HAL
open science

On Lazy Training in Differentiable Programming

Lenaic Chizat, Edouard Oyallon, Francis Bach

To cite this version:

Lenaic Chizat, Edouard Oyallon, Francis Bach. On Lazy Training in Differentiable Programming. NeurIPS 2019 - 33rd Conference on Neural Information Processing Systems, Dec 2019, Vancouver, Canada. pp.2937-2947. hal-01945578v6

HAL Id: hal-01945578

https://inria.hal.science/hal-01945578v6

Submitted on 7 Jan 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

On Lazy Training in Differentiable Programming

Lénaïc Chizat
CNRS, Université Paris-Sud
Orsay, France
lenaic.chizat@u-psud.fr

Edouard Oyallon
CentraleSupelec, INRIA
Gif-sur-Yvette, France
edouard.oyallon@centralesupelec.fr

Francis Bach
INRIA, ENS, PSL Research University
Paris, France
francis.bach@inria.fr

Abstract

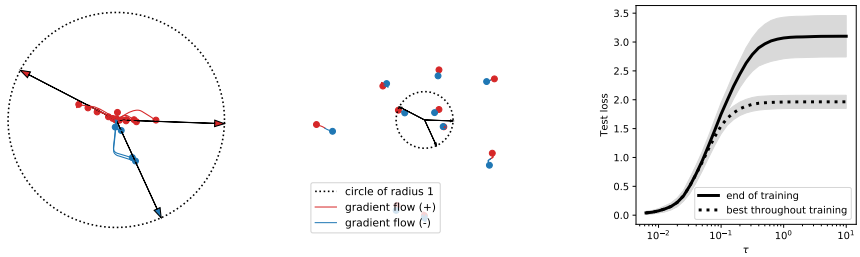
In a series of recent theoretical works, it was shown that strongly over-parameterized neural networks trained with gradient-based methods could converge exponentially fast to zero training loss, with their parameters hardly varying. In this work, we show that this "lazy training" phenomenon is not specific to over-parameterized neural networks, and is due to a choice of scaling, often implicit, that makes the model behave as its linearization around the initialization, thus yielding a model equivalent to learning with positive-definite kernels. Through a theoretical analysis, we exhibit various situations where this phenomenon arises in non-convex optimization and we provide bounds on the distance between the lazy and linearized optimization paths. Our numerical experiments bring a critical note, as we observe that the performance of commonly used non-linear deep convolutional neural networks in computer vision degrades when trained in the lazy regime. This makes it unlikely that "lazy training" is behind the many successes of neural networks in difficult high dimensional tasks.

1 Introduction

Differentiable programming is becoming an important paradigm in signal processing and machine learning that consists in building parameterized models, sometimes with a complex architecture and a large number of parameters, and adjusting these parameters in order to minimize a loss function using gradient-based optimization methods. The resulting problem is in general highly non-convex. It has been observed empirically that, for fixed loss and model class, changes in the parameterization, optimization procedure, or initialization could lead to a selection of models with very different properties [36]. This paper is about one such implicit bias phenomenon, that we call *lazy training*, which corresponds to the model behaving like its linearization around the initialization.

This work is motivated by a series of recent articles [11, 22, 10, 2, 37] where it is shown that over-parameterized neural networks could converge linearly to zero training loss with their parameters hardly varying. With a slightly different approach, it was shown in [17] that infinitely wide neural networks behave like the linearization of the neural network around its initialization. In the present work, we argue that this behavior is not specific to neural networks, and is not so much due to over-parameterization than to an implicit choice of scaling. By introducing an explicit scale factor, we show that essentially any parametric model can be trained in this lazy regime if its output is close to zero at initialization. This shows that guaranteed fast training is indeed often possible, but

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.



(a) Non-lazy training ($\tau = 0.1$) (b) Lazy training ($\tau = 2$) (c) Generalization properties

Figure 1: Training a two-layer ReLU neural network initialized with normal random weights of variance τ^2: lazy training occurs when τ is large. (a)-(b) Trajectory of weights during gradient descent in 2-D (color shows sign of output layer). (c) Generalization in 100-D: it worsens as τ increases. The ground truth is generated with 3 neurons (arrows in (a)-(b)). Details in Section 3.

at the cost of recovering a linear method[1]. Our experiments on two-layer neural networks and deep convolutional neural networks (CNNs) suggest that this behavior is undesirable in practice.

1.1 Presentation of lazy training

We consider a parameter space[2] $\mathbb{R}^p$, a Hilbert space $\mathcal{F}$, a smooth model $h : \mathbb{R}^p \to \mathcal{F}$ (such as a neural network) and a smooth loss $R : \mathcal{F} \to \mathbb{R}_+$. We aim to minimize, with gradient-based methods, the objective function $F : \mathbb{R}^p \to \mathbb{R}_+$ defined as

$$F(w) \coloneqq R(h(w)).$$

With an initialization $w_0 \in \mathbb{R}^p$, we define the linearized model $\bar{h}(w) = h(w_0) + Dh(w_0)(w - w_0)$ around w_0, and the corresponding objective $\bar{F} : \mathbb{R}^p \to \mathbb{R}_+$ as

$$\bar{F}(w) \coloneqq R(\bar{h}(w)).$$

It is a general fact that the optimization path of F and $\bar{F}$ starting from w_0 are close at the beginning of training. We call *lazy training* the less expected situation where these two paths remain close until the algorithm is stopped.

Showing that a certain non-convex optimization is in the lazy regime opens the way for surprisingly precise results, because linear models are rather well understood. For instance, when R is strongly convex, gradient descent on $\bar{F}$ with an appropriate step-size converges linearly to a global minimizer [4]. For two-layer neural networks, we show in Appendix A.2 that the linearized model is a random feature model [26] which lends itself nicely to statistical analysis [6]. Yet, while advantageous from a theoretical perspective, it is not clear *a priori* whether this lazy regime is desirable in practice.

This phenomenon is illustrated in Figure 1 where lazy training for a two-layer neural network with rectified linear units (ReLU) is achieved by increasing the variance τ^2 at initialization (see next section). While in panel (a) the ground truth features are identified, this is not the case for lazy training on panel (b) that manages to interpolate the observations with just a small displacement in parameter space (in both cases, near zero training loss was achieved). As seen on panel (c), this behavior hinders good generalization in the teacher-student setting [30]. The plateau reached for large τ corresponds exactly to the performance of the linearized model, see Section 3.1 for details.

1.2 When does lazy training occur?

A general criterion. Let us start with a formal computation. We assume that w_0 is not a minimizer so that $F(w_0) > 0$, and not a critical point so that $\nabla F(w_0) \neq 0$. Consider a gradient descent step $w_1 \coloneqq w_0 - \eta \nabla F(w_0)$, with a small stepsize $\eta > 0$. On the one hand, the relative change of the objective is $\Delta(F) \coloneqq \frac{|F(w_1) - F(w_0)|}{F(w_0)} \approx \eta \frac{\|\nabla F(w_0)\|^2}{F(w_0)}$. On the other hand, the relative change of the

[1] Here we mean a prediction function linearly parameterized by a potentially infinite-dimensional vector.

[2] Our arguments could be generalized to the case where the parameter space is a Riemannian manifold.

differential of h measured in operator norm is $\Delta(Dh) := \frac{\|Dh(w_1)-Dh(w_0)\|}{\|Dh(w_0)\|} \leq \eta \frac{\|\nabla F(w_0)\| \cdot \|D^2 h(w_0)\|}{\|Dh(w_0)\|}$. Lazy training refers to the case where the differential of h does not sensibly change while the loss enjoys a significant decrease, i.e., $\Delta(F) \gg \Delta(Dh)$. Using the above estimates, this is guaranteed when

$$\frac{\|\nabla F(w_0)\|}{F(w_0)} \gg \frac{\|D^2 h(w_0)\|}{\|Dh(w_0)\|}.$$

For the square loss $R(y) = \frac{1}{2}\|y - y^\star\|^2$ for some $y^\star \in \mathcal{F}$, this leads to the simpler criterion

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

using the approximation $\|\nabla F(w_0)\| = \|Dh(w_0)^\intercal (h(w_0) - y^\star)\| \approx \|Dh(w_0)\| \cdot \|h(w_0) - y^\star\|$. This quantity $\kappa_h(w_0)$ could be called the inverse *relative scale* of the model h at w_0. We prove in Theorem 2.3 that it indeed controls how much the training dynamics differs from the linearized training dynamics when R is the square loss[3]. For now, let us explore situations in which lazy training can be shown to occur, by investigating the behavior of $\kappa_h(w_0)$.

Rescaled models. Considering a scaling factor $\alpha > 0$, it holds

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

Thus, $\kappa_{\alpha h}(w_0)$ simply decreases as α^{-1} when α grows and $\|\alpha h(w_0) - y^\star\|$ is bounded, leading to lazy training for large α. Training dynamics for such rescaled models are studied in depth in Section 2. For neural networks, there are various ways to ensure $h(w_0) = 0$, see Section 3.

Homogeneous models. If h is q-positively homogeneous[4] then multiplying the initialization by λ is equivalent to multiplying the scale factor α by λ^q. In equation,

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

This formula applies for instance to q-layer neural networks consisting of a cascade of homogenous non-linearities and linear, but not affine, operators. Such networks thus enter the lazy regime as the variance of initialization increases, if one makes sure that the initial output has bounded norm (see Figures 1 and 2(b) for 2-homogeneous examples).

Two-layer neural networks. For $m, d \in \mathbb{N}$, consider functions $h_m : (\mathbb{R}^d)^m \to \mathcal{F}$ of the form

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

where $\alpha(m) > 0$ is a normalization, $w = (\theta_1, \ldots, \theta_m)$ and $\phi : \mathbb{R}^d \to \mathcal{F}$ is a smooth function. This setting covers the case of two-layer neural networks (see Appendix A.2). When initializing with independent and identically distributed variables $(\theta_i)_{i=1}^m$ satisfying $\mathbb{E}\phi(\theta_i) = 0$, and under the assumption that $D\phi$ is not identically 0 on the support of the initialization, we prove in Appendix A.2 that for large m it holds

$$\mathbb{E}[\kappa_{h_m}(w_0)] \lesssim m^{-\frac{1}{2}} + (m\alpha(m))^{-1}.$$

As a consequence, as long as $m\alpha(m) \to \infty$ when $m \to \infty$, such models are bound to reach the lazy regime. In this case, the norm of the initial output becomes negligible in front of the scale as m grows due to the statistical cancellations that follow from the assumption $\mathbb{E}\phi(\theta_i) = 0$. In contrast, the critical scaling $\alpha(m) = 1/m$, allows to converge as $m \to \infty$ to a non degenerate dynamic described by a partial differential equation and referred to as the mean-field limit [24, 7, 29, 33].

