions. Intuitively, we would like to capture the fact that typically the uniform loss gap is dominated by a fairly uniform, positive loss gap for HPs which are nearly optimal.

Definition 4. *We will say that the uniform loss gap is locally tight if there exists some radius $\bar{r}$ and constants $0 < c \leq C$ such that for all $\boldsymbol{\nu} \in B(\boldsymbol{\nu}^\star(\infty), \bar{r})$,*

$$c\bar{a}_n \leq \phi_n(\boldsymbol{\nu}) - \phi_\infty(\boldsymbol{\nu}) \leq C\bar{a}_n.$$

This essentially states that the uniform loss gap tightly controls the convergence rate for any nearly optimal set of HPs. This corresponds with the empirical observation that nearly optimal hyperparameters obey identical scaling laws. Under this assumption it is easy to see that $a_n = \Theta(\bar{a}_n)$.

Lemma 12. *Assume HP transfer holds (Def. 3). If the uniform loss gap $\bar{a}_n$ is locally tight then $a_n = \Theta(\bar{a}_n)$.*

Proof. Note that we have

$$\begin{aligned}
\phi_n(\boldsymbol{\nu}^\star(n)) - \phi_\infty(\boldsymbol{\nu}^\star(\infty)) &= \phi_n(\boldsymbol{\nu}^\star(n)) - \phi_\infty(\boldsymbol{\nu}^\star(n)) + \phi_\infty(\boldsymbol{\nu}^\star(n)) - \phi_\infty(\boldsymbol{\nu}^\star(\infty)) \\
&\geq \phi_n(\boldsymbol{\nu}^\star(n)) - \phi_\infty(\boldsymbol{\nu}^\star(n)), \\
\phi_n(\boldsymbol{\nu}^\star(n)) - \phi_\infty(\boldsymbol{\nu}^\star(\infty)) &= \phi_n(\boldsymbol{\nu}^\star(n)) - \phi_n(\boldsymbol{\nu}^\star(\infty)) + \phi_n(\boldsymbol{\nu}^\star(\infty)) - \phi_\infty(\boldsymbol{\nu}^\star(\infty)) \\
&\leq \phi_n(\boldsymbol{\nu}^\star(\infty)) - \phi_\infty(\boldsymbol{\nu}^\star(\infty)).
\end{aligned}$$

Since $\boldsymbol{\nu}^\star(n) \to \boldsymbol{\nu}^\star(\infty)$ we can apply the inequalities in Definition 4 for large enough n to yield the claim. □

Proof of Proposition 1. This follows directly by applying Lemmas 11 and 12 □

B.3 Grid Search

We now turn towards the connection between the previous asymptotic quantities and compute-optimal grid search. Define a *grid* $\mathcal{G}$ in a search space $\mathcal{X}$ to be a collection of points $\{\boldsymbol{\nu}^{(1)}, \ldots, \boldsymbol{\nu}^{(M)}\}$ contained in $\mathcal{X}$. The *grid resolution* $\rho(\mathcal{G}, \mathcal{X})$ is defined as the largest distance of a point in $\mathcal{X}$ to a point in $\mathcal{G}$, that is

$$\rho(\mathcal{G}, \mathcal{X}) := \sup_{\boldsymbol{\nu} \in \mathcal{X}} \min_{\boldsymbol{\nu}' \in \mathcal{G}} \|\boldsymbol{\nu} - \boldsymbol{\nu}'\|.$$

For a grid $\mathcal{G}$ in the search space $\mathcal{X}$, define $\boldsymbol{\nu}^\star(n, \mathcal{G}) = \arg\min_{\boldsymbol{\nu} \in \mathcal{G}} \phi_n(\boldsymbol{\nu})$. Let us assume that we are allocated a flops budget $\mathcal{F}$ in order to perform hyperparameter search and produce a final model.

Recall that for brevity we use $f(x) \sim g(x)$ to mean $f(x) = \Theta(g(x))$. For a grid $\mathcal{G}$ of resolution ρ, we will make the following convenience assumption for Theorems 13 and 14.

Assumption 1 (Grid proximity). *For a grid $\mathcal{G}$ of resolution ρ, we assume that*

$$\min_{\boldsymbol{\nu}\in\mathcal{G}} \|\boldsymbol{\nu} - \boldsymbol{\nu}^\star(n)\| \sim \rho.$$

This assumption is morally true if $\mathcal{G}$ is not chosen with knowledge of the location of $\boldsymbol{\nu}^\star(\infty)$. If for a given $\mathcal{G}$ we chose $\boldsymbol{\nu}^\star(\infty)$ uniformly at random within $\mathcal{G}$, then this assumption holds on average. Suppose we have a compute budget of $\mathcal{F}$ flops, and the amount of flops needed for a single training run scales with n^r for models of width n, where $r = 2$ for standard optimization algorithms on standard architectures. For a scaling $\boldsymbol{\gamma}$, we will evaluate the quality of a set of HPs $\boldsymbol{\nu}$ by performing a full training run for a certain width n using the scaled hyperparameters $\mathcal{H}_n(\boldsymbol{\nu}, \boldsymbol{\gamma})$. The details about the grid search are discussed in Appendix B.3. Let us assume that the number of HPs we perform a grid search over is h. We first consider compute optimal performance when directly tuning the HPs on a large model.

Theorem 13. *Suppose that $a_n \sim n^{-\alpha}$. Given a compute budget of $\mathcal{F}$ flops, if we directly conduct grid search on a width-n model, the compute-optimal performance scales as $\mathcal{F}^{\frac{-2\alpha}{h\alpha+2r}}$ and is obtained at width $n^\star \sim \mathcal{F}^{\frac{2}{h\alpha+2r}}$.*

Proof of Theorem 13. For a grid of resolution $\rho = \rho(\mathcal{G}, \mathcal{X})$ we will have $|\mathcal{G}| \sim \rho^{-h}$ and $\mathcal{F} \sim n^r \rho^{-h}$. Now observe that by uniform convergence of derivatives (Prop. 8), for n large enough ϕ_n will satisfy (τ', δ')-LSC for some constants $\tau', \delta' > 0$ and so $\phi_n(\boldsymbol{\nu}^\star(n, \mathcal{G})) - \phi_n(\boldsymbol{\nu}^\star(n)) \sim \rho^2$ by Assumption 1. Therefore,

$$\begin{aligned}
\phi_n(\boldsymbol{\nu}^\star(n, \mathcal{G})) - \phi_\infty^\star &= \phi_n(\boldsymbol{\nu}^\star(n, \mathcal{G})) - \phi_n(\boldsymbol{\nu}^\star(n)) + \phi_n(\boldsymbol{\nu}^\star(n)) - \phi_\infty^\star \\
&\sim \rho^2 + n^{-\alpha} \\
&\sim n^{2r/h}\mathcal{F}^{-2/h} + n^{-\alpha}.
\end{aligned}$$

We see the final expression is minimized by taking $n^\star \sim \mathcal{F}^{\frac{2}{h\alpha+2r}}$ which yields the rate

$$\phi_n(\boldsymbol{\nu}^\star(n, \mathcal{G})) - \phi_\infty^\star \sim \mathcal{F}^{-\frac{2\alpha}{h\alpha+2r}},$$

as claimed. □

Now consider the strategy of transferring the optimal HPs from a smaller model. We say that transfer is *useful* if this strategy achieves a better loss scaling than directly tuning the large model under the same compute budget, as specified above.

