Contents

1 Introduction

This thesis is about...

2 Neural Networks

This section introduces the basic concepts of neural networks and provides the necessary notation for further sections. We start with the formal definition of neural networks and explain the process of training afterwards. Finally we will introduce several training methods.

2.1 Notation

To start things off, we consider the simple model of a neuron.

Definition 2.1. *Let $d \in \mathbb{N}$, $w \in \mathbb{R}^{d+1}$ and $\sigma : \mathbb{R} \to \mathbb{R}$. A neuron is defined as*

$$\nu : \mathbb{R}^d \to \mathbb{R} : x \mapsto \sigma\Big(\sum_{i=1}^{d} w_i x_i + w_{d+1}\Big).$$

In this notion σ is referred to as activation function and the entries of w as weights.

As a generalization of this concept, one can use multiple neurons in parallel to form a layer.

Definition 2.2. *Let $d, k \in \mathbb{N}$, $w_1, \dots, w_k \in \mathbb{R}^{d+1}$, $\sigma_1, \dots \sigma_k \in \{\sigma : \mathbb{R} \to \mathbb{R}\}$ and $\nu_1, \dots, \nu_k$ be neurons with weights w_i and activation function σ_i for $i = 1, \dots, k$. A layer is defined as*

$$f : \mathbb{R}^d \to \mathbb{R}^k : x \mapsto \begin{bmatrix} \nu_1(x) \\ \vdots \\ \nu_k(x) \end{bmatrix}.$$

To provide a better understanding, these basic concepts can be visualized as graphs.

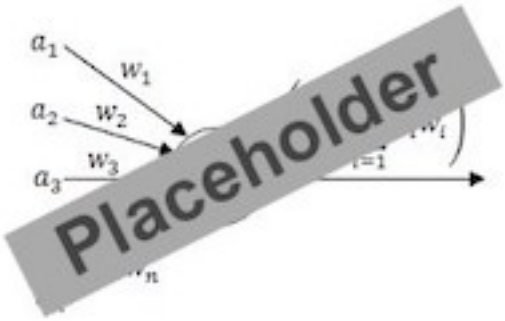

Figure 1: Illustration of a neuron as graph.

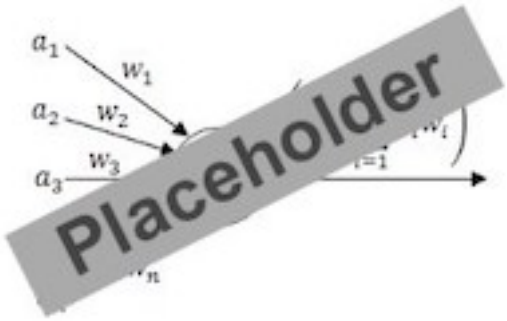

Figure 2: Illustration of a layer as graph.

Logically one can use the output of a layer as input for another layer. Iteratively doing so will construct specific functions known as neural networks. In order to formalize this idea we denote the following definition.

Definition 2.3. *Let $l \in \mathbb{N}$ and $d_0, \dots, d_l \in \mathbb{N}$. A neural network is defined as a function*

$$f : \mathbb{R}^{d_0} \to \mathbb{R}^{d_l} : x \mapsto f_l \circ f_{l-1} \circ \cdots \circ f_1(x),$$

where each f_i for $i = 1, \dots, l$ is a layer with input dimension d_{i-1} and output dimension d_i. Networks with a certain level of complexity, say $l > 2$, are referred to as deep neural networks.

Just like neurons and layers, neural networks can also be visualized as graphs.

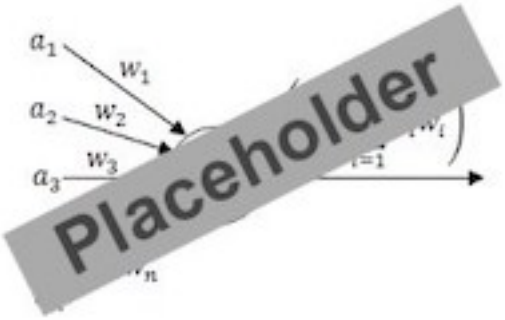

Figure 3: Illustration of a neural network as graph.

Neural networks find massive adoption in machine learning as they can model arbitrary complex functions. For example one could envision functions that predict the weather at a specific place and time or functions that determine if an image contains a cat or a dog. Clearly there is no simple mathematical formula to describe such problems.

