A Note on Lazy Training in Differentiable Programming

Bissmella Bahaduri*
bissmellabahaduri@gmail.com
ENS Paris-Saclay

Richard Goudelin*
rgoudelin@gmail.com
ENS Paris-Saclay

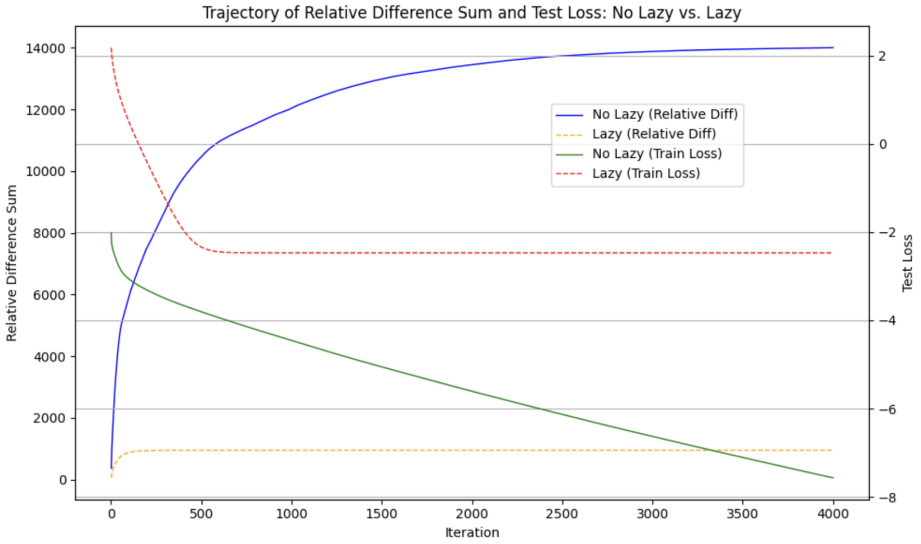

Figure 1: Trajectory of relative parameter changes and $\log_{10}$ test loss: Comparison between lazy training and non-lazy training with non-symmetrized initialization.

Abstract

Large neural network models have demonstrated remarkable performance across various vision and language tasks. Despite their success, understanding and interpreting these models remain challenging due to their inherent complexity. This paper further investigates the dynamics of lazy training regimes, building upon the theoretical frameworks of lazy training. We explore how scaling factors influence the transition to lazy training, where models behave linearly around their initialization. Through theoretical analysis and extensive experiments on two-layer neural networks, we extend the demonstration of how large scaling factors induce minimal parameter updates, leading to faster convergence but potentially limiting the model's ability to capture complex nonlinear relationships. Additionally, we examine the correlation between weight initialization scales and output scaling factors and visualize the loss landscape under different scaling conditions. Our visualizations and empirical and theoretical analyses provide deeper insights into the training dynamics of neural networks and build upon the understanding of lazy training dynamics.

Keywords

Optimization, Machine learning theory, lazy training, learning theory

1 Introduction

Large models are becoming more frequent and have shown promising results in both vision and language tasks. However, the interpretability of these large models has still remained a dilemma. On the other hand, since these foundation models are complex in nature, studying them is practically and theoretically challenging. The dynamics of training wide networks have been studied by analyzing the behavior of shallow networks. Previously, [3] has shown that infinitely wide networks behave in a linear regime around their initialization point. However, [2] argues that this phenomenon is not due to the over-parametrization of the network, but due to a scaling factor. In fact, [2] shows that any parametric model enters a lazy training regime by introducing an explicit scale factor on the output of the network, showing that fast convergence is possible but at the cost of removing nonlinearity, which is crucial for learning a good function. In this note, we extend the work of [2] by advancing the theoretical framework and conducting experiments on two-layer neural networks.

1.1 general context

Let a parameterized smooth function $h : \mathbf{R}^p \to \mathcal{F}$ and a smooth loss function $R : \mathcal{F} \to \mathbf{R}_+$ and an objective function $\mathbf{F} : \mathbf{R}^p \to \mathbf{R}_+$. The goal is to minimize the mentioned object function $\mathbf{F}$ which can be formulated as:

$$\mathbf{F}(w) := R(h(w)).$$

*: Equal contribution

Starting from an initialization w_0, a linearized version of the model is formulated as $\hbar(w) = h(w_0) + Dh(w_0)(w - w_0)$ where D denotes the Jacobian, and its corresponding object function $\bar{F}(w) := \mathbf{R}(\hbar(w))$.

At the initialization both h and $\hbar$ are the same and thus the F and $\bar{F}$. The main objective is to study scenarios where both remain close beyond the initialization path. If both stay close until the end, the model is said to be in a lazy training regime.