Theorem 14. *Suppose $a_n \sim n^{-\alpha}$ and $b_n \sim n^{-\beta}$, and we conduct a grid search on a width-n model and then transfer to a large width-M model. Given a compute budget of $\mathcal{F}$ flops, the compute-optimal performance scales as $\mathcal{F}^{\frac{-\alpha}{r}} + \mathcal{F}^{\frac{-2\beta}{h\beta+r}}$, obtained at widths $n^\star \sim \mathcal{F}^{\frac{1}{h\beta+r}}$ and $M^\star \sim \mathcal{F}^{1/r}$. Transfer is useful iff $\beta > \alpha/2$.*

Proof of Theorem 14. Note that in this setting $\mathcal{F} \sim n^r\rho^{-h} + M^r$. The performance scaling is

$$\begin{aligned}
\phi_M(\boldsymbol{\nu}^\star(n, \mathcal{G})) - \phi_\infty^\star &= \phi_M(\boldsymbol{\nu}^\star(n, \mathcal{G})) - \phi_M(\boldsymbol{\nu}^\star(n)) \\
&\quad + \phi_M(\boldsymbol{\nu}^\star(n)) - \phi_M(\boldsymbol{\nu}^\star(M)) \\
&\quad + \phi_M(\boldsymbol{\nu}^\star(M)) - \phi_\infty^\star \\
&\sim \rho^2 + n^{-2\beta} + M^{-\alpha} \\
&\sim \left(\frac{n^r}{\mathcal{F} - M^r}\right)^{2/h} + n^{-2\beta} + M^{-\alpha}.
\end{aligned}$$

Since $\mathcal{F} - M^r \sim \mathcal{F}$, we should take $M^\star \sim \mathcal{F}^{1/r}$ in which case the above simplifies to

$$\phi_M(\boldsymbol{\nu}^\star(n, \mathcal{G})) - \phi_\infty^\star \sim \frac{n^{2r/h}}{\mathcal{F}^{2/h}} + n^{-2\beta} + \mathcal{F}^{-\alpha/r}$$

which is minimized a $n^\star \sim \mathcal{F}^{\frac{1}{h\beta+r}}$ and yields $\phi_M(\boldsymbol{\nu}^\star(n, \mathcal{G})) - \phi_\infty^\star \sim \mathcal{F}^{\frac{-2\beta}{h\beta+r}} + \mathcal{F}^{\frac{-\alpha}{r}}$. Now note that $\frac{\alpha}{r} > \frac{2\alpha}{h\alpha+2r}$ and $\frac{2\beta}{h\beta+r} > \frac{2\alpha}{h\alpha+2r}$ if and only if $\beta > \alpha/2$, which is the condition for useful transfer. □

B.4 Random Features Regression

Consider the following data generating process where the labels come from a single-index model, and we train a random features ridge regression estimator on N samples,

$$y = \sigma_*(\langle \boldsymbol{x}, \boldsymbol{\beta}_* \rangle) + \varepsilon, \quad \text{where } \boldsymbol{x} \sim \mathcal{N}(0, \boldsymbol{I}_d),\ \|\boldsymbol{\beta}_*\| = 1,\ \mathrm{Var}(\varepsilon) = \sigma_\varepsilon^2.$$
$$f(\boldsymbol{x}) = \langle \boldsymbol{a}_\lambda, \sigma(\boldsymbol{W}\boldsymbol{x}) \rangle, \quad \text{where } \boldsymbol{W} \in \mathbb{R}^{d\times n},\ [\boldsymbol{W}]_{i,j} \sim \mathcal{N}(0, 1/d),$$
$$\boldsymbol{a}_\lambda := \mathrm{argmin}_{\boldsymbol{a}\in\mathbb{R}^n} \textstyle\sum_{i=1}^N (y_i - \langle \boldsymbol{a}, \sigma(\boldsymbol{W}\boldsymbol{x}_i) \rangle)^2 + \lambda \|\boldsymbol{a}\|_2^2.$$

We aim to select the optimal regularization parameter λ that minimizes the prediction risk (generalization error) $\mathcal{R} = \mathbb{E}_{\boldsymbol{x}}[(y - f(\boldsymbol{x}))^2]$. We make the following assumptions.

Assumption 2.

- ***Proportional limit.*** *$N, d, n \to \infty, N/d \to \psi_1, n/d \to \psi_2$ where $\psi_1, \psi_2 \in (0, \infty)$.*
- ***Normalized activation.*** *Both the student and teacher nonlinearities are normalized such that $\mathbb{E}[\sigma], \mathbb{E}[\sigma_*] = 0$, $\|\sigma\|_\gamma, \|\sigma_*\|_\gamma = 1$, and also $\|\sigma'\|_\gamma, \|\sigma_*'\|_\gamma \neq 0$. We further require that σ is a nonlinear odd function with bounded first three derivatives, and σ_* is $\Theta(1)$-Lipschitz.*

Remark 1. *The above assumptions are standard in the high-dimensional asymptotic analysis of random features models, see e.g., Mei & Montanari (2022); Gerace et al. (2020). The non-zero expectation of σ', σ_*' is necessary for the RF model to outperform the null estimator in the proportional regime. The assumption of odd σ simplifies the Gaussian equivalence computation — see Hu & Lu (2022); Ba et al. (2022).*

Asymptotic prediction risk. Under Assumption 2, following Hu & Lu (2022), we know that the asymptotic prediction risk is given as by the following implicit equations,

$$\lim_{n,d,N\to\infty} \mathbb{E}[(y - f(\boldsymbol{x}))^2] =: \mathcal{R}(\lambda) = -(\mu_2^{*2} + \sigma_\varepsilon^2) \cdot \frac{m_1'(\lambda)}{m_1(\lambda)^2} - \mu_1^{*2} \cdot \frac{m_2'(\lambda)}{m_1(\lambda)^2},$$

where the Hermite coefficients $\mu_1^* = \mathbb{E}_{z\sim\mathcal{N}(0,1)}[\sigma_*'(z)], \mu_2^{*2} = 1 - \mu_1^{*2}$, and the coupled Stieltjes transforms $m_1(z)$ and $m_2(z) \in \mathbb{C}^+ \cup \mathbb{R}_+$ are uniquely defined by the following self-consistent equations for $z \in \mathbb{C}^+ \cup \mathbb{R}_+$,

$$\frac{1}{\psi_1}(m_1(z) - m_2(z))(\mu_2^2 m_1(z) + \mu_1^2 m_2(z)) + \mu_1^2 m_1(z) m_2(z)\,(z m_1(z) - 1) = 0, \tag{6}$$
$$\frac{\psi_2}{\psi_1}\left(\mu_1^2 m_1(z) m_2(z) + \frac{1}{\psi_1}(m_2(z) - m_1(z))\right) + \mu_1^2 m_1(z) m_2(z)\,(z m_1(z) - 1) = 0, \tag{7}$$

where $\mu_1 = \mathbb{E}_{z\sim\mathcal{N}(0,1)}[\sigma'(z)], \mu_2^2 = 1 - \mu_1^2$. Note that σ being nonlinear implies $\mu_2 \neq 0$. We omit the argument in $m_1(\lambda), m_2(\lambda)$ except when tracking the λ-dependence. m_1', m_2' stand for derivative with respect to λ. To further simplify the exposition, we define $\eta := \psi_1/\psi_2$, and write the asymptotic prediction risk at width ψ_2 and ridge penalty λ as $\mathcal{R}_{\psi_2}(\lambda) = \mathcal{R}_{\psi_1/\eta}(\lambda)$.