A common practice to learn those functions is, to fix the number of layers, the number of neurons per layer as well as the activation functions, to choose some initial weights and then iteratively adjust the weights in order to approximate the unknown target function. This approach requires some mapping between a neural network and its weights. In the notion of definition 2.3 one can determine the total number of weights in f as

$$n = \sum_{i=0}^{l-1} (d_i + 1) \cdot d_{i+1}.$$

Denoting these weights as a parameter vector $w \in \mathbb{R}^n$ enables us to define a function

$$\tilde{f} : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}^p : (x, w) \mapsto \tilde{f}(x, w),$$

such that for fixed $w_1, w_2 \in \mathbb{R}^n$ the functions $\tilde{f}(x, w_1)$ and $\tilde{f}(x, w_2)$ are neural networks matching in their number of layers, number of neurons as well as activation functions and differing only in their choice of weights. One calls $\tilde{f}(x, w_1)$ and $\tilde{f}(x, w_2)$ realizations of the neural network architecture $\tilde{f}$.

In the following, the terms neural network and neural network architecture will be used synonymously if the meaning is clear by the context. Unless otherwise specified, $d \in \mathbb{N}$ will denote the dimension of the input space, $p \in \mathbb{N}$ will denote the dimension of the output space and $n \in \mathbb{N}$ will denote the number of adjustable weights.

2.2 Training

Training a neural network refers to finding the optimal weights, given a fixed neural network architecture $\tilde{f} : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}^p$. In other words, finding a function $f \in \tilde{\mathcal{F}}$, where

$$\tilde{\mathcal{F}} = \Big\{ f : \mathbb{R}^d \to \mathbb{R}^p : x \mapsto \tilde{f}(x, w) \mid w \in \mathbb{R}^n \Big\}$$

denotes the set of all neural networks with given architecture. In order to define a sense of optimality, we need a mechanism measuring the quality of network outputs. This is done by a loss function

$$\mathcal{L} : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R} : \big(f(x, w), y\big) \mapsto \mathcal{L}\big(f(x, w), y\big),$$

that should be chosen in such a way, that for some input $x \in \mathbb{R}^d$ with target output $y \in \mathbb{R}^p$ it associates some cost with the error between the prediction $f(x, w)$ and the true label y.

Under the assumption, that in reality there exists a probability distribution $\mathcal{D}$ on $\mathbb{R}^d \times \mathbb{R}^p$, describing the correlation between inputs $x \in \mathbb{R}^d$ and target values $y \in \mathbb{R}^p$, we call a weight vector $w_* \in \mathbb{R}^n$ to be optimal, if

$$\forall w \in \mathbb{R}^n : R(w_*) \leq R(w),$$

where $R(w)$ denotes the so called risk or generalization error

$$R : \mathbb{R}^n \to \mathbb{R} : w \mapsto \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \Big[\mathcal{L}\big(f(x, w), y\big) \Big].$$

For later use we will denote the marginal distribution of the inputs as $\mathcal{D}_x$. At this point it should be noted, that in general the underlying probability distribution $\mathcal{D}$ is unknown. Hence we can not evaluate the expectation in previous expression.

For this reason, the training of neural networks requires labeled training data

$$\Big\{ (x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}^p \mid i = 1, \ldots, m \Big\}$$

for some $m \in \mathbb{N}$. Instead of working with the generalization error, the training data enables us to minimize the empirical error function

$$E : \mathbb{R}^n \to \mathbb{R} : w \mapsto \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\big(f(x_i, w), y_i\big) + \lambda \|w\|^2,$$

where $\lambda \in \mathbb{R}$ is some regularization parameter used to control the complexity of w.

Intuitively minimization of the empirical error should lead to a minimization of the generalization error as well. Currently, a lot of research is being done on how these measures relate to each other. Nevertheless this thesis will focus on how to minimize the empirical error given labeled training data.

The problem of finding some weight vector w that minimizes the empirical error function is usually hard to solve as the error may have highly nonlinear dependencies on the weights. This is why in practice one often aims to compare several local minima and to settle for one that is sufficient enough, regardless of whether it is a global minimum or not.

Definition 2.4. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable. The gradient of f is defined as*

$$\nabla f : \mathbb{R}^n \to \mathbb{R}^n : x = (x_1, \dots, x_n) \mapsto \left[\frac{\partial}{\partial x_1} f(x), \cdots, \frac{\partial}{\partial x_n} f(x)\right]^T.$$

Due to the complexity of the error function, in general there is no easy way to find an analytical solution to the problem $\nabla E(w) = 0$. Hence most of the techniques for error minimization are based on iterative approximation. One popular method is gradient descent.

Algorithm 1 Gradient Descent

Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable. Gradient descent proceeds as follows:

1) Initialize $k = 0$, choose $\epsilon \geq 0 \in \mathbb{R}$ and pick some random $x_0 \in \mathbb{R}^n$.

2) **while** $\left\|\nabla f(x_k)\right\|_2 > \epsilon$ **do**:

3) Choose a step size $\alpha_k > 0 \in \mathbb{R}$.

4) Set $x_{k+1} = x_k - \alpha_k \cdot \nabla f(x_k)$ and $k = k + 1$.

5) **end while**

We will see, that in the case where f is bounded from below and ∇f is Lipschitz continuous, we can guarantee the converges of gradient descent to a stationary point.