Assuming w_0 is a non-critical and non-optimal point then an update following a gradient step is $w_1 = w_0 - \eta \nabla F(w_0)$ where η is a learning rate. As a result the relative change of objective function F and relative change of differential of h, Dh is:

$$\Delta(F) = \frac{|F(w_1) - F(w_0)|}{F(w_0)} \approx \eta \frac{||\nabla F(w_0)||^2}{F(w_0)}$$

$$\Delta(Dh) = \frac{||Dh(w_1) - Dh(w_0)||}{||Dh(w_0)||} \leq \eta \frac{||\nabla F(w_0)||.||D^2 h(w_0)||}{||Dh(w_0)||}$$

Then, the model is said to be in a lazy training regime if the model benefits from a huge reduction in objective function while the differential of h stays mostly unchanged, implying $\Delta(F) \gg \Delta(Dh)$ with the above approximation of $\Delta(F)$ and $\Delta(Dh)$.

In the case of square loss, this can be implied with a simpler criterion

$$kh(w_0) := ||h(w_0) - y|| \frac{||D^2 h(w_0)||}{||Dh(w_0)||^2} \ll 1$$

with the approximation $||\Delta F(w_0)|| \approx ||Dh(w_0)||.||h(w_0) - y||$ with some y in $\mathcal{F}$ as ground truth.

Rescaled model: The main idea of [2] is that lazy training happens when the output of the model is rescaled. In the case of a rescaled model scaled by a factor α the criterion in case of square loss becomes:

$$kh(w_0) := \frac{1}{\alpha} ||\alpha h(w_0) - y|| \frac{||D^2 h(w_0)||}{||Dh(w_0)||^2}$$

As α grows bigger the criterion decreases, leading to a lazy training situation.

Homogeneous model: If the model is q-positively homogeneous then initializing the model by parameters scaled by λ is equivalent to a scale factor of λ^q and the criterion in case of square loss becomes:

$$kh(w_0) := \frac{1}{\lambda^q} ||\lambda^q h(w_0) - y|| \frac{||D^2 h(w_0)||}{||Dh(w_0)||^2}$$

Two-layer neural networks: A two-layer neural network can be represented as:

$$h_m(w) = \alpha(m) \sum_{i=1}^{m} \phi(\theta_i)$$

where $m, d \in \mathbf{N}$, $\alpha(m) > 0$ is a normalization, $w = (\theta_1, ..., \theta_m)$ the parameters, and $\phi : \mathbf{R}^d \to \mathcal{F}$ is a smooth function. Assuming that $(\theta_i)_{i=1}^m$ initialized such that $\mathbf{E}\phi(\theta_i) = 0$ and $D\phi$ is not identically 0 on the support of initialization, then for large m

$$\mathbf{E}[k_{h_m}(w_0)] \leq m^{-\frac{1}{2}} + (m\alpha(m))^{-1}$$

Thus, as $m \to \infty$ and $m\alpha(m) \to \infty$, the model enters the lazy regime when $\alpha(m)$ is bigger than $1/m$. In the experiments section, this is shown empirically and explored visually.

Our Contributions: We summarize our core contributions as follows:

- We extend the theoretical analysis by investigating the influence of weight scaling factors on lazy training dynamics.
- We provide visualizations demonstrating how weights evolve in both lazy and non-lazy training regimes.
- We examine the correlation between weight initialization scale factors and output scaling factors.
- We visualize the loss landscape of two-layer neural networks under varying parameter counts and corresponding output scaling factors.
- We discuss the limitations of our study, including the choices in hyperparameterization and the applicability of our findings to broader contexts.

2 Related Works

Close to the current work, the neural tangent kernel [3] provides a theoretical framework for understanding the behavior of neural networks. It has been observed that at initialization, neural networks behave similarly to kernel methods and that the network's output is largely determined by its random weights at initialization. More precisely, it is argued that infinitely wide neural networks converge close to their initialization point and that learning happens through small and linear adjustments near the initialization. As the width of the network grows to infinity, the training dynamics become more linear and deterministic, which fundamentally provides a theoretical framework for understanding how neural networks learn.

More recently, [8] shows how the scaling of the initialization weights affects the hidden layer's spectral bias. Specifically, it shows that a small initialization scaling of the neural network's hidden layer is biased toward having a low-rank structure when trained with gradient flow.

3 Analysis of Lazy Training Dynamics

The goal of this section is to show that a scaled objective function behaves similarly to the linearized model when the objective function is scaled with a large scaling factor.

The main assumption is that the model is differentiable with a locally Lipschitz differential and that the objective function is differentiable with a Lipschitz gradient. This assumption complies with supervised learning, where the objective is to minimize the expected risk over a pair distribution of input and output, and the output/prediction space consists of square-integrable functions. The risk is also a smooth function, and the function mapping from input to output can be defined as a parametric model where the output depends on the input data and the function's parameters, such as neural networks.

Since most neural networks are based on supervised learning or at least semi-supervised learning, this assumption is applicable in most cases for neural networks.

The following analysis is based on gradient flow, which is an approximation of stochastic gradient descent with a small learning rate.