[3] Note that lazy training could occur even when $\kappa_h(w_0)$ is large, i.e. Eq. (1) only gives a sufficient condition.

[4] That is, for $q \geq 1$, it holds $h(\lambda w) = \lambda^q h(w)$ for all $\lambda > 0$ and $w \in \mathbb{R}^p$.

1.3 Content and contributions

The goal of this paper is twofold: (i) understanding in a general optimization setting when lazy training occurs, and (ii) investigating the practical usefulness of models in the lazy regime. It is organized as follows:

- in Section 2, we study the gradient flows for rescaled models αh and prove in various situations that for large α, they are close to gradient flows of the linearized model. When the loss is strongly convex, we also prove that lazy gradient flows converge linearly, either to a global minimizer for over-parameterized models, or to a local minimizer for under-parameterized models.
- in Section 3, we use numerical experiments on synthetic cases to illustrate how lazy training differs from other regimes of training (see also Figure 1). Most importantly, we show empirically that CNNs used in practice could be far from the lazy regime, with their performance not exceeding that of some classical linear methods as they become lazy.

Our focus is on general principles and qualitative description.

Updates of the paper. This article is an expanded version of "A Note on Lazy Training in Supervised Differential Programming" that appeared online in December 2018. Compared to the first version, it has been complemented with finite horizon bounds in Section 2.2 and numerical experiments on CNNs in Section 3.2 while the rest of the material has just been slightly reorganized.

2 Analysis of Lazy Training Dynamics

2.1 Theoretical setting

Our goal in this section is to show that lazy training dynamics for the scaled objective

$$F_\alpha(w) := \frac{1}{\alpha^2} R(\alpha h(w)) \tag{2}$$

are close, when the scaling factor α is large, to those of the scaled objective for the linearized model

$$\bar{F}_\alpha(w) := \frac{1}{\alpha^2} R(\alpha \bar{h}(w)), \tag{3}$$

where $\bar{h}(w) := h(w_0) + Dh(w_0)(w - w_0)$ and $w_0 \in \mathbb{R}^p$ is a fixed initialization. Multiplying the objective by $1/\alpha^2$ does not change the minimizers, and corresponds to the proper time parameterization of the dynamics for large α. Our basic assumptions are the following:

Assumption 2.1. *The parametric model $h : \mathbb{R}^p \to \mathcal{F}$ is differentiable with a locally Lipschitz differential*[5] *Dh. Moreover, R is differentiable with a Lipschitz gradient.*

This setting is mostly motivated by supervised learning problems, where one considers a probability distribution $\rho \in \mathcal{P}(\mathbb{R}^d \times \mathbb{R}^k)$ and defines $\mathcal{F}$ as the space $L^2(\rho_x; \mathbb{R}^k)$ of square-integrable functions with respect to ρ_x, the marginal of ρ on $\mathbb{R}^d$. The risk R is then built from a smooth loss function $\ell : (\mathbb{R}^k)^2 \to \mathbb{R}_+$ as $R(g) = \mathbb{E}_{(X,Y)\sim\rho}\ell(g(X), Y)$. This corresponds to empirical risk minimization when ρ is a finite discrete measure, and to population risk minimization otherwise (in which case only stochastic gradients are available to algorithms). Finally, one defines $h(w) = f(w, \cdot)$ where $f : \mathbb{R}^p \times \mathbb{R}^d \to \mathbb{R}^k$ is a parametric model, such as a neural network, which outputs in $\mathbb{R}^k$ depend on parameters in $\mathbb{R}^p$ and input data in $\mathbb{R}^d$.

Gradient flows. In the rest of this section, we study the *gradient flow* of the objective function F_α which is an approximation of (accelerated) gradient descent [12, 31] and stochastic gradient descent [19, Thm. 2.1] with small enough step sizes. With an initialization $w_0 \in \mathbb{R}^p$, the gradient

[5] $Dh(w)$ is a continuous linear map from $\mathbb{R}^p$ to $\mathcal{F}$. The Lipschitz constant of $Dh : w \mapsto Dh(w)$ is defined with respect to the operator norm. When $\mathcal{F}$ has a finite dimension, $Dh(w)$ can be identified with the Jacobian matrix of h at w.

flow of F_α is the path $(w_\alpha(t))_{t\geq 0}$ in the space of parameters $\mathbb{R}^p$ that satisfies $w_\alpha(0) = w_0$ and solves the ordinary differential equation

$$w'_\alpha(t) = -\nabla F_\alpha(w_\alpha(t)) = -\frac{1}{\alpha} Dh(w_\alpha(t))^\intercal \nabla R(\alpha h(w_\alpha(t))), \tag{4}$$

where $Dh^\intercal$ denotes the adjoint of the differential Dh. We will study this dynamic for itself, and will also compare it to the gradient flow $(\bar{w}_\alpha(t))_{t\geq 0}$ of $\bar{F}_\alpha$ that satisfies $\bar{w}_\alpha(0) = w_0$ and solves

$$\bar{w}'_\alpha(t) = -\nabla \bar{F}_\alpha(\bar{w}_\alpha(t)) = -\frac{1}{\alpha} Dh(w_0)^\intercal \nabla R(\alpha \bar{h}(\bar{w}_\alpha(t))). \tag{5}$$

Note that when $h(w_0) = 0$, the renormalized dynamic $w_0 + \alpha(\bar{w}_\alpha(t) - w_0)$ does not depend on α, as it simply follows the gradient flow of $w \mapsto R(Dh(w_0)(w - w_0))$ starting from w_0.

2.2 Bounds with a finite time horizon

We start with a general result that confirms that when $h(w_0) = 0$, taking large α leads to lazy training. We do not assume convexity of R.

Theorem 2.2 (General lazy training)**.** *Assume that $h(w_0) = 0$. Given a fixed time horizon $T > 0$, it holds $\sup_{t\in[0,T]} \|w_\alpha(t) - w_0\| = O(1/\alpha)$,*

$$\sup_{t\in[0,T]} \|w_\alpha(t) - \bar{w}_\alpha(t)\| = O(1/\alpha^2) \quad \textit{and} \quad \sup_{t\in[0,T]} \|\alpha h(w_\alpha(t)) - \alpha \bar{h}(\bar{w}_\alpha(t))\| = O(1/\alpha).$$

For supervised machine learning problems, the bound on $\|w_\alpha(t) - \bar{w}_\alpha(t)\|$ implies that $\alpha h(w_\alpha(T))$ also *generalizes* like $\alpha \bar{h}(\bar{w}_\alpha(T))$ outside of the training set for large α, see Appendix A.3. Note that the generalization behavior of linear models has been widely studied, and is particularly well understood for random feature models [26], which are recovered when linearizing two layer neural networks, see Appendix A.2. It is possible to track the constants in Theorem 2.2 but they would depend exponentially on the time horizon T. This exponential dependence can however be discarded for the specific case of the square loss, where we recover the *scale* criterion informally derived in Section 1.2.

Theorem 2.3 (Square loss, quantitative)**.** *Consider the square loss $R(y) = \frac{1}{2}\|y - y^\star\|^2$ for some $y^\star \in \mathcal{F}$ and assume that for some (potentially small) $r > 0$, h is $\mathrm{Lip}(h)$-Lipschitz and Dh is $\mathrm{Lip}(Dh)$-Lipschitz on the ball of radius r around w_0. Then for an iteration number $K > 0$ and corresponding time $T \coloneqq K/\mathrm{Lip}(h)^2$, it holds*

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

as long as $\alpha \geq K\|\alpha h(w_0) - y^\star\|/(r\mathrm{Lip}(h))$.

We can make the following observations:

- For the sake of interpretability, we have introduced a quantity K, analogous to an iteration number, that accounts for the fact that the gradient flow needs to be integrated with a step-size of order $1/\mathrm{Lip}(\nabla F_\alpha) = 1/\mathrm{Lip}(h)^2$. For instance, with this step-size, gradient descent at iteration K approximates the gradient flow at time $T = K/\mathrm{Lip}(h)^2$, see, e.g., [12, 31].
- Laziness only depends on the local properties of h around w_0. These properties may vary a lot over the parameter space, as is the case for homogeneous functions seen in Section 1.2.

For completeness, similar bounds on $\|w_\alpha(T) - w_0\|$ and $\|w_\alpha(T) - \bar{w}_\alpha(T)\|$ are also provided in Appendix B.2. The drawback of the bounds in this section is the increasing dependency in time, which is removed in the next section. Yet, the relevance of Theorem 2.2 remains because it does not depend on the conditioning of the problem. Although the bound grows as K^2, it gives an informative estimate for large or ill-conditioned problems, where training is typically stopped much before convergence.

2.3 Uniform bounds and convergence in the lazy regime

This section is devoted to uniform bounds in time and convergence results under the assumption that R is strongly convex. In this setting, the function $\bar{F}_\alpha$ is strictly convex on the affine hyperspace

$w_0 + \ker Dh(w_0)^\perp$ which contains the linearized gradient flow $(\bar{w}_\alpha(t))_{t\geq 0}$, so the latter converges linearly to the unique global minimizer of $\bar{F}_\alpha$. In particular, if $h(w_0) = 0$ then this global minimizer does not depend on α and $\sup_{t\geq 0} \|\bar{w}_\alpha(t) - w_0\| = O(1/\alpha)$. We will see in this part how these properties reflect on the gradient flow $w_\alpha(t)$ for large α.

Over-parameterized case. The following proposition shows global convergence of lazy training under the condition that $Dh(w_0)$ is surjective. As $\operatorname{rank} Dh(w_0)$ gives the number of effective parameters or degrees of freedom of the model around w_0, this over-parameterization assumption guarantees that any model around $h(w_0)$ can be fitted. Of course, this can only happen if $\mathcal{F}$ is finite-dimensional.