Definition 2.5. *Let $(\mathcal{X}, d_\mathcal{X})$ and $(\mathcal{Y}, d_\mathcal{Y})$ be two metric spaces. A function $f : \mathcal{X} \to \mathcal{Y}$ is called Lipschitz continuous, if there exists $L \geq 0$, such that*

$$\forall x, x' \in \mathcal{X} : d_\mathcal{Y}\big(f(x), f(x')\big) \leq L \cdot d_\mathcal{X}(x, x').$$

The smallest constant L, that suffices the previous condition is called Lipschitz constant of f.

Casually speaking, the Lipschitz continuity of ∇f ensures that the gradient does not change heavily for two points that are close to each other.

Definition 2.6. *Let $f : \mathbb{R}^n \to \mathbb{R}$. The set of stationary points of f is defined as*

$$\big\{x \in \mathbb{R}^n \mid \nabla f(x) = 0\big\}.$$

There a three kinds of stationary points: saddle points, local extrema and global extrema. We are especially interested in global minima as they provide the lowest possible error.

Definition 2.7. *Let $\mathcal{X}$ be a subset of some real vector space and $f : \mathcal{X} \to \mathbb{R}$ be bounded from below. We call $x_* \in \mathcal{X}$ a global minimizer of f, if*

$$\forall x \in \mathcal{X} : f(x_*) \leq f(x).$$

To prove convergence of the gradient descent algorithm for bounded f with Lipschitz continuous gradient, we need an intermediate result, that is formulated in the following lemma.

Lemma 2.8. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable, such that the gradient ∇f is Lipschitz continuous with Lipschitz constant $L > 0$, then*

$$\forall x, y \in \mathbb{R}^n : f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|_2^2.$$

Proof. Let $x, y \in \mathbb{R}^n$, define $z_\lambda := x + \lambda(y - x)$ for $\lambda \in \mathbb{R}$ and consider the function

$$\phi : \mathbb{R}^n \mapsto \mathbb{R} : \lambda \mapsto f(z_\lambda) = f\big(x + \lambda(y - x)\big).$$

Since f is differentiable we can apply the chain rule to derive

$$\phi' : \mathbb{R}^n \mapsto \mathbb{R} : \lambda \mapsto (y - x)^T \cdot \nabla f\big(x + \lambda(y - x)\big).$$

Integration over ϕ' from $\lambda = 0$ to $\lambda = 1$ yields

$$\int_0^1 \langle \nabla f(z_\lambda), y - x \rangle \, d\lambda = \int_0^1 \phi'(\lambda) \, d\lambda = \phi(1) - \phi(0) = f(y) - f(x).$$

This equality can be rewritten as

$$\begin{aligned} f(y) - f(x) &= \int_0^1 \langle \nabla f(z_\lambda), y - x \rangle \, d\lambda \\ &= \int_0^1 \langle \nabla f(z_\lambda) - \nabla f(x) + \nabla f(x), y - x \rangle \, d\lambda \\ &= \langle \nabla f(x), y - x \rangle + \int_0^1 \langle \nabla f(z_\lambda) - \nabla f(x), y - x \rangle \, d\lambda. \end{aligned}$$

Using the Cauchy-Schwarz inequality we derive

$$\begin{aligned} \Big| \, f(y) - f(x) - \langle \nabla f(x), y - x \rangle \, \Big| &= \Big| \int_0^1 \langle \nabla f(z_\lambda) - \nabla f(x), y - x \rangle \, d\lambda \, \Big| \\ &\leq \int_0^1 \Big| \, \langle \nabla f(z_\lambda) - \nabla f(x), y - x \rangle \, \Big| \, d\lambda \\ &\leq \int_0^1 \|\nabla f(z_\lambda) - \nabla f(x)\|_2 \cdot \|y - x\|_2 \, d\lambda. \end{aligned}$$

Due to the Lipschitz continuty of ∇f we can use the upper bound

$$\|\nabla f(z_\lambda) - \nabla f(x)\|_2 \leq L \cdot \|z_\lambda - x\|_2 = L \cdot \|\lambda(y - x)\|_2 = L\lambda \cdot \|y - x\|_2$$

for $\lambda \in [0, 1]$, to conclude

$$\Big| \, f(y) - f(x) - \langle \nabla f(x), y - x \rangle \, \Big| \leq \int_0^1 L\lambda \cdot \|y - x\|_2^2 \, d\lambda = \frac{L}{2} \|y - x\|_2^2.$$

Rearranging the inequality finally yields

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|_2^2.$$

This completes the proof, since x, y were chosen arbitrary. □

Theorem 2.9. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable with a global minimizer $x_* \in \mathbb{R}^n$, such that the gradient ∇f is Lipschitz continuous with Lipschitz constant $L > 0$, then gradient descent with $\epsilon = 0$ and constant step size $\alpha_k = 1/L$ produces a sequence $(x_k)_{k \in \mathbb{N}_0} \in \mathbb{R}^n$, such that*

$$\forall n \in \mathbb{N} : \min_{0 \leq k \leq n} \big\|\nabla f(x_k)\big\|_2^2 \leq \frac{2L\big(f(x_0) - f(x_*)\big)}{n+1}.$$