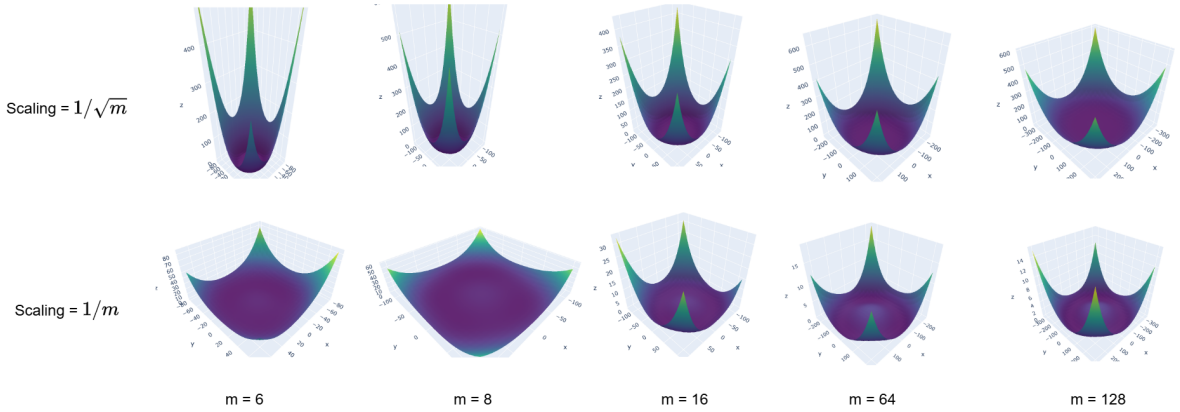

Figure 2: Loss landscape of 2-layer network for different hidden layer size(m).

Gradient flow: Initialized by $w_0 \in \mathbf{R}^p$, the gradient flow of the objective function F_α is the trajectory $(w_\alpha(t))_{t\geq 0}$ in space $\mathbf{R}^p$ satisfying $w_\alpha(0) = w_0$ and solves the ordinary differential equation.

$$w'_\alpha(t) = -\nabla F_\alpha(w_\alpha(t)) = -\frac{1}{\alpha} Dh(w_\alpha(t))^T \nabla R(\alpha h(w_\alpha(t)))$$

Similarly, the gradient flow of the linearized version of the objective function $\bar{F}_\alpha$, $(\bar{w}_\alpha(t))_{t\geq 0}$, satisfies $\bar{w}_\alpha(0) = w_0$ and solves:

$$\bar{w}'_\alpha(t) = -\nabla \bar{F}_\alpha(\bar{w}_\alpha(t)) = -\frac{1}{\alpha} Dh(w_\alpha(t))^T \nabla R(\alpha \bar{h}(\bar{w}_\alpha(t)))$$

Setting $h(w_0) = 0$ simplifies the situation and makes it independent of α.

While we do not go into the details of the theorems provided in [2], in this section, we provide a summary of these theorems.

3.1 Generalized setting

In a generalized setting, disregarding the convexity of the risk function R and assuming $h(w_0) = 0$, under a large scaling factor α leads to a lazy training regime.

Under the mentioned assumption and a fixed time horizon $T > 0$, the difference between the gradient flow of the model and the linearized model is at the order of the inverse square of scaling factor

$$||w_\alpha(T) - \bar{w}_\alpha(t)|| = O(\frac{1}{\alpha^2})$$

showing that as the α increases the gradient flow of the model gets closer to its linearized gradient flow.

In a supervised setting, the above bound means that the model generalizes as the linearized model beyond the training set. The generalization behavior of the linear models has been well studied [6].

More specifically, in case of square loss, and assuming that the model h is Lip(h)-Lipschitz, and Dh is Lip(Dh) $-$ *Lipschitz* on the ball with radius $r > 0$ centered on initialization point w_0. Than in $K > 0$ iteration corresponding to $TLip(h)^2$, the following holds

$$\frac{||(w_\alpha(T)) - \alpha \bar{h}(\bar{w}_\alpha(T))||}{||(w_0) - y||} \leq \frac{K^2}{\alpha} \frac{Lip(Dh)}{Lip(h)^2} ||\alpha h(w_0) - y||$$

under the condition that $\alpha \geq K||\alpha h(w_0) - y||/(rLip(h))$.

Here, it is observable that the relative gap between the gradient flow of the model and its linearized correspondent grows as K^2.