Theorem 2.4 (Over-parameterized lazy training)**.** *Consider a M-smooth and m-strongly convex loss R with minimizer $y^\star$ and condition number $\kappa := M/m$. Assume that $\sigma_{\min}$, the smallest singular value of $Dh(w_0)^\intercal$ is positive and that the initialization satisfies $\|h(w_0)\| \leq C_0 := \sigma_{\min}^3/(32\kappa^{3/2}\|Dh(w_0)\|\mathrm{Lip}(Dh))$ where $\mathrm{Lip}(Dh)$ is the Lipschitz constant of Dh. If $\alpha > \|y^*\|/C_0$, then for $t \geq 0$, it holds*

$$\|\alpha h(w_\alpha(t)) - y^*\| \leq \sqrt{\kappa}\|\alpha h(w_0) - y^*\| \exp(-m\sigma_{\min}^2 t/4).$$

If moreover $h(w_0) = 0$, it holds as $\alpha \to \infty$, $\sup_{t\geq 0} \|w_\alpha(t) - w_0\| = O(1/\alpha)$,

$$\sup_{t\geq 0} \|\alpha h(w_\alpha(t)) - \alpha \bar{h}(\bar{w}_\alpha(t))\| = O(1/\alpha) \quad \textit{and} \quad \sup_{t\geq 0} \|w_\alpha(t) - \bar{w}_\alpha(t)\| = O(\log \alpha/\alpha^2).$$

The proof of this result relies on the fact that $\alpha h(w_\alpha(t))$ follows the gradient flow of R in a time-dependent and non degenerate metric: the pushforward metric [21] induced by h on $\mathcal{F}$. For the first part, we do not claim improvements over [11, 22, 10, 2, 37], where a lot of effort is also put in dealing with the non-smoothness of h, which we do not study here. As for the uniform in time comparison with the tangent gradient flow, it is new and follows mostly from Lemma B.2 in Appendix B where the constants are given and depend polynomially on the characteristics of the problem.

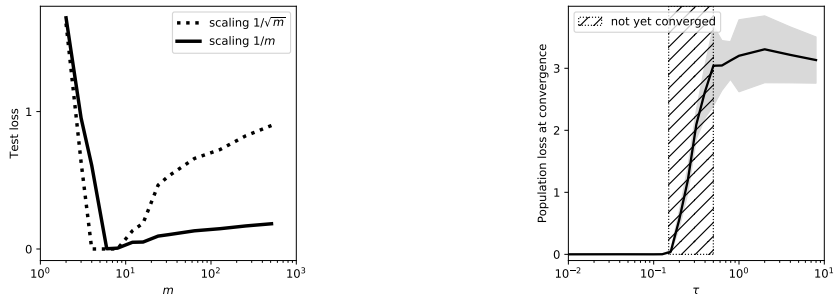
Under-parameterized case. We now remove the over-parameterization assumption and show again linear convergence for large values of α. This covers in particular the case of population loss minimization, where $\mathcal{F}$ is infinite-dimensional. For this setting, we limit ourselves to a qualitative statement[6].

Theorem 2.5 (Under-parameterized lazy training)**.** *Assume that $\mathcal{F}$ is separable, R is strongly convex, $h(w_0) = 0$ and $\operatorname{rank} Dh(w)$ is constant on a neighborhood of w_0. Then there exists $\alpha_0 > 0$ such that for all $\alpha \geq \alpha_0$ the gradient flow* (4) *converges at a geometric rate (asymptotically independent of α) to a local minimum of F_α.*

Thanks to lower-semicontinuity of the rank function, the assumption that the rank is locally constant holds generically, in the sense that it is satisfied on an open dense subset of $\mathbb{R}^p$. In this under-parameterized case, the limit $\lim_{t\to\infty} w_\alpha(t)$ is for α large enough a strict local minimizer, but in general not a global minimizer of F_α because the image of $Dh(w_0)$ does not *a priori* contain the global minimizer of R. Thus it cannot be excluded that there exists parameters w farther from w_0 with a smaller loss. This fact is clearly observed experimentally in Section 3, Figure 2-(b). Finally, a comparison with the linearized gradient flow as in Theorem 2.4 could be shown along the same lines, but would be technically slightly more involved because differential geometry comes into play.

Relationship to the global convergence result in [7]. A consequence of Theorem 2.5 is that in the lazy regime, the gradient flow of the population risk for a two-layer neural network might get stuck in a local minimum. In contrast, it is shown in [7] that such gradient flows converge to global optimality in the infinite over-parameterization limit $p \to \infty$ if initialized with enough diversity in the weights. This is not a contradiction since Theorem 2.5 assumes a finite number p of parameters. In the lazy regime, the population loss might also converge to its minimum when p increases: this is guaranteed if the tangent kernel $Dh(w_0)Dh(w_0)^\intercal$ [17] converges (after normalization) to a universal kernel as $p \to \infty$. However, this convergence might be unreasonably slow in high-dimension, as Figure 1-(c) suggests. As a side note, we stress that the global convergence result in [7] is not limited to lazy dynamics but also covers non-linear dynamics, such as seen on Figure 1 where neurons move.

[6] In contrast to the finite horizon bound of Theorem 2.3, quantitative statements would here involve the smallest positive singular value of $Dh(w_0)$, which is anyways hard to control.



(a) (b)

Figure 2: (a) Test loss at convergence for gradient descent, when α depends on m as $\alpha = 1/m$ or $\alpha = 1/\sqrt{m}$, the latter leading to lazy training for large m (not symmetrized). (b) Population loss at convergence versus τ for SGD with a random $\mathcal{N}(0, \tau^2)$ initialization (symmetrized). In the hatched area the loss was still slowly decreasing.

3 Numerical Experiments

We realized two sets of experiments, the first with two-layer neural networks conducted on synthetic data and the second with convolutional neural networks (CNNs) conducted on the CIFAR-10 dataset [18]. The code to reproduce these experiments is available online[7].

3.1 Two-layer neural networks in the teacher-student setting

We consider the following two-layer *student* neural network $h_m(w) = f_m(w, \cdot)$ with $f_m(w, x) = \sum_{j=1}^{m} a_j \max(b_j \cdot x, 0)$ where $a_j \in \mathbb{R}$ and $b_j \in \mathbb{R}^d$ for $j = 1, \dots, m$. It is trained to minimize the square loss with respect to the output of a two-layer *teacher* neural network with same architecture and $m_0 = 3$ hidden neurons, with random weights normalized so that $\|a_j b_j\| = 1$ for $j \in \{1, 2, 3\}$. For the student network, we use random Gaussian weights, except when *symmetrized initialization* is mentioned, in which case we use random Gaussian weights for $j \leq m/2$ and set for $j > m/2$, $b_j = b_{j-m/2}$ and $a_j = -a_{j-m/2}$. This amounts to training a model of the form $h(w_a, w_b) = h_{m/2}(w_a) - h_{m/2}(w_b)$ with $w_a(0) = w_b(0)$ and guaranties zero output at initialization. The training data are n input points uniformly sampled on the unit sphere in $\mathbb{R}^d$ and we minimize the empirical risk, except for Figure 2b(b) where we directly minimize the population risk with Stochastic Gradient Descent (SGD).

Cover illustration. Let us detail the setting of Figure 1 in Section 1. Panels (a)-(b) show gradient descent dynamics with $n = 15, m = 20$ with symmetrized initialization (illustrations with more neurons can be found in Appendix C). To obtain a 2-D representation, we plot $|a_j(t)|b_j(t)$ throughout training (lines) and at convergence (dots) for $j \in \{1, \dots, m\}$. The blue or red colors stand for the signs of $a_j(t)$ and the unit circle is displayed to help visualizing the change of scale. On panel (c), we set $n = 1000$, $m = 50$ with symmetrized initialization and report the average and standard deviation of the test loss over 10 experiments. To ensure that the bad performances corresponding to large τ are not due to a lack of regularization, we display also the best test error throughout training (for kernel methods, early stopping is a form of regularization [34]).

Increasing number of parameters. Figure 2-(a) shows the evolution of the test error when increasing m as discussed in Section 1.2, *without* symmetrized initialization. We report the results for two choices of scaling functions $\alpha(m)$, averaged over 5 experiments with $d = 100$. The scaling $1/\sqrt{m}$ leads to lazy training, with a poor generalization as m increases, in contrast to the scaling $1/m$ for which the test error remains relatively close to 0 for large m (more experiments with this scaling can be found in [7, 29, 24]).

[7] `https://github.com/edouardoyallon/lazy-training-CNN`

Under-parameterized case. Finally, Figure 2-(b) illustrates the under-parameterized case, with $d = 100, m = 50$ with symmetrized initialization. We used SGD with batch-size 200 to minimize the population square loss, and displayed average and standard deviation of the final population loss (estimated with 2000 samples) over 5 experiments. As shown in Theorem 2.5, SGD converges to a *a priori* local minimum in the lazy regime (i.e., here for large τ). In contrast, it behaves well when τ is small, as in Figure 1. There is also an intermediate regime (hatched area) where convergence is very slow and the loss was still decreasing when the algorithm was stopped.

3.2 Deep CNNs experiments

We now study whether lazy training is relevant to understand the good performances of convolutional neural networks (CNNs).

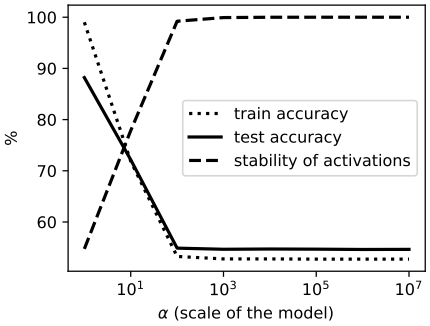
Interpolating from standard to lazy training. We first study the effect of increasing the scale factor α on a standard pipeline for image classification on the CIFAR10 dataset. We consider the VGG-11 model [32], which is a widely used model on CIFAR10. We trained it via mini-batch SGD with a momentum parameter of 0.9. For the sake of interpretability, no extra regularization (e.g., BatchNorm) is incorporated, since a simple framework that outperforms linear methods baselines with some margin is sufficient to our purpose (see Figure 3(b)). An initial learning rate η_0 is linearly decayed at each epoch, following $\eta_t = \frac{\eta_0}{1+\beta t}$. The biases are initialized with 0 and all other weights are initialized with normal Xavier initialization [13]. In order to set the initial output to 0 we use the *centered model* h, which consists in replacing the VGG model $\tilde{h}$ by $h(w) := \tilde{h}(w) - \tilde{h}(w_0)$. Notice that this does not modify the differential at initialization.