Proof. Let $k \in \mathbb{N}$ be arbitrary. Since f is differentiable with Lipschitz continuous gradient, we can apply lemma 2.8 to derive

$$f(x_{k+1}) \leq f(x_k) + \big\langle \nabla f(x_k), x_{k+1} - x_k \big\rangle + \frac{L}{2}\big\|x_{k+1} - x_k\big\|_2^2.$$

By the definition of gradient descent, we can plug in $x_{k+1} - x_k = -1/L \cdot \nabla f(x_k)$ to conclude

$$\begin{aligned} f(x_{k+1}) &\leq f(x_k) + \big\langle \nabla f(x_k), -\frac{1}{L}\nabla f(x_k) \big\rangle + \frac{L}{2}\big\|\frac{1}{L}\nabla f(x_k)\big\|_2^2 \\ &= f(x_k) - \frac{1}{L}\big\|\nabla f(x_k)\big\|_2^2 + \frac{1}{2L}\big\|\nabla f(x_k)\big\|_2^2 \\ &= f(x_k) - \frac{1}{2L}\big\|\nabla f(x_k)\big\|_2^2. \end{aligned}$$

Hence the gradient descent algorithm with constant step size $\alpha_k = 1/L$ guarantees to make progress unless $\nabla f(x_k) \neq 0$. The previous expression is equivalent to

$$\big\|\nabla f(x_k)\big\|_2^2 \leq 2L\big(f(x_k) - f(x_{k+1})\big).$$

Summing up both sides from $k = 0$ to some $n \in \mathbb{N}$ and taking the average results in

$$\frac{1}{n+1}\sum_{k=0}^{n} \big\|\nabla f(x_k)\big\|_2^2 \leq \frac{2L}{n+1}\sum_{k=0}^{n} f(x_k) - f(x_{k+1}) = \frac{2L}{n+1}\big(f(x_0) - f(x_{n+1})\big).$$

Using $f(x_*) \leq f(x_k)$ for every $k \in \mathbb{N}$, we conclude

$$\min_{0 \leq k \leq n} \big\|\nabla f(x_k)\big\|_2^2 \leq \frac{1}{n+1}\sum_{k=0}^{n} \big\|\nabla f(x_k)\big\|_2^2 \leq \frac{2L}{n+1}\big(f(x_0) - f(x_*)\big).$$

This completes the proof, since n was chosen arbitrary. □

Note that in practice it is inefficient to use a constant step size of $1/L$. Nevertheless it is useful for theoretical analysis to guarantee convergence.

Theorem 2.9 states, that in the limit as $n \to \infty$, gradient descent converges to a stationary point $\hat{x} \in \mathbb{R}^n$ with $\nabla f(\hat{x}) = 0$. Since we have seen in the proof, that $f(x_k)$ is monotonically decreasing in k, gradient descent converges to the next stationary point in direction of descent, which is either a local minimum, a global minimum or a saddle point.

Next we will define a special class of functions having the nice property, that every stationary point is a global minimum and therefore gradient descent is an excellent tool for minimization.

Definition 2.10. *Let $\mathcal{X}$ be a subset of some real vector space. $\mathcal{X}$ is said to be convex, if*

$$\forall x, x' \in \mathcal{X} : \forall \lambda \in [0,1] : \lambda x + (1-\lambda)x' \in \mathcal{X}.$$

Casually speaking a set is convex if it contains the connection line between any two points in the set. Based on the notion of convex sets we can define convex functions.

Definition 2.11. *Let $\mathcal{X}$ be a convex subset of some real vector space. A function $f : \mathcal{X} \to \mathbb{R}$ is said to be convex, if*

$$\forall x, x' \in \mathcal{X} : \forall \lambda \in [0,1] : f\big(\lambda x + (1-\lambda)x'\big) \leq \lambda f(x) + (1-\lambda) f(x').$$

For later use we denote the following property of convex functions.

Lemma 2.12. *Let $\mathcal{X}$ be a convex subset of some real vector space. For convex functions $f : \mathcal{X} \to \mathbb{R}$ and $g : \mathcal{X} \to \mathbb{R}$ the sum $f + g$ is convex as well.*

Proof. Let $x, x' \in \mathcal{X}$ and $\lambda \in [0,1]$ be arbitrary, then

$$\begin{aligned}\big(f+g\big)(\lambda x + (1-\lambda)x') &= f(\lambda x + (1-\lambda)x') + g(\lambda x + (1-\lambda)x') \\ &\leq \lambda f(x) + (1-\lambda) f(x') + \lambda g(x) + (1-\lambda) g(x') \\ &= \lambda\big(f+g\big)(x) + (1-\lambda)\big(f+g\big)(x').\end{aligned}$$

This shows the convexity of $f + g$. □

Functions that are convex and continuous differentiable have the nice property, that every stationary point is a global minimum.