3.2 Convergence under lazy training regime

In this section, we study the behavior of the model under the assumption of strict convexity of the risk function R. In such a case the linearized objective function $\bar{F}_\alpha$ is strictly convex in the affine hyperspace around the initialization point w_0 and the gradient is linear resulting in a linear convergence to the global minimizer of $\bar{F}_\alpha$. Again, in a simplified case where the $h(w_0) = 0$ the global minimizer does not depend on α and the gap between the model and its linearized version is at most at the order of $1/\alpha$. Under the above assumptions lazy training is further studied in two cases of over-parametrized and under-parameterized settings

Over-parameterized setting: In an overparameterized case with a large rank of $Dh(w_0)$ the model benefits from a bigger degree of freedom which in turn guarantees that the model can converge around w_0. More specifically, given a well-conditioned risk function R (smooth and strongly convex) and under the assumption that $Dh(w_0)$ has full rank, and the model is initialized in a way that the initialization norm is not big and does not de-stabilize the model, then the model will converge with an exponential rate with respect to time t.

Under-parameterized setting: In this case, the assumption of over-parameterization is relaxed. Then, under the assumption of separability of output space $\mathcal{F}$, strong convexity of R, and constant rank of $Dh(w)$ around w_0 the model converges geometrically to a local minima given a large enough scale factor α. However, the model in this setting does not reach the global minimizer as it is out of reach of the variations of w in this case. This result is reproduced in the experiments section.

Convergence in a local-minima for under-parameterized setting might look in contradiction with [1] which states that the gradient flow of a neural network under correct initialization even if non-convex, converges to a global maximum. However, this is

not a contradiction since here we assume that the network is parameterized with a finite number of parameters while [1] shows convergence to the global maximum under the limit of an infinite number of parameters.

3.3 Impact of Initialization Variance τ on the $\kappa_h(w_0)$ Criterion

We examine how the initialization variance parameter τ, used to initialize weights $w_0 \sim N(0, \tau^2 I)$, influences the criterion

$$\kappa_h(w_0) = \frac{\|h(w_0) - y\| \cdot \|D^2 h(w_0)\|}{\|Dh(w_0)\|^2},$$

Assumptions: We assume that h is positively homogeneous of degree q, i.e., $h(\lambda w) = \lambda^q h(w)$ for all $\lambda > 0$, and that weights are initialized as $w_0 \sim N(0, \tau^2 I)$.

Under these assumptions, the scaling properties yield

$$\kappa_h(w_0) = \frac{\|h(w') - y/\tau^q\| \cdot \|D^2 h(w')\|}{\|Dh(w')\|^2},$$

where $w' = w_0/\tau$.
In a symmetrized initialization, it is ensured that the output of the network at initialization is 0 while for non-symmetrized it is not the case. These results imply two interpretations:

- In the context of a non-symmetrized initialization, where the output satisfies $h(w') \gg 0$, we observe the following behavior:
  - For large values of τ, the term $\frac{y}{\tau^q}$ becomes negligible compared to $h(w')$. As a result, the condition number $\kappa_h(w_0)$ simplifies to:
  $$\kappa_h(w_0) \approx \frac{\|h(w')\| \cdot \|D^2 h(w')\|}{\|Dh(w')\|^2}.$$
  - This approximation indicates that for sufficiently large τ, the criterion can be simplified. Consequently, as τ increases further, the criterion stabilizes, leading to a plateau where the model enters a regime of lazy training.
  - This behavior was verified empirically in our experiments in section 3.3.
- In the context of a symmetrized initialization, where the output satisfies $h(w') \approx 0$, we observe the following behavior:
  - Since $h(w')$ is close to zero, the lazy training criterion is expected to be satisfied at lower values of τ. Consequently, the plateau phenomenon should occur earlier as τ increases and it has been verified empirically in section 3.3.

The detailed derivation of this relationship is provided in Appendix A.

3.4 Correlations between weight scaling and output scaling

Starting from the rescaled κ criterion:

$$\kappa_{\alpha h}(w_0) = \frac{1}{\alpha}\|\alpha h(w_0) - y\| \frac{\|D^2 h(w_0)\|}{\|Dh(w_0)\|^2},$$

we introduce the initialization variance parameter τ, where the initial weights w_0 are drawn from a normal distribution $N(0, \tau^2 I)$.

$$\kappa_{\alpha h}(w) = \frac{\|h(w') - y\frac{1}{\alpha\tau^q}\| \cdot \|D^2 h(w')\|}{\|Dh(w')\|^2}.$$

So to keep the criterion at a given level it must hold: $\alpha\tau^q = constant$ or $\alpha \propto \frac{1}{\tau^q}$

4 Empirical analysis

In this part we conduct extensive experiments and do not limit ourselves just to the experiments reported in[2]. [2] reports experiments on the following two configurations:

- A two-layer network student-teacher setup on synthetic data.
- A CNN model on the CIFAR-10 dataset [4].

The code provided by [2] was originally implemented in Julia. For convenience, it was converted to Python. We extended the experiments on a basic two-layer neural network to enhance interpretability and to observe the phenomena proven in the theoretical section on 2-layer neural networks. Furthermore, we aimed to test the limits of the experiments on this simple setup to ensure greater clarity and interpretability. Our experiments can be found at: https://github.com/RichardGou/MVA_GDA_PROJECT. .