The model h is trained for the square loss multiplied by $1/\alpha^2$ (as in Section 2), with standard data-augmentation, batch-size of 128 [35] and $\eta_0 = 1$ which gives the best test accuracies over the grid 10^k, $k \in \{-3, 3\}$, for all α. The total number of epochs is 70, adjusted so that the performance reaches a plateau for $\alpha = 1$. Figure 3(a) reports the accuracy after training αh for increasing values of $\alpha \in 10^k$ for $k = \{0, 1, 2, 3, 4, 5, 6, 7\}$ ($\alpha = 1$ being the standard setting). For $\alpha < 1$, the training loss diverges with $\eta_0 = 1$. We also report the *stability of activations*, which is the share of neurons over ReLU layers that, after training, are activated for the same inputs than at initialization, see Appendix C. Values close to 100% are strong indicators of an effective linearization.

We observe a significant drop in performance as α grows, and then the accuracy reaches a plateau, suggesting that the CNN progressively reaches the lazy regime. This demonstrates that the linearized model (large α) is not sufficient to explain the good performance of the model for $\alpha = 1$. For large α, we obtain a low limit training accuracy and do not observe overfitting, a surprising fact since this amounts to solving an over-parameterized linear system. This behavior is due to a poorly conditioned linearized model, see Appendix C.

Performance of linearized CNNs. In this second set of experiments, we investigate whether variations of the models trained above in a lazy regime could increase the performance and, in particular, could outperform other linear methods which also do not involve learning a representation [26, 25]. To this end, we train widened CNNs in the lazy regime, as widening is a well-known strategy to boost performances of a given architecture [35]. We multiply the number of channels of each layer by 8 for the VGG model and 7 for the ResNet model [16] (these values are limited by hardware constraints). We choose $\alpha = 10^7$ to train the linearized models, a batch-size of 8 and, after cross-validation, $\eta_0 = 0.01, 1.0$ for respectively the standard and linearized model. We also multiply the initial weights by respectively 1.2 and 1.3 for the ResNet-18 and VGG-11, as we found that it slightly boosts the training accuracies. Each model is trained with the cross-entropy loss divided by α^2 until the test accuracy stabilizes or increases, and we check that the average stability of activations (see Appendix C) was 100%.

As seen on Figure 3(b), widening the VGG model slightly improves the performances of the linearized model compared to the previous experiment but there is still a substantial gap of performances from other non-learned representations [28, 25] methods, not to mention the even wider gap with their non-lazy counterparts. This behavior is also observed on the state-of-the-art ResNet architecture. Note that [3] reports a test accuracy of 77.4% without data augmentation for a linearized CNN with a specially designed architecture which in particular solves the issue of ill-conditioning. Whether



(a)

Model	Train acc.	Test acc.
ResNet wide, linearized	55.0	56.7
VGG-11 wide, linearized	61.0	61.7
Prior features [25]	-	82.3
Random features [28]	-	84.2
VGG-11 wide, standard	99.9	89.7
ResNet wide, standard	99.4	91.0

(b)

Figure 3: (a) Accuracies on CIFAR10 as a function of the scaling α. The stability of activations suggest a linearized regime when high. (b) Accuracies on CIFAR10 obtained for $\alpha = 1$ (standard, non-linear) and $\alpha = 10^7$ (linearized) compared to those reported for some linear methods without data augmentation: random features and prior features based on the scattering transform.

variations of standard architectures and pipelines can lead to competitive performances with linearized CNNs, remains an open question.

Remark on wide NNs. It was proved [17] that neural networks with standard initialization (random independent weights with zero mean and variance $O(1/n_\ell)$ at layer ℓ, where n_ℓ is the size of the previous layer), are bound to reach the lazy regime as the sizes of all layers grow unbounded. Moreover, for very large neural networks of more than 2 layers, this choice of initialization is essentially mandatory to avoid exploding or vanishing initial gradients [15, 14] if the weights are independent with zero mean. Thus we stress that we do not claim that wide neural networks do not show a lazy behavior, but rather that those which exhibit good performances are far from this asymptotic behavior.

4 Discussion

Lazy training is an implicit bias phenomenon in differentiable programming, that refers to the situation when a non-linear parametric model behaves like a linear one. This arises when the *scale* of the model becomes large which, we have shown, happens implicitly under some choices of hyper-parameters governing normalization, initialization and number of iterations. While the lazy training regime provides some of the first optimization-related theoretical insights for deep neural networks [10, 2, 37, 17], we believe that it does not explain yet the many successes of neural networks that have been observed in various challenging, high-dimensional tasks in machine learning. This is corroborated by numerical experiments where it is seen that the performance of networks trained in the lazy regime degrades and in particular does not exceed that of some classical linear methods. Instead, the intriguing phenomenon that still defies theoretical understanding is the one illustrated on Figure 1(c) for small τ and on Figure 3(a) for $\alpha = 1$: neural networks trained with gradient-based methods (and neurons that move) have the ability to perform high-dimensional feature selection through highly non-linear dynamics.

Acknowledgments

We acknowledge supports from grants from Région Ile-de-France and the European Research Council (grant SEQUOIA 724063). Edouard Oyallon was supported by a GPU donation from NVIDIA. We thank Alberto Bietti for interesting discussions and Brett Bernstein for noticing an error in a previous version of this paper.

References

[1] Ralph Abraham, Jerrold E. Marsden, and Tudor Ratiu. *Manifolds, Tensor Analysis, and Applications*, volume 75. Springer Science & Business Media, 2012.

[2] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 242–252, 2019.

[3] Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *arXiv preprint arXiv:1904.11955*, 2019.

[4] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

[5] Youness Boutaib. On Lipschitz maps and their flows. *arXiv preprint arXiv:1510.07614*, 2015.

[6] Luigi Carratino, Alessandro Rudi, and Lorenzo Rosasco. Learning with SGD and random features. In *Advances in Neural Information Processing Systems*, pages 10192–10203, 2018.

[7] Lénaïc Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in neural information processing systems*, 2018.

[8] Youngmin Cho and Lawrence K. Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems*, pages 342–350, 2009.

[9] Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances In Neural Information Processing Systems*, pages 2253–2261, 2016.

[10] Simon S. Du, Lee Jason D., Li Haochuan, Wang Liwei, and Zhai Xiyu. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

[11] Simon S. Du, Xiyu Zhai, Barnabás Póczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019.

[12] Walter Gautschi. *Numerical analysis*. Springer Science & Business Media, 1997.

[13] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[14] Boris Hanin and David Rolnick. How to start training: The effect of initialization and architecture. In *Advances in Neural Information Processing Systems*, pages 571–581, 2018.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

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

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

[18] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[19] Harold Kushner and G. George Yin. *Stochastic Approximation and Recursive Algorithms and Applications*, volume 35. Springer Science & Business Media, 2003.

[20] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as Gaussian processes. In *International Conference on Learning Representations*, 2018.

[21] John M. Lee. Smooth manifolds. In *Introduction to Smooth Manifolds*, pages 1–29. Springer, 2003.

[22] Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems*, pages 8167–8176, 2018.

[23] Alexander G. de G. Matthews, Mark Rowland, Jiri Hron, Richard E. Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations*, 2018.

[24] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

[25] Edouard Oyallon and Stéphane Mallat. Deep roto-translation scattering for object classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2865–2873, 2015.

[26] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.

[27] Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in neural information processing systems*, pages 1313–1320, 2009.

[28] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *Proceedings of the 36th International Conference on Machine Learning*, pages 5389–5400, 2019.

[29] Grant M. Rotskoff and Eric Vanden-Eijnden. Neural networks as interacting particle systems: Asymptotic convexity of the loss landscape and universal scaling of the approximation error. In *Advances in neural information processing systems*, 2018.

[30] David Saad and Sara A Solla. On-line learning in soft committee machines. *Physical Review E*, 52(4):4225, 1995.

[31] Damien Scieur, Vincent Roulet, Francis Bach, and Alexandre d'Aspremont. Integration methods and optimization algorithms. In *Advances in Neural Information Processing Systems*, pages 1109–1118, 2017.

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[33] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A central limit theorem. *Stochastic Processes and their Applications*, 2019.

[34] Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

[35] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12, 2016.

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

[37] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic gradient descent optimizes over-parameterized deep ReLU networks. *Machine Learning Journal*, 2019.

Supplementary material

Supplementary material for the paper: "On Lazy Training in Differentiable Programming" authored by Lénaïc Chizat , Edouard Oyallon and Francis Bach (NeurIPS 2019). This supplementary material is organized as follows:

- Appendix A: Remarks on the linearized model
- Appendix B: Proofs of the theoretical results
- Appendix C: Experimental details and additional results

A The linearized model in supervised machine learning

A.1 Differentiable models and their linearization

In this section, we give some details on the interpretation of the linearized model in the case of supervised machine learning. In this setting, a differentiable model is a typically a function $f : \mathbb{R}^p \times \mathbb{R}^d \to \mathbb{R}^k$ where $\mathbb{R}^p$ is the parameter space, $\mathbb{R}^d$ is the input space and $\mathbb{R}^k$ the output space. One defines a Hilbert space $\mathcal{F}$ of functions from $\mathbb{R}^d$ to $\mathbb{R}^k$, typically $L^2(\rho_x, \mathbb{R}^k)$ where ρ_x is the distribution of input samples. The function $h : \mathbb{R}^p \to \mathcal{F}$ considered in the article is then the function which to a vector of parameters associates a predictor $h : w \mapsto f(w, \cdot)$.

In first order approximation around the initial parameters $w_0 \in \mathbb{R}^p$, the parametric model $f(w, x)$ reduces to the following *linearized* or *tangent* model :

$$\bar{f}(w, x) = f(w_0, x) + D_w f(w_0, x)(w - w_0). \tag{6}$$

where $D_w f$ is the differential of f in the variable w. The corresponding hypothesis class is *affine* in the space of predictors. It should be stressed that when f is a neural network, $\bar{f}$ is generally not a linear neural network because it is not linear in $x \in \mathbb{R}^d$, but in the features $D_w f(w_0, x) \in \mathbb{R}^{p \times k}$ which generally depend non-linearly on x. For large neural networks, the dimension of the features might be much larger than d, which makes $\bar{f}$ similar to a non-parametric method. Finally, if f is already a linear model, then f and $\bar{f}$ are identical.