Large-width limit. First we consider the test performance of the "infinite-width" model, which corresponds to taking $\psi_2 = n/d \to \infty$ or $\eta \to 0$. Note that the prediction risk in this limit is well-defined and has been computed in prior works (see e.g., Bartlett et al. (2021)). First recall that $m_1, m_2 > 0$ and $z m_1(z) - 1$ remains uniformly bounded for any $1/\eta$, hence from (7) we know that at the large-width limit,

$$\mu_1^2 m_1 m_2 + \psi_1^{-1}(m_2 - m_1) =: \mathscr{T}_1(m_1, m_2) = 0, \quad \lambda m_1 + \mu_2^2 m_1 + \mu_1^2 m_2 - 1 =: \mathscr{T}_2(m_1, m_2, \lambda) = 0.$$

Reparameterize $t := 1 + \mu_1^2 \psi_1 m_1 > 1$, we have $\lambda(t) = \frac{\mu_1^2 \psi_1}{t-1} - \frac{\mu_1^2}{t} - \mu_2^2$. By the chain rule,

$$\partial_t m_1 = \frac{1}{\mu_1^2 \psi_1}, \quad \partial_t m_2 = \frac{1}{\mu_1^2 \psi_1 t^2}, \quad \partial_t \lambda = -\frac{\mu_1^2 S(t)}{t^2 (t-1)^2},$$

where we defined $S(t) := \psi_1 t^2 - (t-1)^2$. Hence at $\eta = 0$ we have

$$\frac{m_1'}{m_1^2} = -\frac{\psi_1 t^2}{S(t)}, \quad \frac{m_2'}{m_1^2} = -\frac{\psi_1}{S(t)}.$$

Therefore,

$$\mathcal{R}_\infty(\lambda(t)) := \lim_{\psi_2\to\infty} \mathcal{R}_{\psi_2}(\lambda(t)) = \frac{\psi_1((\mu_2^{*2}+\sigma_\varepsilon^2)t^2+\mu_1^{*2})}{S(t)}.$$

Differentiating the risk yields the closed-form expression of the optimal ridge penalty (consistent with Dobriban & Wager (2018); Wu & Xu (2020)),

$$\lambda^*(\infty) = \frac{\mu_1^2(\sigma_\varepsilon^2+\mu_2^{*2})}{\mu_1^{*2}} - \mu_2^2. \tag{8}$$

We restrict ourself to the setting where non-vanishing regularization is needed at the large-width limit, i.e., $\lambda^*(\infty) > 0$. Denote t_* as the corresponding optimal value of $t = 1+\mu_1^2\psi_1 m_1$, the optimality condition $\mu_1^{*2} = \frac{(\mu_2^{*2}+\sigma_\varepsilon^2)t_*(t_*-1)}{\psi_1 t_* - t_* + 1}$ implies that

$$\psi_1 t_* - t_* + 1 > 0, \quad S(t_*) = (\psi_1 t_* - t_* + 1)t_* + (t_* - 1) > 0.$$

Hence we have the following characterization of the curvature

$$\mathcal{R}''_\infty(\lambda^*(\infty)) = \frac{2(\mu_2^{*2}+\sigma_\varepsilon^2)\psi_1}{S(t_*)\lambda'(t_*)^2(\psi_1 t_* - t_* + 1)} > 0, \quad \lambda'(t_*) = -\frac{\mu_1^2 S(t_*)}{t_*^2(t_*-1)^2} < 0. \tag{9}$$

Note that (9) validates the local strong convexity of $\mathcal{R}_\infty$.

Finite-width sensitivity. Now consider the system given by (6)(7)

$$E(m_1, m_2, \lambda, \eta) := \begin{bmatrix} \mathscr{T}_2(m_1, m_2, \lambda) \\ \mathscr{T}_1(m_1, m_2) + \eta \mathscr{T}_3(m_1, m_2, \lambda) \end{bmatrix},$$

where $\mathscr{T}_3 = \mu_1^2 m_1 m_2(\lambda m_1 - 1)$. Differentiate $E = 0$ with respect to η and evaluate at $\eta = 0$,

$$J_0 \begin{bmatrix} \partial_\eta m_1 \\ \partial_\eta m_2 \end{bmatrix} = -\begin{bmatrix} 0 \\ T \end{bmatrix}\Big|_{\eta=0}, \quad \text{where} \quad J_0 := \partial_{m_1,m_2}(\mathscr{T}_2, \mathscr{T}_1) = \begin{bmatrix} \lambda+\mu_2^2 & \mu_1^2 \\ \mu_1^2 m_2 - \psi_1^{-1} & \mu_1^2 m_1 + \psi_1^{-1} \end{bmatrix}. \tag{10}$$

Recall that $\mathscr{T}_2 = 0$ yields $\lambda m_1 - 1 = -(\mu_2^2 m_1 + \mu_1^2 m_2)$, and hence $T|_{\eta=0} = -\mu_1^2 m_1 m_2(\mu_2^2 m_1 + \mu_1^2 m_2)$. On the other hand, by direct computation

$$\det J_0 = \frac{\mu_1^2 S(t)}{\psi_1 t(t-1)}, \quad \Rightarrow \quad \det J_0(t_*) > 0. \tag{11}$$

Solving the linear system (10) yields

$$\begin{aligned} \partial_\eta m_1 &= -\frac{\mu_1^4 m_1 m_2(\mu_2^2 m_1 + \mu_1^2 m_2)}{\det J_0} = -\frac{(t-1)^4(\mu_1^2+\mu_2^2 t)}{\mu_1^4\psi_1^2 t S(t)}, \\ \partial_\eta m_2 &= -\frac{(\lambda+\mu_2^2)\mu_1^2 m_1 m_2(\mu_2^2 m_1 + \mu_1^2 m_2)}{\det J_0} = \frac{(t-1)^3(\mu_1^2+\mu_2^2 t)(\psi_1 t - t + 1)}{\mu_1^4\psi_1^2 t^2 S(t)}. \end{aligned}$$

Differentiating the prediction risk with respect to η and evaluate at the large-width limit $\eta = 0$,

$$\partial_{\eta=0}\mathcal{R}_\infty(\lambda) = -(\mu_2^{*2}+\sigma_\varepsilon^2)\left(\frac{\partial_\eta m_1'}{m_1^2} - \frac{2m_1'\partial_\eta m_1}{m_1^3}\right) - \mu_1^{*2}\left(\frac{\partial_\eta m_2'}{m_1^2} - \frac{2m_2'\partial_\eta m_1}{m_1^3}\right),$$

where $\partial_{\eta=0}\mathcal{R}_\infty(\lambda) = \partial_\eta \mathcal{R}_{\psi_1/\eta}(\lambda)\big|_{\eta=0}$. A similar determinant calculation yields

$$\partial_\eta m_1' = -\frac{\mu_1^2\mathscr{S}}{\det J_0}, \quad \partial_\eta m_2' = -\frac{(\lambda+\mu_2^2)\mathscr{S}}{\det J_0},$$

where

$$\begin{aligned} \mathscr{S} :=& \mu_1^2 m_2(2\lambda m_1 - 1)m_1' + \mu_1^2 m_1(\lambda m_1 - 1)m_2' + \mu_1^2 m_1^2 m_2 \\ =& \frac{(t-1)^3}{\mu_1^4\psi_1^2}\left[-\frac{t}{S(t)} + \frac{(2t+1)}{S(t)}\cdot\frac{t-1}{\mu_1^2\psi_1}\cdot\frac{\mu_1^2+\mu_2^2 t}{t}\right] + \frac{(t-1)^3}{\mu_1^4\psi_1^3 t}. \end{aligned}$$