4.1 two layer neural network

In this setting, we use a student network that aims to imitate the output of a teacher network. The student and teacher networks are separate from each other and do not share any parameters. More specifically, the student network $s_m(w) = f_m(w, x)$ is parameterized by w with $f_m(w, x) = \sum_{i=1}^{m} a_i \,\mathrm{RELU}(b_i \cdot x)$, where $a_i \in \mathbf{R}$ and $b_i \in \mathbf{R}^d$, and RELU denotes the ReLU activation function. The teacher network T has a similar architecture with $m_0 = 3$ hidden neurons and normalized weights $\|a_i b_i\| = 1$ for $i \in \{1, 2, 3\}$. For the student network, random Gaussian weight initialization is used unless specified otherwise. The training data consists of n points sampled uniformly from the unit sphere in $\mathbf{R}^d$. The training method is based on stochastic gradient descent applied to the test loss mainly. Compared to the original Julia code that we converted to Python, we introduced early stopping.

In the following, we discuss two key concepts: scaling by α and initialization τ-scale.

- **Scaling by** α: This refers to multiplying the output of the network by α, i.e., $\alpha \cdot h(w)$.
- **Initialization** τ**-scale**: This involves setting the standard deviation of the weight initialization to τ, where the weights are initialized by drawing from a normal distribution $\mathcal{N}(0, \tau^2)$.

4.2 Visualization of Lazy Training

In [2], lazy training is demonstrated by visualizing the movement of individual weights on a 2D plot. This showed that, in the lazy regime, weights move only slightly from their initialization to their final values at the end of training. However, we aimed to provide a clearer and more quantitative visualization of lazy training.

The theoretical framework suggests that in the lazy regime, the model behaves almost like a linearized model, with only minimal changes to the weights during training. To explore this, we investigated how the weights are modified over the course of training

compared to their initial values, w_0. Specifically, we plotted the sum of the absolute differences between the current weights and their initialization, calculated at each training iteration:

$$S_t = \sum_i \frac{|w_{t,i} - w_{0,i}|}{|w_{0,i}|},$$

Using the same experimental setup as the original paper, our results—illustrated in Figure 3—show that in the lazy regime, the weights change very little and quickly stabilize. In contrast, under non-lazy training, the weights undergo continuous and substantial changes throughout training, capturing complex nonlinear relationships. This demonstrates empirically two key aspects:

- **Fundamental differences in parameter dynamics**: In the lazy regime, the model parameters exhibit minimal movement, closely following the linearized gradient flow. In contrast, the non-lazy regime involves significant parameter updates throughout training, enabling the model to capture complex, nonlinear relationships.
- **Faster Convergence in the Lazy Regime**: The lazy regime achieves faster convergence due to its simplified, near-linear dynamics. This has been theoretically proven for strongly convex losses, where lazy training exhibits linear convergence to a global or local minimum depending on the degree of parameterization. In our observations, the training loss converges faster in the case of a lazy model.

4.3 Impact of τ on Laziness

The objective of this experiment is to observe the plateau phenomenon described in section 3.3.

For this experiment in 3, we used the exact same Julia code as in the original paper. However, we modified the code by removing the symmetrized initialization and extended the range of τ values from 10^{-1} to 10^6.

We observe a distinct phenomenon of plateauing. While this phenomenon is not directly observed through the lazy criterion as described in the theoretical section, it is evident in the behavior of the test loss. Specifically, the test loss exhibits a plateau, indicating that beyond a certain value of τ, the model enters a lazy training regime. In this regime, the model fails to capture the nonlinear relationships in the data, leading to a higher test loss.

Here, we also observe the phenomenon explained in Section 3.3. In the case of symmetrized initialization, the model's output at initialization, $h(w_0)$, is close to zero. Consequently, the lazy training criterion is satisfied at lower values of τ compared to non-symmetrized initialization. This implies that the lazy training regime is achieved earlier with respect to τ when using symmetrized initialization.

4.4 Correlation between τ for initialization and α the scaling

In this experiment, we aimed to demonstrate the inverse correlation derived in the theoretical section, where we verify that $\alpha \propto \frac{1}{\tau^q}$ in order to maintain the same value for the $\kappa_{\alpha h}$ criterion. We calculated the test loss of the model at convergence for each pair of (τ, α), with α representing the scaling parameter. Convergence was defined using early stopping, requiring an improvement of at least 10^{-4} over 100 training iterations for the loss.

The experiment was repeated three times, and the test loss values were averaged. To improve visibility, we plotted the test loss in logarithm scale.

As shown in Figure 5, to maintain the same level of test loss, α decays proportionally to $\frac{1}{\tau^q}$, where in this experiment$q = 2$ remains undetermined due to our two layers neural network student.