Kernel method with an offset. In the case of the square loss, training the affine model (6) is equivalent to training a linear model in the variables

$$(\tilde{x}, \tilde{y}) \coloneqq (D_w f(w_0, x), y - f(w_0, x)).$$

When $k = 1$, this is equivalent to a kernel method with the *tangent kernel* [17] defined as $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$

$$K(x, x') = D_w f(w_0, x) D_w f(w_0, x')^\intercal. \tag{7}$$

This kernel is different from the one traditionally associated to neural networks [27, 9] which involve the derivative with respect to the output layer only. Also, the output data is shifted by the initialization of the model $h(w_0) = f(w_0, \cdot)$. This term inherits from the randomness due to the initialization: it is for instance shown in [20, 23] that the distribution of $h(w_0)$ converges to a Gaussian process for certain over-parameterized neural networks initialized with random normal weights.

A.2 Two-layer neural networks

Lazy training has some interesting consequences when looking more particularly at two-layer neural networks. These are functions of the form

$$f_m(w, x) = \alpha(m) \sum_{j=1}^{m} b_j \cdot \sigma(a_j \cdot x),$$

where $m \in \mathbb{N}$ is the size of the hidden layer and $\sigma : \mathbb{R} \to \mathbb{R}$ is an activation function and the parameters[8] are $(\theta_j)_{j=1}^m$ where $\theta_j = (a_j, b_j) \in \mathbb{R}^{d+1}$, so here the number of parameters is $p = m(d + 1)$. We have also introduced a scaling $\alpha(m) > 0$ as in Section 1.2.

[8] We have omitted the bias/intercept, which is recovered by fixing the last coordinate of x to 1.

Justification for asymptotics. In this paragraph, we justify the formula for the asymptotic upper bound on $\kappa_{h_m}(w_0)$ given for such models in Section 1.2. Using the assumption that $\mathbb{E}\phi(\theta_i) = 0$ and the fact that the parameters are independents, one has $\mathbb{E}\|h(w_0)\|^2 = m\alpha(m)^2\mathbb{E}\|\phi(\theta)\|^2$. For the differential, from the law of large numbers, we have the estimate

$$\frac{1}{m\alpha(m)^2} Dh(w_0)\, Dh(w_0)^\intercal = \frac{1}{m}\sum_{i=1}^{m} D\phi(\theta_i)D\phi(\theta_i)^\intercal \underset{m\to\infty}{\longrightarrow} \mathbb{E}\left[D\phi(\theta)D\phi(\theta)^\intercal\right].$$

It follows that $\mathbb{E}\|Dh(w_0)\|^2 = \mathbb{E}\|Dh(w_0)Dh(w_0)^\intercal\| \sim m\alpha(m)^2\|\mathbb{E}[D\phi(\theta)D\phi(\theta)^\intercal]\|$ because we have assumed that $D\phi$ is not identically 0 on the support of θ. One also has

$$\|D^2h(w_0)\| = \sup_{\substack{u\in\mathbb{R}^{d\times m}\\ \|u\|\leq 1}} \alpha(m)\sum_{i=1}^{m} u_i^\intercal D^2\phi(\theta_i)u_i \leq \alpha(m)\sup_{\theta_i}\|D^2\phi(\theta_i)\| \leq \alpha(m)\mathrm{Lip}(D\phi).$$

From the definition of $\kappa_{h_m}(w_0)$ and the upper bound $\|h_m(w_0)-y^\star\| \leq \|h(w_0)\|+\|y^\star\|$ we conclude that

$$\mathbb{E}[\kappa_{h_m}(w_0)] \lesssim m^{-\frac{1}{2}} + (m\alpha(m))^{-1}.$$

Limit kernels and random feature. In this section, we show that the tangent kernel is a *random feature kernel* for neural networks with a single hidden layer. For simplicity, we consider the scaling $\alpha(m) = 1/\sqrt{m}$ as in [11] which leads to a non-degenerated limit of the kernel[9] as $m \to \infty$. The associated tangent kernel in Eq. (7) is the sum of two kernels $K_m(x,x') = K_m^{(a)}(x,x') + K_m^{(b)}(x,x')$, one for each layer, where

$$K_m^{(a)}(x,x') = \frac{1}{m}\sum_{j=1}^{m}(x\cdot x')b_j^2\sigma'(a_j\cdot x)\sigma'(a_j\cdot x') \quad \text{and} \quad K_m^{(b)}(x,x') = \frac{1}{m}\sum_{j=1}^{m}\sigma(a_j\cdot x)\sigma(a_j\cdot x').$$

If we assume that the initial weights a_j (resp. b_j) are independent samples of a distribution on $\mathbb{R}^d$ (resp. a distribution on $\mathbb{R}$), these are random feature kernels [26] that converge as $m \to \infty$ to the kernels

$$K^{(a)}(x,x') = \mathbb{E}_{(a,b)}\left[(x\cdot x')b^2\sigma'(a\cdot x)\sigma'(a\cdot x')\right] \quad \text{and} \quad K^{(b)}(x,x') = \mathbb{E}_a\left[\sigma(a\cdot x)\sigma(a\cdot x')\right].$$

The second component $K^{(b)}$, corresponding to the differential with respect to the output layer, is the one traditionally used to make the link between these networks and random features [27]. When $\sigma(s) = \max\{s, 0\}$ is the rectified linear unit activation and the distribution of the weights a_j is rotation invariant in $\mathbb{R}^d$, one has the following explicit formulae [8]:

$$K^{(a)}(x,x') = \frac{(x\cdot x')\mathbb{E}(b^2)}{2\pi}(\pi-\varphi), \quad K^{(b)}(x,x') = \frac{\|x\|\|x'\|\mathbb{E}(\|a\|^2)}{2\pi d}((\pi-\varphi)\cos\varphi + \sin\varphi) \tag{8}$$

where $\varphi \in [0,\pi]$ is the angle between the two vectors x and x'. See Figure 4 for an illustration of this kernel and the convergence of its random approximations. The link with (independent) random sampling is lost for deeper neural networks, but it is shown in [17] that tangent kernels still converge when the size of networks increase, for certain architectures.

A.3 Generalization for the lazy model

As noted in the main text, in supervised machine learning, $\mathcal{F}$ is often a Hilbert space of functions on $\mathbb{R}^d$ and the model h is often of the form $h(w) = f(w,\cdot)$ where $f : \mathbb{R}^p \times \mathbb{R}^d \to \mathbb{R}^k$. A natural question that arises in this context and that is not directly answered by the theorems of Section 2, is whether the trained lazy model and the trained tangent model also generalize the same way, i.e. whether at training time T, it holds $f(w(T),x) \approx \bar{f}(\bar{w}(T),x)$ for points $x \in \mathbb{R}^d$ that are not in the training set, where $\bar{f}(w,x) = f(w_0,x) + D_w f(w_0,x)(w-w_0)$. We will see here that it is actually a simple consequence of the bounds.

[9] Since the definition of gradients depends on the choice of a metric, this scaling is not of intrinsic importance. Rather, it reflects that we work with the Euclidean metric on $\mathbb{R}^p$. The choice of scaling however becomes important when dealing with training (see also discussion in Section 1.2).

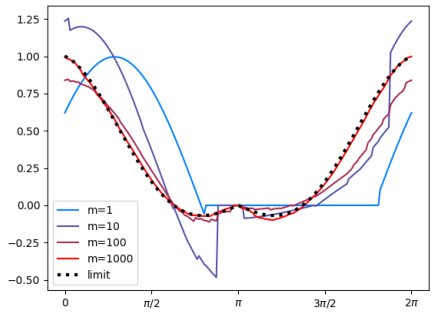


Figure 4: Random realizations of the kernels K_m and the limit kernel K of Eq. (8). We display the value of $K(x, x')$ as a function of $\varphi = \text{angle}(x, x')$ with x fixed, on a section of the sphere in $\mathbb{R}^{10}$. Parameters are normal random variables of variance 1, so $\mathbb{E}(b^2) = 1$ and $\mathbb{E}(\|a\|^2) = d$.

Proposition A.1 (Generalizing like the tangent model)**.** *Assume that for some $C > 0$ it holds $\|w_\alpha(T) - \bar{w}(T)\| \leq C\log(\alpha)/\alpha^2$. Assume moreover that there exists a set $\mathcal{X} \subset \mathbb{R}^d$ such that $M_1 := \sup_{x\in\mathcal{X}} \|D_w f(w_0, x)\| < \infty$ and $M_2 := \sup_{x\in\mathcal{X}} \text{Lip}(w \mapsto D_w f(w, x)) < \infty$. Then it holds*

$$\sup_{x\in\mathcal{X}} \|\alpha f(w_\alpha(T), x) - \alpha \bar{f}(\bar{w}_\alpha(T), x)\| \leq C\frac{\log\alpha}{\alpha}\left(M_1 + \frac{1}{2}C \cdot M_2 \cdot \log(\alpha)\right) \underset{\alpha\to\infty}{\longrightarrow} 0.$$

Proof. Let us call A the quantity to be upper bounded, and start with the decomposition

$$A \leq \sup_{x\in\mathcal{X}} \|\alpha f(w_\alpha(T), x) - \alpha \bar{f}(w_\alpha(T), x)\| + \sup_{x\in\mathcal{X}} \|\alpha \bar{f}(w_\alpha(T), x) - \alpha \bar{f}(\bar{w}_\alpha(T), x)\| = A_1 + A_2$$

By Taylor's theorem applied at each point $x \in X$, one has

$$A_1 \leq \frac{\alpha}{2} M_2 \|w_\alpha(T) - \bar{w}_\alpha(T)\|^2 \leq \frac{C^2 \cdot M_2 \log(\alpha)^2}{2\alpha}.$$

It also holds

$$A_2 = \alpha \sup_{x\in\mathcal{X}} \|D_w f(w_0, x)(w_\alpha(T) - \bar{w}_\alpha(T))\| \leq \frac{M_1 C \log(\alpha)}{\alpha}$$

and the conclusion follows. □

B Proofs of the theoretical results

In all the forthcoming proofs, we use the notations $y(t) = \alpha h(w_\alpha(t))$ and $\bar{y}(t) = \alpha \bar{h}(\bar{w}_\alpha(t))$ for the dynamics in $\mathcal{F}$ (they also depend on α although this is not reflected in the notation). We also write $\Sigma(w) := Dh(w)Dh(w)^\intercal$ for the so-called tangent kernel [17], which is a quadratic form on $\mathcal{F}$. By using the chain rule, we find that the trajectories in $\mathcal{F}$ solve the differential equation

$$\frac{d}{dt} y(t) = -\Sigma(w_\alpha(t))\nabla R(y(t)),$$
$$\frac{d}{dt} \bar{y}(t) = -\Sigma(w(0))\nabla R(\bar{y}(t)).$$

with $y(0) = \bar{y}(0) = \alpha h(w_0)$. Remark that the first differential equation is coupled with $w_\alpha(t)$.