Using the above, a tedious algebraic calculation gives the following expression of the sensitivity of the prediction risk (at fixed λ) with respect to η:

$$\partial_{\eta=0}\mathcal{R}_\infty(\lambda) = \frac{(t-1)^2 Q(t)}{\mu_1^2 S(t)^2}, \tag{12}$$

where $Q(t) = \sum_{k=0}^{2} c_k t^k$ with coefficients

$$\begin{aligned}
c_2 &= \mu_1^2(\mu_2^{*2}+\sigma_\varepsilon^2) - \mu_2^2(\mu_2^{*2}+\sigma_\varepsilon^2) + 2\mu_2^2\mu_1^{*2}\,(\psi_1 - 1),\\
c_1 &= -3\mu_1^2(\mu_2^{*2}+\sigma_\varepsilon^2) + \mu_1^2\mu_1^{*2}(\psi_1-1) + \mu_2^2(\mu_2^{*2}+\sigma_\varepsilon^2) + \mu_2^2\mu_1^{*2}(\psi_1+3),\\
c_0 &= 2\mu_1^2(\mu_2^{*2}+\sigma_\varepsilon^2) + \mu_1^{*2}\,(2\mu_1^2\psi_1 + 2\mu_1^2 - 1).
\end{aligned}$$

Importantly, under the optimal λ defined in (8), we have the factorization

$$Q(t_*) = \frac{2(\mu_2^{*2}+\sigma_\varepsilon^2)(\mu_1^2+\mu_2^2 t_*)(t_*-1)S(t_*)}{\psi_1 t_* - t_* + 1},$$

and since $(t_*-1), S(t_*), (\psi_1 t_* - t_* + 1) > 0$, we conclude the derivative (12) at t_* is strictly positive

$$\partial_{\eta=0}\mathcal{R}_\infty(\lambda_*(\infty)) = \frac{2(\mu_2^{*2}+\sigma_\varepsilon^2)(\mu_1^2+\mu_2^2 t_*)(t_*-1)^3}{\mu_1^2 S(t_*)(\psi_1 t_* - t_* + 1)} =: C_\eta > 0. \tag{13}$$

Putting things together. Recall that the asymptotic prediction risk $\mathcal{R}$ is C^2 in λ and C^1 in η. Given the Jacobian invertibility (11) and local strong convexity (9), the implicit function theorem (IFT) implies that there exists a neighborhood defined by some $\eta_0 > 0$ and a unique C^1 map $\bar{\lambda}^* : [0, \eta_0) \to \mathbb{R}_+$, such that $\bar{\lambda}^*(0) = \lambda^*(\infty)$, $\partial_\lambda \mathcal{R}_\eta(\bar{\lambda}^*(\eta)) = 0$, and $\partial_\lambda^2 \mathcal{R}_\eta(\bar{\lambda}^*(\eta)) > 0$. Consequently, we may take a first-order expansion and conclude (setting $\eta = \psi_1/\psi_2$ under with ψ_1)

$$\lambda^*(\psi_2) = \lambda^*(\infty) + \partial_{\eta\to 0_+}\bar{\lambda}^*(\eta)\cdot\frac{\psi_1}{\psi_2} + o(\psi_2^{-1}), \quad \partial_{\eta\to 0_+}\bar{\lambda}^*(\eta) := -\frac{\partial_\lambda\partial_\eta\mathcal{R}_\infty(\lambda^*(\infty))}{\partial_\lambda^2\mathcal{R}_\infty(\lambda^*(\infty))} =: C_\lambda. \tag{14}$$

Note that the denominator in C_λ is strictly positive by (9). Moreover, since $\lambda'(t_*) \neq 0$, we may write $\partial_\lambda\partial_\eta\mathcal{R}_\infty(\lambda) = \frac{\partial_t(\partial_{\eta=0}\mathcal{R}(\lambda(t)))}{\partial_t\lambda(t)}$, and compute $C_\lambda = \frac{(3\psi_1 - 4\mu_1^2\psi_1+1)t_*^2 + 2(2\mu_1^2\psi_1 - 1)t_* + 1}{2\psi_1 t_*^2}$. This confirms that the hyperparameter gap vanishes at a rate of $O(\psi_2^{-1})$.

For the loss gap, denote $\delta_\eta = \lambda^*(\psi_1/\eta) - \lambda^*(\infty)$ for $\eta \in [0, \eta_0)$, the IFT and Taylor expansion gives

$$\begin{aligned}
\mathcal{R}_{\psi_1/\eta}(\lambda^*(\infty)+\delta_\eta) &= \mathcal{R}_\infty(\lambda^*(\infty)) + \partial_{\lambda=\lambda^*(\infty)}\mathcal{R}_\infty(\lambda^*(\infty))\delta_\eta + \partial_{\eta=0}\mathcal{R}_\infty(\lambda^*(\infty)) + O(\delta_\eta^2+\eta^2)\\
&\overset{(i)}{=} \mathcal{R}_\infty(\lambda^*(\infty)) + C_\eta\cdot\frac{\psi_1}{\psi_2} + o(\psi_2^{-1}),
\end{aligned}$$

where (i) is due to the stationarity condition $\partial_{\lambda=\lambda^*(\infty)}\mathcal{R}_\infty(\lambda^*(\infty)) = 0$ and $\delta_\eta = O(\eta)$ from (14), and $C_\eta > 0$ is explicitly given in (13). The strict positivity of C_η ensures that the loss gap scales exactly as $\Theta(\psi_2^{-1})$. Hence by Proposition 1 and Theorem 14 we know that the ridge penalty in RF regression exhibits *fast and useful transfer*, i.e., the suboptimality gap $|\mathcal{R}_\infty(\lambda^*(\psi_2)) - \mathcal{R}_\infty(\lambda^*(\infty))| \sim \psi_2^{-2} \ll |\mathcal{R}_{\psi_2}(\lambda^*(\psi_2)) - \mathcal{R}_\infty(\lambda^*(\infty))|$, which aligns with the observations in Figure 2 and concludes Proposition 3.

B.5 Fast Transfer

In this section we formalize our quantitative bound on the HP gap b_n in terms of the top-κ invariance and residual flatness arising from our decomposition. For the sake of simplicity we will assume that $\mathcal{X}$ is small enough so that the local strong convexity condition holds globally.

Definition 5. *For $f \in C^2(\mathcal{X})$, define the curvature $\mu(f)$ and the Lipschitz constant $\mathrm{Lip}(f)$ as:*

$$\mu(f) := \inf_{\boldsymbol{\nu} \in \mathcal{X}} f''(\boldsymbol{\nu}), \;\; \mathrm{Lip}(f) := \sup_{\boldsymbol{\nu} \in \mathcal{X}} |f'(\boldsymbol{\nu})|.$$

Definition 6 (Decomposition Rate)**.** *Let ϕ_n and $\mathcal{X}$ be as in Definition 3, and ϕ_n^κ and $\boldsymbol{\nu}_\kappa^\star(n)$ as in Eq. (3). We define the following quantities associated with the decomposition:*

- ***Top-κ invariance gap:*** $\varepsilon_{\mathrm{inv}}(n, \kappa) := \frac{\|\phi_n^\kappa - \phi_\infty^\kappa\|_{\sup}}{\mu(\phi_n^\kappa) \vee \mu(\phi_\infty^\kappa)}$
- ***Residual flatness gap:*** $\varepsilon_{\mathrm{flat}}(n, \kappa) := \frac{\mathrm{Lip}(\phi_n^{-\kappa})}{\mu(\phi_n)} + \frac{\mathrm{Lip}(\phi_\infty^{-\kappa})}{\mu(\phi_\infty)}$
- ***Decomposition objective:*** $\mathcal{J}(\kappa) := 2\sqrt{\varepsilon_{\mathrm{inv}}(n, \kappa)} + \varepsilon_{\mathrm{flat}}(n, \kappa)$
- ***Decomposition HP gap:*** $t_n := \min_\kappa \mathcal{J}(\kappa)$ s.t. $\mu(\phi_n^\kappa) \wedge \mu(\phi_\infty^\kappa) > 0$.