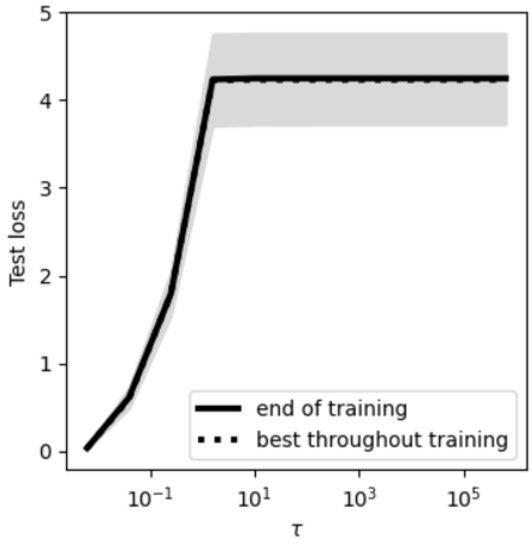

Figure 3: The plateau phenomenon as τ increases in the case of non-symmetrized initialization.

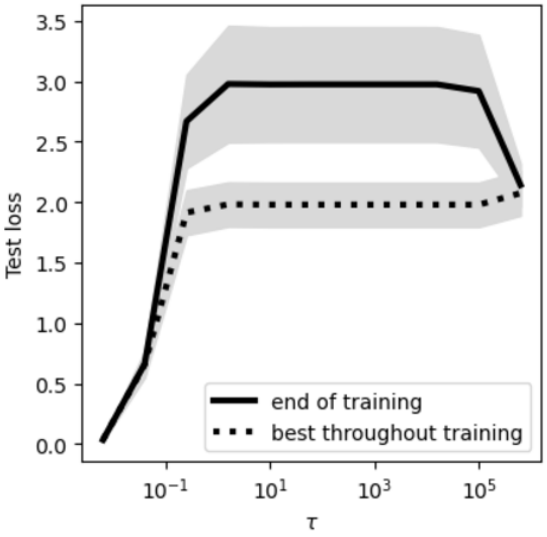

Figure 4: The plateau phenomenon as τ increases in the case of symmetrized initialization.

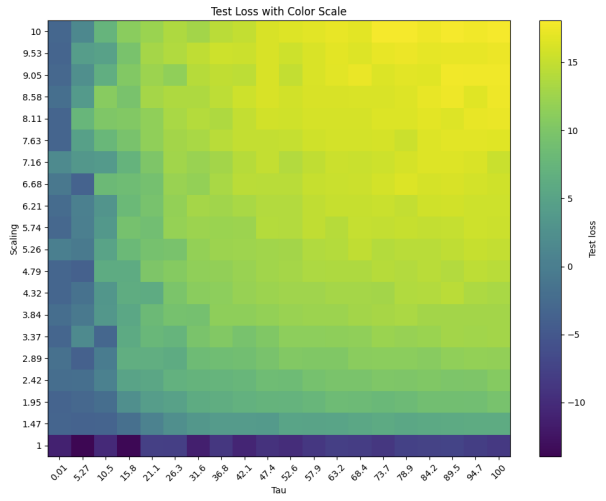

Figure 5: Inverse correlation between τ and scaling.

4.5 Asymmetric Initialization and Number of Parameters

In this setting the symmetry of the initialization is relaxed and the effect of the number of parameters is studied. Two scaling factors of $1/\sqrt{m}$ and $1/m$ are studied showing the first case leads to loss of generalization while the second remains stable as m grows.

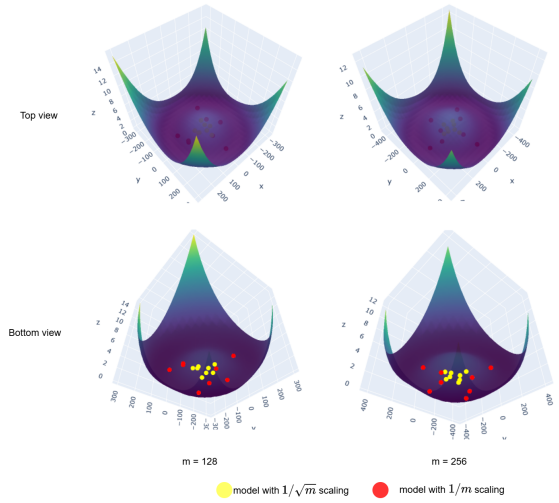

Figure 6: Loss landscape showing convergence of models with different output scaling.

4.6 Underparameterized case

In this setting the parameters are set such that $m << d$ and under the symmetrized initialization it is shown that in case of large variance of initialization weights τ^2 the model converges to a local minimum. We do more extensive visualization in this case.