B.1 Proof for Theorem 2.2 (finite horizon, non-quantitative)

For this first proof, we only track the dependency in α, and we use C to denote a quantity independent of α, that may vary from line to line. For $T > 0$, it holds

$$\int_0^T \|w'_\alpha(t)\| dt = \int_0^T \|\nabla F_\alpha(w_\alpha(t))\| dt \leq \sqrt{T}\left(\int_0^T \|\nabla F_\alpha(w_\alpha(t))\|^2 dt\right)^{\frac{1}{2}}.$$

It follows, by using the fact that $\frac{d}{dt}F_\alpha(w_\alpha(t)) = -\|\nabla F_\alpha(w_\alpha(t))\|^2$, that $\sup_{t\in[0,T]}\|w_\alpha(t) - w(0)\| \le (T\cdot F_\alpha(w_\alpha(t)))^{\frac{1}{2}} \lesssim \frac{1}{\alpha}$. In particular, we deduce that $\sup_{t\in[0,T]}\|y(t)-y(0)\| \le C$ and $\sup_{t\in[0,T]}\|\nabla R(y(t))\| \le C$.

Let us now consider the evolution of $\Delta(t) := \|y(t) - \bar{y}(t)\|$. It satisfies $\Delta(0) = 0$ and

$$\begin{aligned}\Delta'(t) &\le \|\Sigma(w_\alpha(t))\nabla R(y(t)) - \Sigma(w(0))\nabla R(\bar{y}(t))\| \\ &\le \|(\Sigma(w_\alpha(t)) - \Sigma(w(0)))\nabla R(y(t))\| + \|\Sigma(w(0))(\nabla R(y(t)) - \nabla R(\bar{y}(t))\| \\ &\le C_1/\alpha + C_2\Delta(t)\end{aligned}$$

The ordinary differential equation $u'(t) = C_1/\alpha + C_2 u(t)$ with initial condition $u(0) = 0$ admits the unique solution $u(t) = \frac{C_1}{\alpha C_2}(\exp(C_2 t) - 1)$. Since $\Delta(t)$ is a sub-solution of this system, it follows that $\Delta(t) \le \frac{C_1}{\alpha C_2}(\exp(C_2 t) - 1) \le C/\alpha$ (notice the exponential dependence in the final time and some other characteristics of the problem). Finally, consider the quantity $\delta(t) = \|w_\alpha(t) - \bar{w}_\alpha(t)\|$. It holds

$$\begin{aligned}\delta'(t) &\le \alpha^{-1}\|Dh(w_\alpha(t))^\intercal \nabla R(f(t)) - Dh(w_0)^\intercal \nabla R(\bar{y}(t))\| \\ &\le \alpha^{-1}\|Dh(w_\alpha(t))^\intercal - Dh(w_0)^\intercal\|\|\nabla R(y(t))\| + \alpha^{-1}\|Dh(w_0)\|\|\nabla R(y) - \nabla R(\bar{y}(t))\| \\ &\le C\alpha^{-2}\end{aligned}$$

We thus conclude, since $\delta(0) = 0$, that $\sup_{t\in[0,T]}\|\delta(t)\| \le \alpha^{-2}$.

B.2 Proof of Theorem 2.3 (finite horizon, square loss)

Step 1. With the square loss, the objective is still potentially non-convex, but we have the property

$$\frac{d}{dt}\|y(t) - y^*\|^2 = -\langle \Sigma(w(t))(y(t) - y^\star), y(t) - y^\star\rangle \le 0.$$

The proof scheme is otherwise similar as above, but we carry all constants. Let us denote $T_{exit} = \inf\{t > 0\ ;\ \|w_\alpha(t) - w_0\| > r\}$. For $t \le T_{exit}$ it holds

$$\|w_\alpha'(t)\| = \|\nabla F_\alpha(w_\alpha(t))\| \le \alpha^{-1}\|y(t) - y^\star\|\|Dh(w_\alpha(t))\| \le \alpha^{-1}\|y(0) - y^\star\|\mathrm{Lip}(h)$$

It follows that $\|w_\alpha(t) - w(0)\| \le t\alpha^{-1}\|y(0) - y^\star\|\mathrm{Lip}(h)$ (this bound is tighter for small times, compared to the bound in $\sqrt{t}$ used in the previous proof). Since we have assumed that $\alpha \ge k\|y(0) - y^\star\|/(r\mathrm{Lip}(h))$, it holds $\|w_\alpha(t) - w_0\| \le (t/K)\cdot r\mathrm{Lip}(h)^2 = r$ so $T_{exit} > T$.

Step 2. Now we consider $\Delta(t) = \|y(t) - \bar{y}(t)\|$. It holds

$$\begin{aligned}\frac{1}{2}\frac{d}{dt}\Delta(t)^2 &= \langle y'(t) - \bar{y}'(t), y(t) - \bar{y}(t)\rangle \\ &\le -\langle \Sigma(w_\alpha(t))\nabla R(y(t)) - \Sigma(w(0))\nabla R(\bar{y}(t)), y(t) - \bar{y}(t)\rangle \\ &\le -\langle (\Sigma(w_\alpha(t)) - \Sigma(w(0)))\nabla R(y(t)), y(t) - \bar{y}(t)\rangle\end{aligned}$$

where we have used the fact that $\langle \Sigma(w(0))(\nabla R(y(t)) - \nabla R(\bar{y}(t)), y(t) - \bar{y}(t)\rangle \ge 0$, which is specific to the square loss. Taking the norms and dividing both sides by $\Delta(t)$, it follows

$$\Delta'(t) \le \mathrm{Lip}(\Sigma)\cdot\|w_\alpha(t) - w(0)\|\|y(0) - y^\star\| \le 2\mathrm{Lip}(h)^2\mathrm{Lip}(Dh)t\alpha^{-1}\|y(0) - y^\star\|^2$$

where we have used $\mathrm{Lip}(\Sigma) \le 2\mathrm{Lip}(h)\mathrm{Lip}(Dh)$. Since $\Delta(0) = 0$, it follows

$$\Delta(t) \le \frac{t^2}{\alpha}\mathrm{Lip}(h)^2\mathrm{Lip}(Dh)\|y(0) - y^\star\|^2.$$

The bound in the statement then follows by writing this upper bound at time $T = K/\mathrm{Lip}(h)^2$.

Step 3. Finally, consider $\delta(t) = \|w_\alpha(t) - \bar{w}_\alpha(t)\|$. The bound that we will obtain is not reported in the main text due to space constraints, but proved here for the sake of completeness. As in the previous proof, it holds

$$\alpha\delta'(t) \le \|Dh(w_\alpha(t))^\intercal - Dh(w_0)^\intercal\|\|\nabla R(y(t))\| + \|Dh(w_0)\|\|\nabla R(y) - \nabla R(\bar{y}(t))\| = A(t) + B(t).$$

Let us bound these two quantities separately. On the one hand, it holds for $t \in [0, T]$,

$$A(t) \leq \mathrm{Lip}(Dh)\|w_\alpha(t) - w_0\|\|y(0) - y^\star\| \leq \frac{t}{\alpha}\mathrm{Lip}(h)\mathrm{Lip}(Dh)\|y(0) - y^\star\|^2.$$

On the other hand, it holds for $t \in [0, T]$,

$$B(t) \leq \frac{t^2}{\alpha}\mathrm{Lip}(h)^3\mathrm{Lip}(Dh)\|y(0) - y^\star\|^2.$$

By integrating these two bounds and summing, we get

$$\begin{aligned}\delta(T) &\leq \frac{T^2}{\alpha^2}\mathrm{Lip}(h)^2\mathrm{Lip}(Dh)\|y(0) - y^\star\|^2 \left(\frac{2}{\mathrm{Lip}(h)} + \frac{4T}{3}\mathrm{Lip}(h)\right) \\ &\leq \frac{K^2}{\alpha^2}\frac{\mathrm{Lip}(Dh)}{\mathrm{Lip}(h)^3}\|y(0) - y^\star\|^2 \left(2 + 4K/3\right).\end{aligned}$$

After rearranging the terms, we obtain

$$\frac{\alpha\mathrm{Lip}(h)}{\|y(0) - y^\star\|}\|w_\alpha(T) - \bar{w}_\alpha(T)\| \leq \frac{K^2}{\alpha}\frac{\mathrm{Lip}(Dh)}{\mathrm{Lip}(h)^2}\|y(0) - y^\star\| \left(2 + 4K/3\right)$$

Note that this bound is arranged so that both sides of the inequality are dimensionless, in the sense that they would not change under a simple rescaling of either the norm on $\mathcal{F}$ or on $\mathbb{R}^p$. The left-hand side should be understood as the relative difference between the non-linear and the linearized dynamics, while the right-hand side involves the *scale* of Section 1.2.

B.3 Proof of Theorem 2.4 (over-parameterized case)

Consider the radius $r_0 \coloneqq \sigma_{\min}/(2\mathrm{Lip}(Dh))$. By smoothness of h, it holds $\Sigma(w) \succeq \sigma_{\min}^2\mathrm{Id}/4$ as long as $\|w - w_0\| < r_0$. Thus Lemma B.1 below guarantees that $y(t)$ converges linearly, up to time $T \coloneqq \inf\{t \geq 0\ ;\ \|w_\alpha(t) - w_0\| > r_0\}$. It only remains to find conditions on α so that $T = +\infty$. The variation of the parameters $w_\alpha(t)$ can be bounded for $0 \leq t \leq T$ as

$$\|w_\alpha'(t)\| \leq \frac{1}{\alpha}\|Dh(w_\alpha(t))\|\|\nabla R(y(t))\| \leq \frac{2M}{\alpha}\|Dh(w_0)\|\|y(t) - y^*\|.$$

By Lemma B.1, it follows that for $0 \leq t \leq T$,

$$\begin{aligned}\|w_\alpha(t) - w_0\| &\leq \frac{2M^{3/2}}{\alpha m}\|Dh(w_0)\|\|y(0) - y^*\| \int_0^t e^{-(m\sigma_{\min}^2/4)s}ds \\ &\leq \frac{8\kappa^{3/2}}{\alpha\sigma_{\min}^2}\|Dh(w_0)\|\|y(0) - y^*\|.\end{aligned}$$

This quantity is smaller than r_0, and thus $T = \infty$, if $\|y(0) - y^*\| \leq 2\alpha C_0$. This is in particular guaranteed by the conditions on $h(w_0)$ and α in the theorem.