Lemma 2.13. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex and continuous differentiable and $x \in \mathbb{R}^n$, then*

$$\nabla f(x) = 0 \;\Leftrightarrow\; \forall x' \in \mathbb{R}^n : f(x) \leq f(x').$$

Proof. Let $x \in \mathbb{R}^n$ with $\nabla f(x) = 0$. For arbitrary $x' \in \mathbb{R}^n$ we define the function

$$\phi : \mathbb{R} \to \mathbb{R} : \lambda \mapsto f(x) + \lambda\big(f(x') - f(x)\big) - f\big(x + \lambda(x' - x)\big),$$

which is continuous differentiable with derivative

$$\phi' : \mathbb{R} \to \mathbb{R} : \lambda \mapsto f(x') - f(x) - \nabla f\big(x + \lambda(x' - x)\big)^T (x' - x).$$

Since ϕ is non-negative on the interval $[0,1]$ with $\phi(0) = 0$, we derive

$$\phi'(0) = f(x') - f(x) - \nabla f(x)^T (x' - x) \geq 0.$$

Using $\nabla f(x) = 0$ we conclude

$$\forall x' \in \mathbb{R}^n : f(x) \leq f(x').$$

This completes the proof, since the backwards direction is trivial. □

Thus for lower bounded functions $f : \mathbb{R}^n \to \mathbb{R}$, that are convex and differentiable with Lipschitz continuous gradient ∇f, we can guarantee the convergence of gradient descent to a global minimum due to theorem 2.9. Unfortunately this does not apply to the error function used for neural networks, but we will introduce a theoretical approach to prevent this problem in section 3.

Recalling the definition of the empirical error function

$$E : \mathbb{R}^n \to \mathbb{R} : w \mapsto \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\big(f(x_i, w), y_i\big) + \lambda \|w\|^2,$$

we realize, that the computation effort needed to compute the gradients, depends on the number of training samples m. This is a severe problem for the gradient descent algorithm, since we have to train on a significant amount of data, to optimize the millions of parameters in deep neural networks. Thus, gradient descent on the empirical error function is computationally and time expensive, that even with heavy computing power it can be unfeasible to find good solutions. However, there exist several other optimization methods, that apply the same concept of minimizing a function along the slope of its surface.

Stochastic gradient descent is a variation of gradient descent, that aims to approximate the gradient of the objective function instead of computing the exact gradient. Doing so, stochastic gradient descent makes a trade-off between the accuracy and the time needed for gradient computation. The basic idea is to compute the gradient on a random fraction of the data set and use this as an estimation of the gradient on the whole data set. In each iteration $k \in \mathbb{N}$ one randomly draws a subset $I_k \subset \{1, \dots, m\}$ of size $|I_k| = b \in \mathbb{N}$ and uses

$$\nabla E(w_k, I_k) := \frac{1}{b} \sum_{i \in I_k} \mathcal{L}\big(f(x_i, w_k), y_i\big) + \lambda \|w_k\|^2$$

as an approximation of $\nabla E(w_k)$. Although only a small amount of data is used for one single parameter update, during training most of the data will be used to fit the model due to the large number of iterative updates.

Algorithm 2 Stochastic Gradient Descent (SGD)

Let $E : \mathbb{R}^n \to \mathbb{R}$ be a differentiable empirical error function. SGD proceeds as follows:

1) Choose a batch size $b \in \mathbb{N}$, a number of epochs $K \in \mathbb{N}$ and pick some random $w_0 \in \mathbb{R}^n$.

2) **for** $k = 1, \ldots, \frac{Km}{b}$ **do**:

3) Draw a random subset $I_k \subset \{1, \ldots, m\}$ of size $|I_k| = b$.

4) Choose a step size $\alpha_k > 0 \in \mathbb{R}$.

5) Set $w_{k+1} = w_k - \alpha_k \cdot \nabla E(w_k, I_k)$.

6) **end for**

In here, an epoch describes one iteration on a number of samples that equals the size of the whole training dataset. A theoretical convergence analysis of stochastic gradient descent can be found in [?], but would exceed the scope of this theory.

Until now we have not discussed how to choose the step sizes α_k. To simplify things, in practice one usually chooses a step size $\alpha > 0$, that stays constant across each iteration. Unfortunately a constant step size does not adapt to the specific local properties of the error surface at iteration $k \in \mathbb{N}$. One method that aims to improve the SGD algorithm by including past gradients for the current parameter update is the following.

Algorithm 3 Adaptive Moment Estimation (Adam)

1: Requires differentiable empirical error function $E : \mathbb{R}^n \to \mathbb{R}$ and random $w_0 \in \mathbb{R}^n$.