The decomposition HP gap t_n is defined to be a natural upper bound on the HP gap b_n as we show in Proposition 15 which makes use of Lemmas 10 and 16. This upper bound is obtained by choosing an optimal HP dependent truncation index $\kappa_n^\star$ minimizing $\mathcal{J}(\kappa)$ such that $\varepsilon_{\mathrm{inv}}(n, \kappa_n^\star)$ which quantifies top-κ invariance and $\varepsilon_{\mathrm{flat}}(n, \kappa_n^\star)$ which quantifies residual flatness are both appropriately small. Note that t_n is well-defined for n large enough because we can take $\kappa \equiv n$ and from the assumptions of Definition 3 both ϕ_n and ϕ_∞ are strongly convex.

Proposition 15. *Assume the setting of Definition 3 where the local strong convexity is global. The decomposition HP gap t_n in Definition 6 satisfies $t_n \geq b_n$ where b_n is the HP gap from Definition 1.*

We remark that we introduce the quantity t_n primarily as a theoretical quantity for conceptual purposes. The quantity t_n will be small when top-κ invariance and residual flatness holds and since $t_n \geq b_n$ this will imply b_n is small as well. We also note that it is natural to chose the optimal truncation index $\kappa_n^\star$ used in t_n to be a function of the width n. This is because as $n \to \infty$ we expect $b_n \to 0$ and so it is desirable that $t_n \to 0$ as well which will not be the case if we used a fixed κ because $\varepsilon_{\mathrm{flat}}$ in $\mathcal{J}(\kappa)$ will converge to a non-zero value. Overall, one can view t_n being small[3] as an explicit sufficient condition for fast transfer. We conjecture that (some version of) this condition holds in practice when training neural networks on natural data with optimizers like Adam or SGD. For future work, it would be interesting to provide natural settings where such a formal statement is provably true.

To prove Proposition 15 we will first need a perturbation result similar to Lemma 10.

Lemma 16. *Let $f : \mathcal{X} \to \mathbb{R}$ and $g : \mathcal{X} \to \mathbb{R}$ such that f is τ strongly-convex and $\sup_{\boldsymbol{x} \in \mathcal{X}} |g'(\boldsymbol{x})| \leq \varepsilon$. Let $\boldsymbol{x}^\star = \arg\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x})$ and $\tilde{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}) + g(\boldsymbol{x})$. Then*

$$\|\boldsymbol{x}^\star - \tilde{\boldsymbol{x}}\| \leq \frac{\varepsilon}{\tau}.$$

Proof. By first order optimality we have $f'(\tilde{\boldsymbol{x}}) + g'(\tilde{\boldsymbol{x}})$, hence $|f'(\tilde{\boldsymbol{x}})| = |g'(\tilde{\boldsymbol{x}})| \leq \varepsilon$. By strong convexity $\tau \|\tilde{\boldsymbol{x}} - \boldsymbol{x}^\star\| \leq |f'(\tilde{\boldsymbol{x}})| \leq \varepsilon$ which gives the result. □

Proof of Proposition 15. For a given n and κ such that $\mu(\phi_n^\kappa) \wedge \mu(\phi_\infty^\kappa) > 0$,

$$\begin{aligned}
b_n &= \|\boldsymbol{\nu}^\star(n) - \boldsymbol{\nu}^\star(\infty)\| \\
&= \|\boldsymbol{\nu}^\star(n) - \boldsymbol{\nu}_\kappa^\star(n) + \boldsymbol{\nu}_\kappa^\star(n) - \boldsymbol{\nu}_\kappa^\star(\infty) + \boldsymbol{\nu}_\kappa^\star(\infty) - \boldsymbol{\nu}^\star(\infty)\| \\
&\leq \|\boldsymbol{\nu}^\star(n) - \boldsymbol{\nu}_\kappa^\star(n)\| + \|\boldsymbol{\nu}_\kappa^\star(n) - \boldsymbol{\nu}_\kappa^\star(\infty)\| + \|\boldsymbol{\nu}_\kappa^\star(\infty) - \boldsymbol{\nu}^\star(\infty)\| \\
&\leq 2\sqrt{\varepsilon_{\mathrm{inv}}(n, k)} + \|\boldsymbol{\nu}^\star(n) - \boldsymbol{\nu}_\kappa^\star(n)\| + \|\boldsymbol{\nu}_\kappa^\star(\infty) - \boldsymbol{\nu}^\star(\infty)\| \\
&\leq 2\sqrt{\varepsilon_{\mathrm{inv}}(n, \kappa)} + \varepsilon_{\mathrm{flat}}(n, \kappa),
\end{aligned}$$

where the first inequality is the triangle inequality, the second comes from Lemma 10, and the last inequality comes from Lemma 16. Taking the minimum of the right hand side over valid κ yields the claim $t_n \geq b_n$. □

[3] Relative to the bound on b_n implied solely by a_n (see Proposition 1).

C TRUNCATION INDEX SELECTION

Empirically finding the minimizer $\kappa^\star(n)$ in the definition of t_n (Def. 6) is not tractable due to complicated nature of the decomposition objective $\mathcal{J}$. Instead of trying to minimize $\mathcal{J}$, we will use a simpler surrogate process which we outline below.

Note that given a finite grid of HPs $\{\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_g\}$, we only need to produce a truncation index for each $\boldsymbol{\nu}_i$ with $i \in [g]$. Let $\boldsymbol{\kappa} = (\kappa_1, \ldots, \kappa_g)$ represent the vector of these values where $\kappa_i = \kappa(\boldsymbol{\nu}_i)$ for $i \in [g]$. We define $\phi_n^{\boldsymbol{\kappa}}$ to be the set of pointwise evaluations $\{(\boldsymbol{\nu}_i, \phi_n^{\kappa_i}(\boldsymbol{\nu}_i))\}_{i\in[g]}$ and identify it with the function obtained from its linear interpolation. Our goal is to return a set of truncation index vectors $\hat{\boldsymbol{\kappa}}(n)$.

Let $n_{\max}$ denote the largest width model under consideration and fix a width $n < n_{\max}$. Consider the following proxy objective with parameters $\boldsymbol{\tau} = (\tau_1, \tau_2)$ and $\tau_1, \tau_2 > 0$,

$$\mathcal{J}_{\text{proxy}}(\boldsymbol{\kappa}; \boldsymbol{\tau}) := \frac{1}{g} \sum_{i\in[g]} |\phi_n^{\kappa_i}(\boldsymbol{\nu}_i) - \phi_{n_{\max}}^{\kappa_i}(\boldsymbol{\nu}_i)| + \tau_1 \cdot \text{Lip}(\phi_n^{-\boldsymbol{\kappa}}) + \tau_2 \cdot \text{Lip}(\phi_{n_{\max}}^{-\boldsymbol{\kappa}}), \tag{15}$$

and define its minimizer to be $\boldsymbol{\kappa}^\star(\boldsymbol{\tau}) := \arg\min_{\boldsymbol{\kappa}} \mathcal{J}_{\text{proxy}}(\boldsymbol{\kappa}; \boldsymbol{\tau})$, which can be found approximately using coordinate descent (see Algorithm 2). The objective $\mathcal{J}_{\text{proxy}}$ is similar to the objective $\mathcal{J}$ in Definition 6, except that instead of a supnorm we use an average ℓ_1-norm to promote tractability, we use $n_{\max}$ as an infinite-width proxy, and we absorb all the curvature based scalings $\mu(\cdot)$ into constants τ_1, τ_2. We will set $\hat{\boldsymbol{\kappa}}(n) = \boldsymbol{\kappa}^\star(\hat{\boldsymbol{\tau}})$ for a "reasonable" choice of $\hat{\boldsymbol{\tau}}$. In particular, $\hat{\boldsymbol{\tau}}$ will be chosen to be the smallest[4] $\boldsymbol{\tau}$ so that $\phi_n^{\boldsymbol{\kappa}^\star(\boldsymbol{\tau})}$ and $\phi_{n_{\max}}^{\boldsymbol{\kappa}^\star(\boldsymbol{\tau})}$ are approximately convex and the minimizers are close to the minimizers of ϕ_n and $\phi_{n_{\max}}$ respectively, assuming such $\hat{\boldsymbol{\tau}}$ exists. The full details of the process are given in Algorithm 1 in Appendix C.