When $h(w_0) = 0$, the previous bound also implies the "laziness" property $\sup_{t\geq 0}\|w_\alpha(t) - w_0\| = O(1/\alpha)$ since in that case $y(0)$ does not depend on α. For the comparison with the tangent gradient flow, the first bound is obtained by applying the stability Lemma B.2, and noticing that the quantity denoted by K in that lemma is in $O(1/\alpha)$ thanks to the previous bound on $\|w_\alpha(t) - w_0\|$. For the last bound, we compute the integral over $[0, +\infty)$ of the bound

$$\begin{aligned}\alpha\|w_\alpha'(t) - \bar{w}_\alpha'(t)\| &= \|Dh(w_\alpha(t))^\intercal\nabla R(y(t)) - Dh(w_0)^\intercal\nabla R(\bar{y}(t))\| \\ &\leq \|Dh(w_\alpha(t)) - Dh(w_0)\|\|\nabla R(y(t))\| + \|Dh(w_0)\|\|\nabla R(y(t)) - \nabla R(\bar{y}(t))\|.\end{aligned}$$

It is easy to see from the derivations above that the integral of the first term is in $O(1/\alpha)$. For the second term, we define $t_0 \coloneqq 4\log\alpha/(\mu\sigma_{\min}^2)$ and on $[0, t_0]$ we use the smoothness bound

$$\|\nabla R(y(t)) - \nabla R(\bar{y}(t))\| \leq M\|y(t) - \bar{y}(t)\|$$

which integral over $[0, t_0]$ is in $O(\log\alpha/\alpha)$, while on $[t_0, +\infty)$ we use the crude bound

$$\|\nabla R(y(t)) - \nabla R(\bar{y}(t))\| \leq \|\nabla R(y(t))\| + \|\nabla R(\bar{y}(t))\|$$

which integral over $[t_0, +\infty)$ is in $O(1/\alpha)$ thanks to the definition of t_0 and the exponential decrease of ∇R along both trajectories. This is sufficient to conclude. As a side note, we remark that the assumption that Dh is globally Lipschitz could be avoided by considering the more technical definition

$$\mathrm{Lip}(Dh) \coloneqq \inf\left\{L > 0\ ;\ Dh \text{ is } L\text{-Lipschitz on a ball centered at } w_0 \text{ of radius } \frac{\sigma_{\min}}{2L}\right\} > 0,$$

because then the path $w_\alpha(t)$ never escapes the ball of radius $\frac{\sigma_{\min}}{2L}$ around w_0 for $\alpha > \|y^*\|/C_0$.

Lemma B.1 (Strongly-convex gradient flow in a time-dependent metric)**.** *Let $F : \mathcal{F} \to \mathbb{R}$ be a m-strongly-convex function with M-Lipschitz continuous gradient and with global minimizer y^* and let $\Sigma(t) : \mathcal{F} \to \mathcal{F}$ be a time dependent continuous self-adjoint linear operator with eigenvalues lower bounded by $\lambda > 0$ for $0 \le t \le T$. Then solutions on $[0, T]$ to the differential equation*

$$y'(t) = -\Sigma(t)\nabla F(y(t)),$$

satisfy, for $0 \le t \le T$,

$$\|y(t) - y^*\| \le (M/m)^{1/2}\|y(0) - y^*\| \exp(-m\lambda t).$$

Proof. By strong convexity, it holds $\bar{F}(y) \coloneqq F(y) - F(y^*) \le \frac{1}{2m}\|\nabla F(y)\|^2$. It follows

$$\frac{d}{dt}\bar{F}(y(t)) = -\nabla F(y(t))^\intercal\, \Sigma(t)\, \nabla F(y(t)) \le -\lambda\|\nabla F(y(t))\|^2 \le -2m\lambda\bar{F}(y),$$

and thus $\bar{F}(y(t)) \le \exp(-2m\lambda)\, \bar{F}(y(0))$ by Grönwall's Lemma. We now use the strong convexity inequality $\|y - y^*\|^2 \le \frac{2}{m}\bar{F}(y)$ in the left-hand side and the smoothness inequality $\bar{F}(y) \le \frac{1}{2}M\|y - y^*\|^2$ in the right-hand side. This yields $\|y(t) - y^*\|^2 \le \frac{M}{m}\exp(-2m\lambda)\,\|y(0) - y^*\|^2$. □

B.4 Stability Lemma

The following stability lemma is at the basis of the equivalence between lazy training and linearized model training in Theorem 2.4. We limit ourselves to a rough estimate sufficient for our purposes.

Lemma B.2. *Let $R : \mathcal{F} \to \mathbb{R}_+$ be a m-strongly convex function and let $\Sigma(t)$ be a time dependent positive definite operator on $\mathcal{F}$ such that $\Sigma(t) \succeq \lambda \mathrm{Id}$ for $t \ge 0$. Consider the paths $y(t)$ and $\bar{y}(t)$ on $\mathcal{F}$ that solve for $t \ge 0$,*

$$y'(t) = -\Sigma(t)\nabla R(y(t)) \qquad \textit{and} \qquad \bar{y}'(t) = -\Sigma(0)\nabla R(\bar{y}(t)).$$

Defining $K \coloneqq \sup_{t\ge 0}\|(\Sigma(t) - \Sigma(0))\nabla R(y(t))\|$, it holds for $t \ge 0$,

$$\|y(t) - \bar{y}(t)\| \le \frac{K\|\Sigma(0)\|^{1/2}}{\lambda^{3/2}m}.$$

Proof. Let $\Sigma_0^{1/2}$ be the positive definite square root of $\Sigma(0)$, let $z(t) = \Sigma_0^{-1/2}y(t)$, $\bar{z}(t) = \Sigma_0^{-1/2}\bar{y}(t)$ and let $h : \mathbb{R}_+ \to \mathbb{R}_+$ be the function defined as $h(t) = \frac{1}{2}\|z(t) - \bar{z}(t)\|^2$. It holds

$$\begin{aligned}
h'(t) &= \langle z'(t) - \bar{z}'(t), z(t) - \bar{z}(t)\rangle \\
&= -\langle \Sigma_0^{-1/2}\Sigma(t)\nabla R(\Sigma_0^{1/2}z(t)) - \Sigma_0^{1/2}\nabla R(\Sigma_0^{1/2}\bar{z}(t)), z(t) - \bar{z}(t)\rangle \\
&= -\langle \Sigma_0^{1/2}\nabla R(\Sigma_0^{1/2}z(t)) - \Sigma_0^{1/2}\nabla R(\Sigma_0^{1/2}\bar{z}(t)), z(t) - \bar{z}(t)\rangle && (A(t)) \\
&\quad - \langle \Sigma_0^{-1/2}(\Sigma(t) - \Sigma(0))\nabla R(\Sigma_0^{1/2}z(t)), z(t) - \bar{z}(t)\rangle. && (B(t))
\end{aligned}$$

Since the function $z \mapsto R(\Sigma_0^{1/2}z)$ is λm-strongly convex, one has that $A(t) \le -2\lambda m h(t)$. Using the quantity K introduced in the statement, one has also $\|B(t)\| \le K\|z(t) - \bar{z}(t)\|/\sqrt{\lambda} = K\sqrt{2h(t)/\lambda}$. Summing these two terms yields the bound

$$h'(t) \le K\sqrt{2h(t)/\lambda} - 2\lambda m h(t).$$

The right-hand side is a concave function of $h(t)$ which is nonnegative for $h(t) \in [0, K^2/(2\lambda^3 m^2)]$ and negative for higher values of $h(t)$. Since $h(0) = 0$ it follows that for all $t \ge 0$, one has $h(t) \le K^2/(2\lambda^3\mu^2)$ and the result follows since $\|y(t) - \bar{y}(t)\| \le \|\Sigma(0)\|^{1/2}\sqrt{2h(t)}$. □

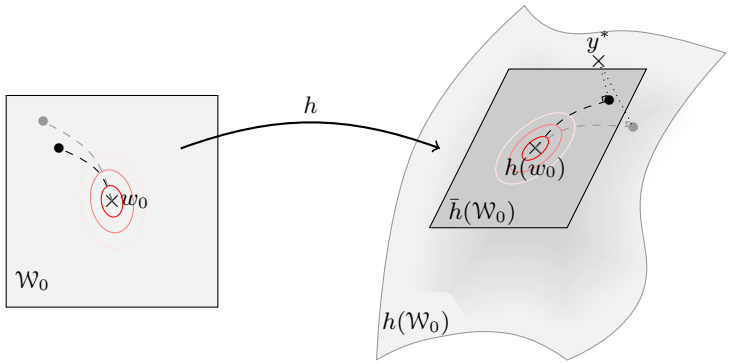


Figure 5: There is a small neighborhood $\mathcal{W}_0 \subset \mathbb{R}^p$ of the initialization w_0, which image by h is a differentiable manifold in $\mathcal{F}$. In the lazy regime, the optimization paths (both in $\mathcal{W}$ and in $\mathcal{F}$) for the non-linear model h (dashed gray paths) are close to those of the linearized model $\bar{h}$ (dashed black paths) until convergence or stopping time (Section 2). This figure illustrates the under-parameterized case where $p < \dim(\mathcal{F})$.