2: Requires constant step size $\alpha > 0 \in \mathbb{R}$, batch size $b \in \mathbb{N}$ and number of epochs $K \in \mathbb{N}$.

3: Requires momentum factors $\beta_1, \beta_2 \in [0, 1)$ and a scalar $\epsilon > 0 \in \mathbb{R}$.

4: **for** $k = 1, \ldots, \frac{Km}{b}$ **do**:

5: Draw a random subset $I_k \subset \{1, \ldots, m\}$ of size $|I_k| = b$.

6: Set $m_k = \beta_1 m_{k-1} + (1 - \beta_1) \cdot \nabla E(w_k, I_k)$ and $\hat{m}_k = \frac{m_k}{1-\beta_1^k}$.

7: Set $v_k = \beta_2 v_{k-1} + (s - \beta_2) \cdot \nabla E(w_k, I_k)^2$ and $\hat{v}_k = \frac{v_k}{1-\beta_2^k}$.

8: Set $w_{k+1} = w_k - \alpha \cdot \frac{\hat{m}_k}{\sqrt{\hat{v}_k} + \epsilon}$.

9: **end for**

In here, all operations on vectors are performed element-wise. The scalar ϵ is used to prevent division by zero. The default setup proposed by the authors is to choose $\epsilon = 10^{-8}$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We will always apply this default setting when working with Adam.

The three optimization methods introduced so far, all have in common, that they perform parameter updates on first order information only. This is due to the fact, that Hessian matrix computation for deep neural networks is infeasible due to the large number of parameters. However, later on we will see one can train neural networks in tiny subspaces, such that second order methods are applicable. One optimization algorithm, that includes second order information is the following.

Algorithm 4 Broyden-Fletcher-Goldfarb-Shanno (BFGS)

1: Requires differentiable $f : \mathbb{R}^n \to \mathbb{R}$, random $x_0 \in \mathbb{R}^n$ and $\epsilon \geq 0 \in \mathbb{R}$.

2: Initialize $k \leftarrow 0$, $B_0 \leftarrow I_n$.

3: **while** $\|\nabla f(x_k)\|_2 > \epsilon$ **do**:

4: Choose a step size $\alpha_k > 0 \in \mathbb{R}$.

5: Update $x_{k+1} \leftarrow x_k - \alpha_k B_k \nabla f(x_k)$.

6: Let $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$.

7: Let $\rho_k = (y_k^T s_k)^{-1}$ and $V_k = I_n - \rho_k y_k s_k^T$.

8: Update $B_{k+1} \leftarrow V_k^T B_k V_k + \rho_k s_k s_k^T$ and $k \leftarrow k + 1$.

9: **end while**

3 Kernel Gradient Descent

The goal of this section is to introduce and investigate the neural tangent kernel, which was analyzed by Arthur Jacot, Franck Gabriel and Clément Holger in [2]. We start by motivating and giving the definition of functional derivatives and functional gradient descent. This leads to the definition of multi-dimensional kernels, which enable us to generalize the functional gradient descent to so called kernel gradient descent. Having this concept in mind, we will prove that the training of neural networks in the parameter space is linked to kernel gradient descent in the function space with respect to a special kernel, the neural tangent kernel.

3.1 Functional Derivatives

As seen in section 2.2, the training of neural networks via parameter optimization is usually accomplished by minimizing the empirical error

$$E : \mathbb{R}^n \to \mathbb{R} : w \mapsto \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\big(f(x_i, w), y_i\big) + \lambda \|w\|^2$$

on labeled training data $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}^p$ for $i = 1, \dots, m$. Due to the non-convexity of the error function E it is in general hard to find a global minimum. However, we will see, that this problem can be prevented by moving from the parameter space $\mathbb{R}^n$ to the function space

$$\mathcal{F} := \Big\{ f : \mathbb{R}^d \to \mathbb{R}^p \Big\}.$$

That is we are no longer optimizing the specific parameters of the unknown target function but rather the function itself. This requires the definition of a cost

$$C : \mathcal{F} \to \mathbb{R} : f \mapsto \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\big(f(x_i), y_i\big),$$

associating some cost to each element of the function space $\mathcal{F}$. In this notion, we can write

$$E = C \circ (w \mapsto f(x, w)).$$

In contrast to the weight optimization approach, the choice of a convex loss function $\mathcal{L}$ provides a convex cost C, since by lemma 2.12 the sum of convex functions is convex again. We will introduce the functional gradient descent method, that enables us to minimize C in the function space $\mathcal{F}$ and is an analog to the gradient descent method in the parameter space $\mathbb{R}^n$. Since C is convex in the function space, convergence to a global minimum will be guaranteed in contrast to the non-convex parameter optimization problem.

Note that this approach is great for theoretical analysis due to the advantage of convexity. In practice however, we are still in need of the explicit parameters $w \in \mathbb{R}^n$, that describe the objective function. Nevertheless we can use the analysis of functional gradient descent to get a better understanding of gradient descent in the function space.