4.7 visualization of loss landscape

Loss landscape visualization[5] can explain how the optimization dynamics of the model works. To this end, we have visualized the loss landscape for the two-layer neural network under $1/m$ and $1/\sqrt{m}$ scalings. Figure 2 depicts the loss landscape for different values of m. What can be seen is that for $1/m$ the loss landscape is wide under different values of m as m increases but as m increases significantly a small bump keeps appearing in the middle of the loss landscape. However, for $1/\sqrt{m}$ scaling the loss landscape is narrow and deep for small m and as m grows the depth grows as well but it becomes wider at the bottom. It is well known that a wider loss landscape leads to a better generalization.

For further analysis, we projected the weights of the model with $1/\sqrt{m}$ scaling onto the loss landscape of the model with $1/m$ scaling. As expected, the models with $1/\sqrt{m}$ scaling converge at the bump location where the loss is higher. This observation aligns with the theoretical findings, where we demonstrated that for $\alpha(m) = 1/\sqrt{m}$, the following holds:

$$\mathrm{E}[k_{h_m}(w_0)] \leq 2m^{-\frac{1}{2}},$$

indicating an upper bound that decreases faster compared to $\alpha(m) = 1/m$. This results in more lazy behavior as m increases. Figure 6 illustrates this phenomenon.

5 Discussion and limitations

5.1 Hyperparameters: Step Size

In this section, we discuss the choice of the step size hyperparameter in the numerical implementation of the paper.

The original paper explains that in the 'lazy' training regime, gradient-based methods like SGD may converge to a local minimum. However, as highlighted in [7], this behavior may be more a matter of momentum, initialization, and parameter choices than the laziness of the training itself.

[2] includes a classical experiment that we extended, as shown in Figure 3, demonstrating how the loss increases with τ. However, this experiment was conducted using a step size defined in a somewhat obscure manner as:

$$\text{stepsize} = \min(10, \frac{0.1}{\tau^2}).$$

This choice of step size is neither explicitly discussed nor justified in [2], leaving a critical detail unexplained. Specifically, in the case of non-symmetrized initialization with a large τ (where parameters are initialized as $N(0, \tau^2)$), the resulting larger weights generally correspond to larger gradients. Intuitively, this might suggest that a larger step size would be appropriate, particularly when the parameters are far from optimal.

However, the implemented step size rule paradoxically decreases the step size as τ increases. This design choice could reflect an effort to stabilize training by mitigating the risk of exploding gradients, which are more likely with larger weights. While this approach may improve stability, it potentially slows convergence in cases where the gradients are well-behaved. This interplay between step size, initialization, and stability warrants further clarification and analysis.

To simplify and clarify this aspect, we instead set the step size to a fixed value of 0.005. The results of this modification are shown in Figure 7.

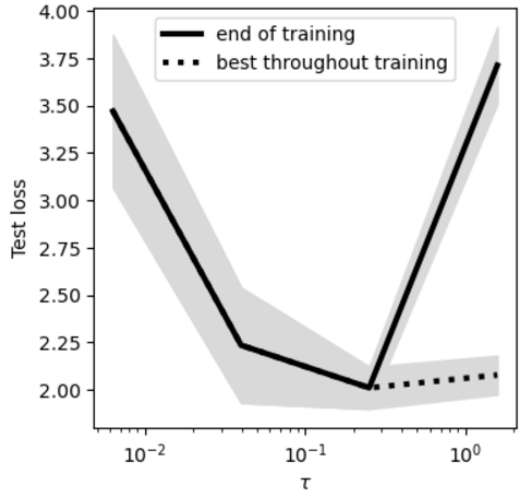

Figure 7: Plot of the test loss compared to τ for initialization.

As observed in Figure 7, we obtained the inverse of the original experiment's results (focusing on the best throughout training, represented by the dotted line) by modifying this crucial yet poorly explained parameter. This raises the question: can we genuinely characterize the training as "lazy" as τ increases under this setup?

We hypothesize that a similar figure as Figure 3 could be reproduced with lower step size values for smaller τ. Nevertheless, this observation highlights the critical importance of selecting appropriate hyperparameters and thoroughly explaining their role in the training process.

5.2 Convergence speed in lazy and non-lazy regimes

Although it is argued in [2] that a lazy training regime helps with a faster convergence, but it is not shown empirically by experiments. It is important to provide empirical justification for such arguments. However, we provide empirical support and show by experiment that indeed a lazy training regime helps with faster convergence.

5.3 Real-life Applications of Lazy Training

In practical settings, large weight initializations and significant output scaling factors that induce lazy training are generally not employed. For example, in the CNN experiments of [2], weights are initialized with large τ values—a practice that would not be used in real-world training scenarios. While [2] provides valuable insights into the conditions and mechanisms underlying lazy training, the applicability of such conditions to real-world scenarios is somewhat limited.