In this section we describe our procedure (Algorithm 1) for selecting $\hat{\boldsymbol{\kappa}}(n)$ (see Section 4.1). We do not claim this procedure is optimal in any sense and emphasize that we are just searching for a valid $\hat{\boldsymbol{\kappa}}(n)$ so that a certain sufficient condition holds qualitatively in order to support our conjecture for fast hyperparameter transfer. For convenience, we only consider the case where we sweep a single HP, although this can be straightforwardly extended.

As part of our algorithm, we need to measure the convexity and flatness of a function f given a set of pointwise evaluations $\{(\nu_i, y_i)\}_{i=1}^g$ where $\nu_1 < \cdots < \nu_g$ and $y_i = f(\nu_i)$. For each interior index $i = 2, \ldots, g-1$, define the three-point second-derivative estimate

$$\hat{f}''(\nu_i) := 2\left(\frac{y_{i-1}}{(\nu_{i-1} - \nu_i)(\nu_{i-1} - \nu_{i+1})} + \frac{y_i}{(\nu_i - \nu_{i-1})(\nu_i - \nu_{i+1})} + \frac{y_{i+1}}{(\nu_{i+1} - \nu_{i-1})(\nu_{i+1} - \nu_i)} \right).$$

The *convexity error* is the fraction of interior points with negative curvature estimate:

$$\text{ConvErr}(\{(\nu_i, y_i)\}_{i=1}^g) := \frac{1}{g-2} \sum_{i=2}^{g-1} \mathbf{1}\left\{ \hat{f}''(\nu_i) < 0 \right\}. \tag{16}$$

The *Lipschitz constant* is the maximum slope magnitude:

$$\text{Lip}(\{(\nu_i, y_i)\}_{i=1}^g) := \max_{1\le i\le g-1} \left| \frac{y_{i+1} - y_i}{\nu_{i+1} - \nu_i} \right|. \tag{17}$$

[4]For definiteness, we order (τ_1, τ_2) by their sum, breaking ties in the first-coordinate.

Algorithm 1: Compute $\hat{\boldsymbol{\kappa}}(n)$

Input:

$\mathcal{N}$	set of widths, with $n_{\max} = \max \mathcal{N}$
$\{\nu_i\}_{i=1}^{g}$	grid of g hyperparameter points
$\mathcal{T}$	candidate list of τ values
ε_{cvx}, $\varepsilon_{\text{amin}}$	tolerances (convexity, argmin proximity)
$\Phi_n \in \mathbb{R}^{g \times n}$	arrays $\phi_n^k(\nu_i)$ for each $n \in \mathcal{N}$

Output: $\hat{\boldsymbol{\kappa}}(n)$ truncation vectors in $[n]^g$ for $n < n_{\max}$, or FAIL

1. **Grid:** $\mathcal{G}_{\boldsymbol{\tau}} := \{(\tau_1, \tau_2) : \tau_1, \tau_2 \in \mathcal{T}\}$ sorted by $\tau_1 + \tau_2$
2. **For each** $n \in \mathcal{N} \setminus \{n_{\max}\}$**:**
   (a) **For** $(\tau_1, \tau_2) \in \mathcal{G}_{\boldsymbol{\tau}}$ (in order):
      i. $\boldsymbol{\kappa} \leftarrow \text{MINIMIZEPROXY}(\Phi_n, \Phi_{n_{\max}}; \tau_1, \tau_2)$ `// Alg. 2; minimizes Eq. (15)`
      ii. $E_{\text{cvx}} := \max\{\text{ConvErr}(\phi_n^{\boldsymbol{\kappa}}),\ \text{ConvErr}(\phi_{n_{\max}}^{\boldsymbol{\kappa}})\}$ `// Eq. (16)`
      iii. $a_n := \arg\min_\nu \phi_n^{\kappa}, \quad a_n^{\text{tot}} := \arg\min_\nu \phi_n;$
      $a_\infty := \arg\min_\nu \phi_{n_{\max}}^{\kappa}, \quad a_\infty^{\text{tot}} := \arg\min_\nu \phi_{n_{\max}}.$
      $\Delta := \max\{|a_n - a_n^{\text{tot}}|,\ |a_\infty - a_\infty^{\text{tot}}|\}$ `// argmin proximity`
      iv. **If** $E_{\text{cvx}} \le \varepsilon_{\text{cvx}}$ **and** $\Delta \le \varepsilon_{\text{amin}}$, **set** $\hat{\boldsymbol{\kappa}}(n) \leftarrow \boldsymbol{\kappa}$ and **break** to the next n.
   (b) **If** no pair in $\mathcal{G}_{\boldsymbol{\tau}}$ is accepted, **set** $\hat{\boldsymbol{\kappa}}(n) \leftarrow$ FAIL.

Algorithm 2: Minimize Proxy Objective Eq. (15)

Input:

Φ_n, $\Phi_{n_{\max}}$	arrays $\phi_n^k(\nu_i)$ and $\phi_{n_{\max}}^k(\nu_i)$, shapes $g \times n$ and $g \times n_{\max}$
(τ_1, τ_2)	positive penalty weights

Output: $\boldsymbol{\kappa}^\star \in [n]^g$ (approximate minimizer via coordinate descent)

1. **Initialize:** $\boldsymbol{\kappa} \leftarrow (n/2, \ldots, n/2)$
2. **Repeat** until no coordinate changes:
   For $i = 1, \ldots, g$:
      i. *Local scores for each* $k \in [n]$*:*
      $$\text{score}(k) = \mathcal{J}_{\text{proxy}}(\tilde{\boldsymbol{\kappa}}) \text{ where } \tilde{\boldsymbol{\kappa}} \text{ is } \boldsymbol{\kappa} \text{ with } \kappa_i \text{ switched to } k$$
      ii. *Coordinate update:* $\kappa_i \leftarrow \arg\min_{k \in [n]} \text{score}(k)$.
3. **Return** $\boldsymbol{\kappa}^\star := (\kappa_i)_{i=1}^{g}$.

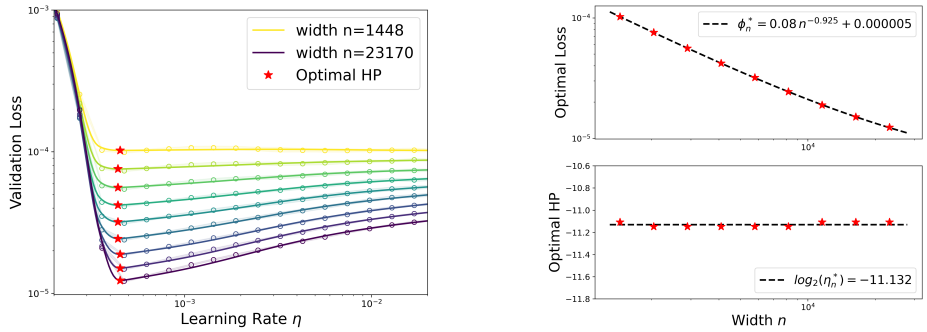
D EXPERIMENTAL DETAILS AND ADDITIONAL EXPERIMENTS

D.1 SYNTHETIC EXPERIMENTS

We provide an additional synthetic setting where fast transfer is observed. We consider a regression setting with shallow ReLU neural network: $f(\boldsymbol{x}) = \sum_{i=1}^{n} a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i)$, where we aim to tune the learning rate η that minimizes the validation loss. We set the target to be the norm function

$$y = \|\boldsymbol{x}\|_2^2, \quad \text{where } \boldsymbol{x} \sim \mathcal{N}(0, \boldsymbol{I}_d).$$

Note that this function is easier to represent (with a shallow neural network) than the classification example in Section 3.2 due to the absence of indicator function. We set $d = 2^8, n = 2^{14}$, and run the Adam optimizer (Kingma & Ba, 2014) for $T = 2^{14}$ steps with batch size 2^8 to minimize the squared loss. The initialization and learning rate are set according to μP (Yang & Hu, 2021). In Figure 9 we observe that while the validation MSE loss decays at an approximate $1/n$ rate, the optimal learning rate remains almost invariant across (reasonably large) width.