Let us start with a slight reformulation of well known results from functional analysis, that will enable us to define the functional derivative of the cost C.

Definition 3.1. *Given a real vector space $\mathcal{V}$, we define the dual space*

$$\mathcal{V}^* := \Big\{\mu : \mathcal{V} \to \mathbb{R} \mid \mu \text{ linear and continuous}\Big\}.$$

The elements of $\mathcal{V}^*$ are referred to as linear functionals.

Until now it is unclear how all of the elements in $\mathcal{F}^*$ arise. We will see that they can be constructed by the elements of $\mathcal{F}$. This procedure is based on the following lemma.

Lemma 3.2. *Given a fixed probability distribution $\mathcal{D}_x$ on the input space $\mathbb{R}^d$, one can equip the function space $\mathcal{F}$ with the seminorm*

$$\|\cdot\|_{\mathcal{D}_x} : \mathcal{F} \to \mathbb{R} : f \mapsto \sqrt{\langle f, f\rangle_{\mathcal{D}_x}}$$

in terms of the symmetric positive semidefinite bilinear form

$$\langle\cdot,\cdot\rangle_{\mathcal{D}_x} : \mathcal{F}\times\mathcal{F} \to \mathbb{R} : (f,g) \mapsto \langle f, g\rangle_{\mathcal{D}_x} := \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[f(x)^T g(x)\Big].$$

Proof. Let $f_1, f_2, f, g \in \mathcal{F}$ and $\lambda \in \mathbb{R}$. $\langle\cdot,\cdot\rangle_{\mathcal{D}_x}$ is indeed symmetric and bilinear since it suffices

(i) $\langle f_1 + f_2, g\rangle_{\mathcal{D}_x} = \langle f_1, g\rangle_{\mathcal{D}_x} + \langle f_2, g\rangle_{\mathcal{D}_x}$:

$$\begin{aligned}\langle f_1 + f_2, g\rangle_{\mathcal{D}_x} &= \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[\big(f_1(x) + f_2(x)\big)^T g(x)\Big]\\ &= \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[f_1(x)^T g(x) + f_2(x)^T g(x)\Big]\\ &= \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[f_1(x)^T g(x)\Big] + \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[f_2(x)^T g(x)\Big]\\ &= \langle f_1, g\rangle_{\mathcal{D}_x} + \langle f_2, g\rangle_{\mathcal{D}_x},\end{aligned}$$

(ii) $\langle \lambda f, g\rangle_{\mathcal{D}_x} = \lambda\langle f, g\rangle_{\mathcal{D}_x}$:

$$\langle \lambda f, g\rangle_{\mathcal{D}_x} = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[\lambda f(x)^T g(x)\Big] = \lambda\cdot\mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[f(x)^T g(x)\Big] = \lambda\langle f, g\rangle_{\mathcal{D}_x},$$

(iii) $\langle f, g\rangle_{\mathcal{D}_x} = \langle g, f\rangle_{\mathcal{D}_x}$:

$$\langle f, g\rangle_{\mathcal{D}_x} = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[f(x)^T g(x)\Big] = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[g(x)^T f(x)\Big] = \langle g, f\rangle_{\mathcal{D}_x}.$$

Furthermore $\langle\cdot,\cdot\rangle_{\mathcal{D}_x}$ is positive semidefinite, since for any $f \in \mathcal{F}$ it holds, that

$$\langle f, f\rangle_{\mathcal{D}_x} = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[f(x)^T f(x)\Big] = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[\sum_{i=1}^{d} f_i(x)^2\Big] \geq 0.$$

Thus $\|\cdot\|_{\mathcal{D}_x} = \sqrt{\langle\cdot,\cdot\rangle_{\mathcal{D}_x}}$ is a seminorm on the function space $\mathcal{F}$. □

Lemma 3.3. *The seminorm $\|\cdot\|_{\mathcal{D}_x}$ induces the pseudometric*

$$d : \mathcal{F}\times\mathcal{F} \to \mathbb{R}_+ : (f,g) \mapsto \big\|f - g\big\|_{\mathcal{D}_x},$$

such that $(\mathcal{F}, d)$ is a complete pseudometric space.

Proof. Let $f, g, h \in \mathcal{F}$. d is indeed a pseudometric since it suffices

(i) $d(f,f) = 0$:

$$d(f,f) = \big\|f - f\big\|_{\mathcal{D}_x} = 0,$$

(ii) $d(f,g) = d(g,f)$:

$$d(f,g) = \big\|f - g\big\|_{\mathcal{D}_x} = \big\|g - f\big\|_{\mathcal{D}_x} = d(g,f),$$

(iii) $d(f,h) \leq d(f,g) + d(g,h)$:

$$\begin{aligned} d(f,h) &= \big\|f - h\big\|_{\mathcal{D}_x} \\ &= \big\|f - g + g - h\big\|_{\mathcal{D}_x} \\ &\leq \big\|f - g\big\|_{\mathcal{D}_x} + \big\|g - h\big\|_{\mathcal{D}_x} = d(f,g) + d(g,h). \end{aligned}$$

The completeness of $\mathcal{F}$ arises from the fact, that every Cauchy sequence in $\mathbb{R}$ converges. □

Based on the completeness of $\mathcal{F}$ one can introduce the concept of orthogonal projections.