B.5 Proof of Theorem 2.5 (under-parameterized case)

The setting of this theorem is depicted on Figure 5. By the rank theorem (a result of differential geometry, see [21, Thm. 4.12] or [1] for a statement in separable Hilbert spaces), there exists open sets $\mathcal{W}_0, \bar{\mathcal{W}}_0 \subset \mathbb{R}^p$ and $\mathcal{F}_0, \bar{\mathcal{F}}_0 \subset \mathcal{F}$ and diffeomorphisms $\varphi : \mathcal{W}_0 \to \bar{\mathcal{W}}_0$ and $\psi : \mathcal{F}_0 \to \bar{\mathcal{F}}_0$ such that $\varphi(w_0) = 0$, $\psi(h(w_0)) = 0$ and $\psi \circ h \circ \varphi^{-1} = \pi_r$, where π_r is the map that writes, in suitable bases, $(x_1, \dots, x_p) \mapsto (x_1, \dots, x_r, 0, \dots)$. Up to restricting these domains, we may assume that $\bar{\mathcal{F}}_0$ is convex. We also denote by Π_r the r-dimensional hyperplan in $\mathcal{F}$ that is spanned by the first r vectors of the basis. The situation is is summarized in the following commutative diagram:

$$\begin{tikzcd}
\mathcal{W}_0 \arrow[r, "h"] \arrow[d, "\varphi"'] & \mathcal{F}_0 \arrow[d, "\psi"] \\
\bar{\mathcal{W}}_0 \arrow[r, "\pi_r"'] & \bar{\mathcal{F}}_0
\end{tikzcd}$$

In the rest of the proof, we denote by $C > 0$ any quantity that depends on m, M and Lipschitz smoothness constants of $h, \psi, \varphi, \psi^{-1}, \varphi^{-1}$, but not on α. Although we do not do so, this could be translated into explicit constants that depends on the smoothness of h and R, on the strong convexity constant of R and on the smallest positive singular value of $Dh(w_0)$ using quantitative versions of the rank theorem [5, Thm. 2.7].

Step 1. Our proof is along the same lines as that of Theorem 2.4, but performed in Π_r which can be thought of as a straighten up version of $h(\mathcal{W}_0)$. Consider the function G_α defined for $g \in \bar{\mathcal{F}}_0$ as $G_\alpha(g) = R(\alpha\psi^{-1}(g))/\alpha^2$. The gradient and Hessian of G_α satisfy, for $v_1, v_2 \in \mathbb{R}^p$,

$$\begin{aligned}
\nabla G_\alpha(g) &= \frac{1}{\alpha}(D\psi(g)^{-1})^\intercal \nabla R(\alpha\psi^{-1}(g)), \\
D^2 G_\alpha(g)(v_1, v_2) &= v_1^\intercal (D\psi(g)^{-1})^\intercal \nabla^2 R(\alpha\psi^{-1}(g)) D\psi(g)^{-1} v_2 \\
&\quad + \frac{1}{\alpha} D^2\psi(g)^{-1}(v_1, v_2)^\intercal \nabla R(\alpha\psi^{-1}(g)).
\end{aligned}$$

The second order derivative of G_α is the sum of a first term with eigenvalues in an interval $[C^{-1}, C]$, and a second term that goes to 0 as α increases. It follows that G_α is smooth and strongly convex for α large enough. Note that if R or ψ^{-1} are not twice continuously differentiable, then the Hessian computations should be understood in the distributional sense (this is sufficient because the functions involved are Lipschitz smooth). Also, let g^* be a minimizer of the lower-semicontinuous closure of

G_α on the closure of $\bar{\mathcal{F}}_0$. By strong convexity of R and our assumptions, it holds

$$\|g^*\|^2 \leq \frac{2}{m}(G_\alpha(0) - G_\alpha(g^*)) \leq \frac{2R(0)}{\alpha^2 m},$$

so g^* is in the interior of $\bar{\mathcal{F}}_0$ for α large enough and is then the unique minimizer of G_α.

Step 2. Now consider $T := \inf\{t \geq 0\ ;\ w_\alpha(t) \notin \mathcal{W}_0\}$. For $t \in [0, T)$, the trajectory $w_\alpha(t)$ of the gradient flow (4) has "mirror" trajectories in the four spaces in the diagram above. Let us look more particularly at $g(t) := \pi_r \circ \varphi(w_\alpha(t)) = \psi \circ h(w_\alpha(t))$ for $t < T$. In the following computation, we write $D\varphi$ for the value of the differential at the corresponding point of the dynamic $D\varphi(w_\alpha(t))$ (and similarly for other differentials). By noticing that $Dh = D\psi^{-1} D\pi_r D\varphi$, we have

$$\begin{aligned} g'(t) &= -\frac{1}{\alpha} D\psi Dh Dh^\intercal \nabla R(\alpha\psi^{-1}(g(t)) \\ &= -\frac{1}{\alpha} D\pi_r D\varphi D\varphi^\intercal D{\pi_r}^\intercal (D\psi^{-1})^\intercal \nabla R(\alpha\psi^{-1}(g(t)). \end{aligned}$$

so $g(t)$ remains in Π_r. Also, the first $r \times r$ block of $D\pi_r D\varphi D\varphi^\intercal D{\pi_r}^\intercal$ is positive definite on Π_r, with a positive lower bound (up to taking $\mathcal{W}_0$ and $\mathcal{F}_0$ smaller if necessary). Thus by Lemma B.1, there are constants $C_1, C_2 > 0$ independent of α such that, for $t \in [0, T)$, $\|g(t) - g^*\| \leq C_1 \|g(0) - g^*\| \exp(-C_2 t)$.

Step 3. Now we want to show that $T = +\infty$ for α large enough. It holds

$$w'(t) = -\frac{1}{\alpha} Dh^\intercal \nabla R(\alpha h(w_\alpha(t)) = D\varphi^\intercal D{\pi_r}^\intercal \nabla G_\alpha(g(t))$$

and, by Lipschitz-smoothness of G_α (Step 1), $\|\nabla G_\alpha(g(t))\| \leq \frac{C}{\alpha}\|g(t) - g^*\|$ hence

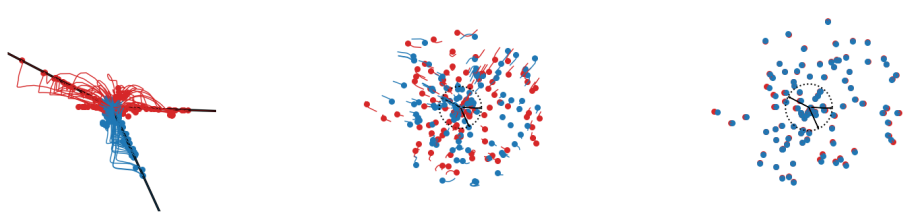
$$\|w_\alpha(t) - w_0\| \leq \frac{C}{\alpha}\int_0^t \exp(-C_2 s) ds \leq \frac{C}{\alpha C_2}.$$

Thus, by choosing α large enough, one has $w_\alpha(t) \in \mathcal{W}_0$ for all $t \geq 0$, so $T = \infty$ and the theorem follows.

C Experimental details and additional results

C.1 Many neurons dynamics visualized

The setting of Figure 6 is the same as for panels (a)-(b) in Figure 1 except that $m = 200, n = 200$: it allows to visualize behavior of the training dynamics for a larger number of neurons. Symmetrized initialization to set $f(w_0, \cdot) = 0$ was used on panel (c) but not on panel (b), where we see that the neurons need to move slightly more in order to compensate for the non-zero initialization. As on Figure 1, we observe a good behavior in the non-lazy regime for small τ.



(a) Non-lazy training ($\tau = 0.1$) (b) Lazy ($\tau = 2$, not symmetrized) (c) Lazy ($\tau = 2$, symmetrized)

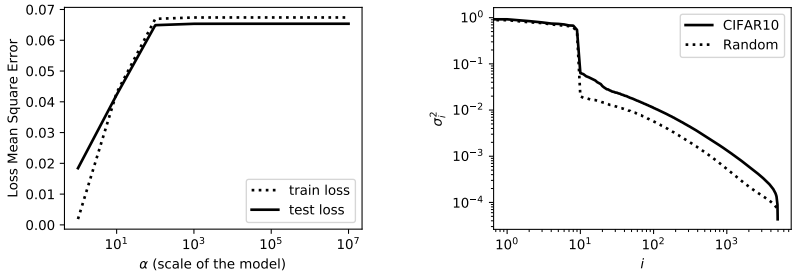
Figure 6: Training a two-layer ReLU neural network initialized with normal random weights of variance τ^2, as in Figure 1, but with more neurons. In this 2-homogeneous setting, changing τ^2 is equivalent to changing α by the same amount so lazy training occurs for large τ.

C.2 Stability of activations

We define here the "stability of activations" mentioned in Section 3.2. We consider a ReLU layer ℓ of size n_ℓ in a neural network and the test input data $(x_i)_{i=1}^N$ (the test images of CIFAR10 in our case). We call $z_{ij}(T) \in \mathbb{R}$ the value of the pre-activation (i.e. the value that goes through the ReLU function as an input) of index j on the data sample i, obtained with the parameters of the network at epoch T. The "stability of activations" for this layer is defined as $s_\ell := \frac{Q}{n_\ell \times N}$ where L is the number of ReLU layers, Q is the number of indices (i, j) that satisfy $\mathrm{sign}(z_{ij}(T_{last})) = \mathrm{sign}(z_{ij}(T_{init}))$ for $i \in \{1, \dots, B\}$ and $j \in \{1, \dots, n_\ell\}$, where T_{init} refers to initialization and T_{last} to the end of training. The quantity that we report on Figure 3(a) is the average of s_ℓ over all ReLU layers of the VGG-11 network, for various values of α.

C.3 Spectrum of the tangent kernel

In the setting of Figure 3(a), we want to understand why the linearized model (that is, trained for large α) could not reach low training accuracies in spite of being highly over-parameterized. Figure 7(a) shows the train and test losses after 70 epochs where we see that the training loss is far from 0 for all $\alpha \geq 10$. We report on Figure 7(b) the normalized and sorted eigenvalues σ_i^2 of the tangent kernel $Dh(w_0)Dh(w_0)^\intercal$ (notice the log-log scale) evaluated for two distinct input data sets $(x_i)_{i=1}^n$ of size $n = 500$: (i) images randomly sampled from the training set of CIFAR10 and (ii) images with uniform random pixel values. Since there are 10 output channels, the corresponding space $\mathcal{F}$ has 10×500 dimensions. We observe that there is a gap of 1 order of magnitude between the 0.2% largest eigenvalues and the remaining ones—which causes the ill conditionning—and then a decrease of order approximately $O(1/i)$. We observe a similar pattern with the CIFAR10 inputs and completely random inputs, which suggests that this conditioning is intrinsic to the linearized VGG model. Note that modifying the neural network architecture to improve this conditioning, or using optimization methods that are better adapted to ill-conditionned models, is beyond the scope of the present paper.



(a) (b)

Figure 7: (a) End-of training train and test loss. (b) Spectrum of the tangent kernel $Dh(w_0)Dh(w_0)^\intercal$ for the VGG11 model on two data sets.