(a) Validation loss vs. learning rate.

(b) Scaling of optimal loss and learning rate.

Figure 9: Optimal learning rate (validation loss) for two-layer ReLU network to learn the norm function.

D.2 LLM EXPERIMENTS

For experiments on the Llama architecture we use the following schedule for the EMA.

EMA Warmup We warm up the EMA decay coefficient α_t linearly from $\alpha_{\text{start}} = 0.98$ to $\alpha_{\text{end}} = 0.9995$ over 2000 steps in the effective window $(1 - \alpha_t)^{-1}$. To capture early-training variation without increasing linearization error, we subsample the EMA trajectory every τ steps, with τ itself warmed up linearly from $\tau_{\text{start}} = 2$ to $\tau_{\text{end}} = 10$ over the same period.

D.3 LLAMA ADAM LR

Llama Adam Configuration Below is the default setup for all Llama experiments using Adam.

Dataset	WikiText-103
Epochs	1
Hidden layers	4
Optimizer	Adam ($\beta_1 = 0.9,\ \beta_2 = 0.999$)
Batch Size	128
LR Schedule	WSD with 4% linear warmup & 20% cooldown

LR Grid Search

- LR $\in \{1.0, 1.4, 2.0, 2.8, 4.0, 4.8, 5.7, 6.7, 8.0\} \times 10^{-3}$
- $n \in \{128, 256, 512, 1024, 2048\}$

All curvature computations are done in $\log_2(\text{LR})$, since we sweep the peak learning rate on a log-grid. The loss $\mathcal{L}$ is evaluated on the validation split. We average runs over 3 random seeds.

D.4 LLAMA ADAM β_1 AND β_2

We repeat the same experiment from Appendix D.3 for the Adam momentum hyperparameters, fixing LR $= 0.004$. For the β_1 sweep we vary $\beta_1 \in \{0.63, 0.78, 0.86, 0.92, 0.95\}$ and for the β_2 sweep we vary $\beta_2 \in \{0.95, 0.98, 0.99, 0.995, 0.998, 0.9999\}$. The sweeps are performed in logspace in the effective window size $(1-\beta)^{-1}$. The analog of Figure 7 is presented at step 7185 for β_1 in Figure 10 and for β_2 in Figure 11. We note that the performance is insensitive to β_1 except when it is too large and insensitive to β_2 except when it is too small. The computed values of $\hat{\boldsymbol{\kappa}}(n)$ shown in Figure 12 are fairly similar for all the hyperparameters, suggesting that the dimension of this invariant subspace may be mostly data and architecture dependent. It will be interesting in the future to perform similar experiments for other HPs such as weight decay and if our decomposition viewpoint can shed insight onto HPs which do not show fast transfer.

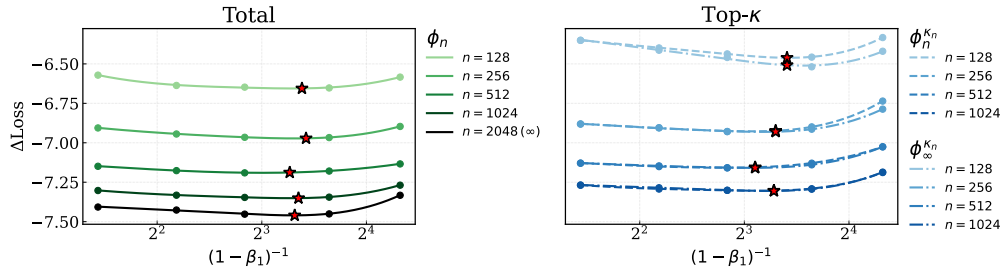


Figure 10: Same as Fig. 7, but for Adam β_1 at step 7185.

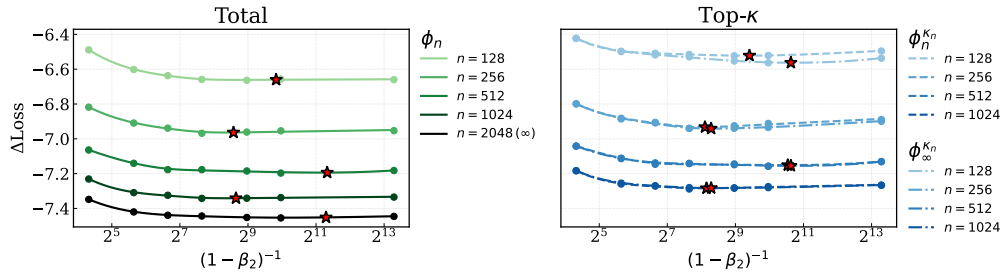


Figure 11: Same as Fig. 7, but for Adam β_2 at step 7185.

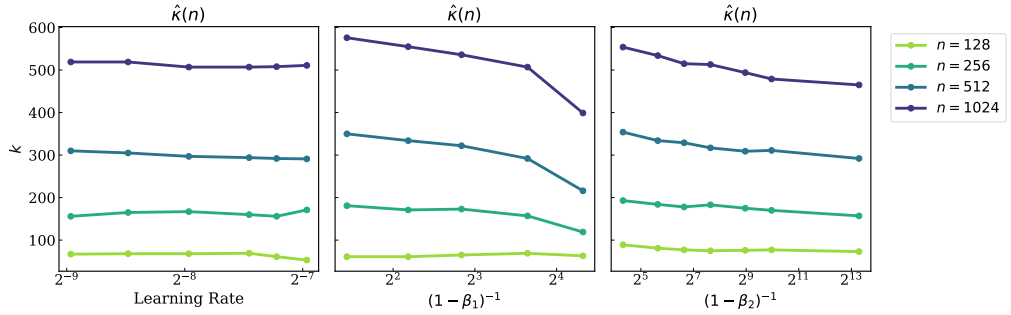


Figure 12: The computed values of $\hat{\boldsymbol{\kappa}}(n)$ for Adam LR, β_1, and β_2.

D.5 LLAMA MUON

As in Section 4.2, we train a Llama-style transformer architecture with a warmup stable (WSD) learning rate schedule on WikiText-103, but using the Muon (Jordan et al., 2024) optimizer instead of Adam. The training configuration is shown below. The learning rate sweeps are shown in Figure 13a.

Llama Muon Configuration Below is the setup used for the Muon training.