6 Conclusion

In this study, we further explored the dynamics of lazy training on top of existing theoretical frameworks in neural networks. Scaling factors in weight initialization and output scaling are the main reasons for inducing a lazy training regime, where models behave linearly around their initialization. Our further theoretical analysis and experiments with two-layer neural networks provide insights into how larger scaling factors or larger scales at initialization lead to faster convergence by limiting parameter updates, though at the expense of not capturing complex nonlinear relationships. Additionally, we identified an inverse correlation between weight initialization scales and output scaling factors, which maintains the $\kappa_h(w_0)$ criterion across different (α,τ) couples. Our visualizations of the loss landscape further illustrated how scaling influences optimization paths and generalization. Future work may extend these insights to more complex architectures, with concrete scaling and initialization, for a better understanding of how neural networks learn.

References

[1] Lenaic Chizat and Francis Bach. 2018. On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in neural information processing systems* 31 (2018).

[2] Lenaic Chizat, Edouard Oyallon, and Francis Bach. 2019. On lazy training in differentiable programming. *Advances in neural information processing systems* 32 (2019).

[3] Arthur Jacot, Franck Gabriel, and Clément Hongler. 2018. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems* 31 (2018).

[4] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[5] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems* 31 (2018).

[6] Ali Rahimi and Benjamin Recht. 2007. Random features for large-scale kernel machines. *Advances in neural information processing systems* 20 (2007).

[7] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. 2013. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*. PMLR, 1139–1147.

[8] Aditya Vardhan Varre, Maria-Luiza Vladarean, Loucas Pillaud-Vivien, and Nicolas Flammarion. 2024. On the spectral bias of two-layer linear networks. *Advances in Neural Information Processing Systems* 36 (2024).

A Appendix: Detailed Analysis of τ and the Plateau Phenomenon

A.1 Impact of Initialization Variance τ on the $\kappa_h(w_0)$ Criterion

This appendix provides a detailed mathematical demonstration of how the initialization variance parameter τ influences the condition number $\kappa_h(w_0)$ and the resulting plateau phenomenon.

A.1.1 Objective. We aim to analyze how τ, which scales weights initialized from a normal distribution $N(0, \tau^2)$, affects the criterion:

$$\kappa_h(w_0) = \frac{\|h(w_0) - y^\star\| \cdot \|D^2 h(w_0)\|}{\|Dh(w_0)\|^2}.$$

Here:

- $h : \mathbb{R}^p \to \mathcal{F}$ represents the model,
- w_0 is the initial weight vector, $w_0 \sim N(0, \tau^2 I)$,
- $y^\star$ is the target output,
- $Dh(w_0)$ and $D^2 h(w_0)$ are the first and second derivatives of h evaluated at w_0.

A.1.2 Assumptions.

(1) **Homogeneity of the Model:** The model h is assumed to be positively homogeneous of degree q:

$$h(\lambda w) = \lambda^q h(w) \quad \forall \lambda > 0, \quad w \in \mathbb{R}^p.$$

(2) **Initialization:** Weights are initialized as $w_0 \sim N(0, \tau^2 I)$, ensuring that the initialization variance is controlled by τ.

A.1.3 Scaling Behavior from Homogeneity. Given the homogeneity of h, the scaling of $h(w_0)$, $Dh(w_0)$, and $D^2 h(w_0)$ with respect to τ can be derived as follows:

Scaling of $h(w_0)$:

$$h(w_0) = h\left(\frac{w_0}{\tau} \cdot \tau\right) = \tau^q h\left(\frac{w_0}{\tau}\right),$$

where $h(w_0)$ scales as τ^q.

Scaling of $Dh(w_0)$: Differentiating the homogeneity property:

$$Dh(w_0) = \tau^{q-1} Dh\left(\frac{w_0}{\tau}\right),$$

indicating that $Dh(w_0)$ scales as τ^{q-1}.

Scaling of $D^2 h(w_0)$: Similarly,

$$D^2 h(w_0) = \tau^{q-2} D^2 h\left(\frac{w_0}{\tau}\right),$$

implying that $D^2 h(w_0)$ scales as τ^{q-2}.

A.1.4 Simplifying $\kappa_h(w_0)$ Using τ-Scaling. Substituting the scaling relationships into the definition of $\kappa_h(w_0)$, we have:

$$\kappa_h(w_0) = \frac{\|h(w_0) - y^\star\| \cdot \|D^2 h(w_0)\|}{\|Dh(w_0)\|^2}.$$

Substituting the scaling relationships:

$$\kappa_h(w_0) = \frac{\|\tau^q h(w') - y^\star\| \cdot \tau^{q-2} \|D^2 h(w')\|}{\tau^{2(q-1)} \|Dh(w')\|^2}.$$

Simplifying further:

$$\kappa_h(w_0) = \frac{\tau^q \|h(w') - y^\star/\tau^q\| \cdot \tau^{q-2} \|D^2 h(w')\|}{\tau^{2(q-1)} \|Dh(w')\|^2}.$$

Canceling out τ-terms:

$$\kappa_h(w_0) = \frac{\|h(w') - y^\star/\tau^q\| \cdot \|D^2 h(w')\|}{\|Dh(w')\|^2}.$$

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009