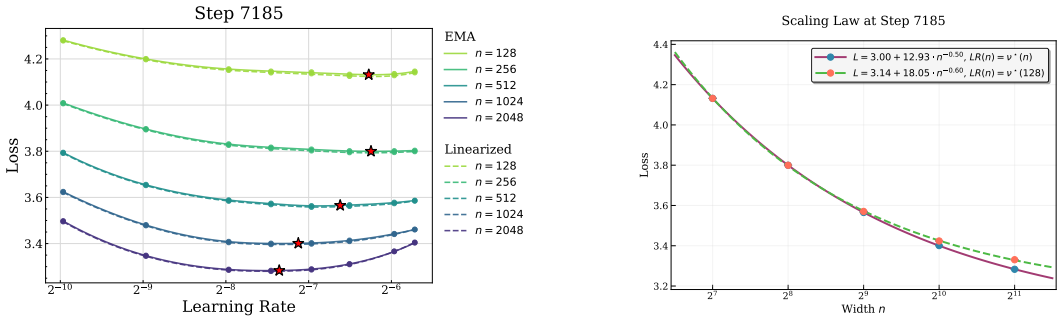
Dataset	WikiText-103
Epochs	1
Hidden layers	4
Optimizer	Muon ($\beta = 0.95$, Adam LR$= 0.004$, $\beta_1 = 0.9$, $\beta_2 = 0.999$)
Batch Size	128
LR Schedule	WSD with 4% linear warmup & 20% cooldown

LR Grid Search

- LR $\in \{0.1, 0.2, 0.4, 0.57, 0.8, 1.1, 1.6, 1.9\} \times 10^{-2}$
- $n \in \{128, 256, 512, 1024, 2048\}$

Although the Muon algorithm achieves a better loss, the reducible loss scaling rate is similar to Adam (see Fig. 13b compared with Fig. 4b) but the optimal learning rate convergence does not seem as rapid (see Fig. 13a compared with Fig. 4a). Our hypothesis is that the conditions for our fast transfer conjecture hold for the Adam optimizers due to the low-rank nature of the updates, but this property is broken by the whitening step in Muon which harms fast transfer. We further probe these aspects using the lens of our decomposition which provides interesting insights into the dynamics of Muon in relation to the network width.

In Figure 14 we see that the k for which top-k invariance holds in Muon is much smaller and for large n accounts for a small fraction of the total loss. As such it is likely that our fast transfer hypothesis is incompatible with the Muon optimizer and we are unable to find supporting evidence. We conjecture that this is related to our observation of the less perfect transfer observed in Figures 13a and 13b, suggesting that our condition for fast transfer is somewhat necessary.



(a) EMA loss (solid) and linearized loss (dashed).

(b) Scaling law for the EMA loss.

Figure 13: Training a 4-layer Llama transformer on WikiText-103 with Muon. **Left:** EMA and linearized losses nearly coincide. **Right:** We plot the EMA loss as a function of width n using (1): the optimal learning rates $\nu^\star(n)$ (2): the optimal width-128 learning rate $\nu^\star(128)$. At large n, $\nu^\star(128)$ is slightly suboptimal.

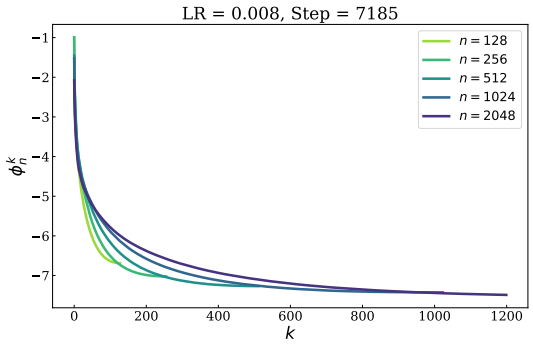


Figure 14: The top-k loss ϕ_n^k. The ϕ_n^k descend slowly with k especially for large n showing that top-k invariance holds for a small range of k and the residual is significant.

D.6 CIFAR-10 MLP SGD

We also probe the generality of our observations to a different dataset, optimizer, and architecture: CIFAR-10 training using SGD on a 2-layer MLP with ReLU activation and no biases.

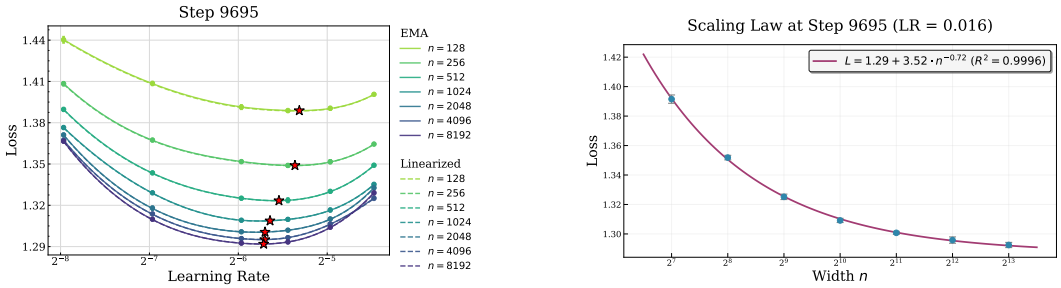
CIFAR-10 Training Configuration Below is the setup used for our CIFAR-10 experiment.

Dataset	CIFAR-10
Epochs	100
Layers	2
Optimizer	Momentum SGD ($\beta = 0.9$)
Batch Size	512
LR Schedule	WSD with 4% linear warmup & 20% cooldown
Data Augmentation	Mixup, Random Resized Cropping

We use the same EMA warmup and subsampling schedule as detailed in Appendix D. We use our largest width $n_{\max} = 8192$ as the infinite-width proxy for computing $\hat{\boldsymbol{\kappa}}(n)$ using Algorithm 1.

LR Grid Search

- LR $\in \{1.0, 1.4, 2.0, 2.8, 4.0, 4.8, 5.7, 6.7, 8.0\} \times 10^{-3}$
- $n \in \{128, 256, 512, 1024, 2048, 4096, 8192\}$



(a) The final EMA loss (solid) and final linearized.

(b) Scaling law for the EMA loss at step 9695.

Figure 15: Training a 2-layer MLP on CIFAR-10 using SGD. We see that this problem exhibits fast transfer.

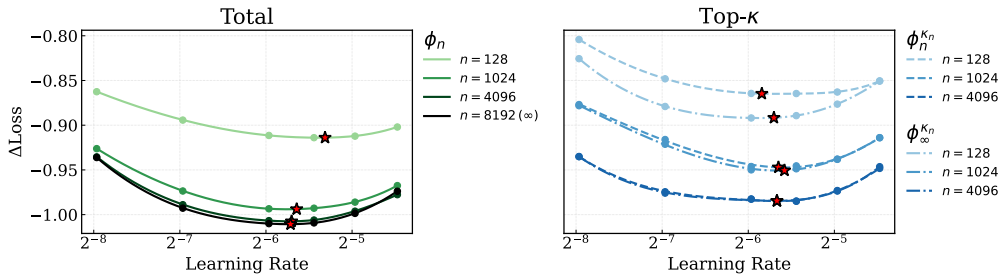


Figure 16: Same as Fig. 7, but for the setup described above.

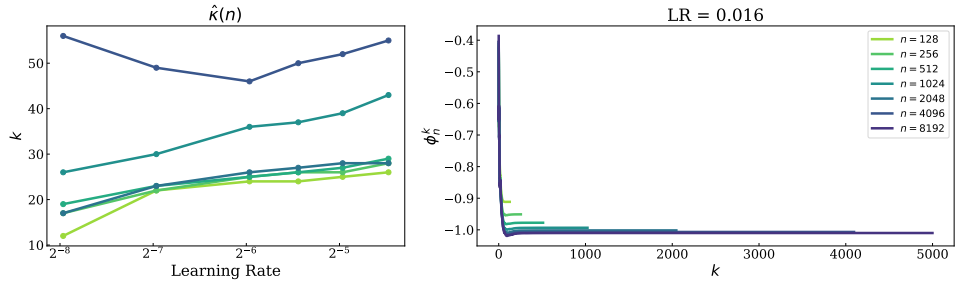


Figure 17: Same as Fig. 12 but for the above setup. This setting appears much more low-rank than the one in Section 4.2, based on how much smaller k_n/n is (left, see Fig. 12) and more quickly ϕ_n^k flattens out with k (right, see Fig. 6).