Lemma 3.4. *Let $\mathcal{A} \subset \mathcal{F}$ be non-empty, closed and convex. For every $f \in \mathcal{F}$ there exists some $a^* \in \mathcal{A}$ such that*

$$\big\|f - a^*\big\|^2_{\mathcal{D}_x} = \inf_{a \in \mathcal{A}} \big\|f - a\big\|^2_{\mathcal{D}_x}.$$

Such an element a^* is called orthogonal projection of f on $\mathcal{A}$.

Proof. Let $f \in \mathcal{F}$. There exists a sequence $(a_n)_{n \in \mathbb{N}} \subset \mathcal{A}$ such that

$$\lim_{n \to \infty} \big\|f - a_n\big\|^2_{\mathcal{D}_x} = \inf_{a \in \mathcal{A}} \big\|f - a\big\|^2_{\mathcal{D}_x} =: \delta.$$

For $n, m \in \mathbb{N}$ we can derive

$$\begin{aligned} \big\|a_n - a_m\big\|^2_{\mathcal{D}_x} &= \big\|(f - a_n) - (f - a_m)\big\|^2_{\mathcal{D}_x} \\ &= \big\|f - a_n\big\|^2_{\mathcal{D}_x} + \big\|f - a_m\big\|^2_{\mathcal{D}_x} - 2\big\langle f - a_n, f - a_m \big\rangle_{\mathcal{D}_x}. \end{aligned}$$

Furthermore we have

$$\big\|(f - a_n) + (f - a_m)\big\|^2_{\mathcal{D}_x} = \big\|f - a_n\big\|^2_{\mathcal{D}_x} + \big\|f - a_m\big\|^2_{\mathcal{D}_x} + 2\big\langle f - a_n, f - a_m \big\rangle_{\mathcal{D}_x}.$$

Combining both expressions yields

$$\begin{aligned} \big\|a_n - a_m\big\|^2_{\mathcal{D}_x} &= 2\big\|f - a_n\big\|^2_{\mathcal{D}_x} + 2\big\|f - a_m\big\|^2_{\mathcal{D}_x} - \big\|(f - a_n) + (f - a_m)\big\|^2_{\mathcal{D}_x} \\ &= 2\big\|f - a_n\big\|^2_{\mathcal{D}_x} + 2\big\|f - a_m\big\|^2_{\mathcal{D}_x} - 4\big\|f - \frac{1}{2}(a_n + a_m)\big\|^2_{\mathcal{D}_x}. \end{aligned}$$

Since $\mathcal{A}$ is convex, $\frac{1}{2}(a_n + a_m) \in \mathcal{A}$, such that for arbitrary $\epsilon > 0$ there exists $N \in \mathbb{N}$ with

$$\big\|a_n - a_m\big\|^2_{\mathcal{D}_x} \leq 2(\delta + \epsilon) + 2(\delta + \epsilon) - 4\delta = 4\epsilon$$

for any $n, m > N$. Thus $(a_n)_{n \in \mathbb{N}}$ is a Cauchy sequence. Since $\mathcal{F}$ is complete and $\mathcal{A}$ is closed, $(a_n)_{n \in \mathbb{N}}$ converges to some $a^* \in \mathcal{A}$ with $\big\|f - a^*\big\|^2_{\mathcal{D}_x} = \delta$. □

Lemma 3.4 is a weakening of the Hilbert projection theorem stating that, given a Hilbert space, it even exists a unique orthogonal projection. In our case we lose the uniqueness, beacuse $\mathcal{F}$ is only a pseudometric space and not a metric space. However this is just a side note, since the existence of an orthogonal projection suffices for the further theory.

Corollary 3.5. *Let $f \in \mathcal{F}$ and $\mathcal{A} \subset \mathcal{F}$ be a non-empty and closed subspace. For every $a \in \mathcal{A}$ it holds, that $\langle f - a^*, a\rangle_{\mathcal{D}_x} = 0$, where $a^* \in \mathcal{A}$ denotes an orthogonal projection of f on $\mathcal{A}$.*

Proof. Let $a \in \mathcal{A}$ and $\lambda \in \mathbb{R}$. Since $\mathcal{A}$ is a subspace we can infer $a^* + \lambda a \in \mathcal{A}$. Thus we derive

$$\big\|(a^* + \lambda a) - f\big\|_{\mathcal{D}_x}^2 \geq \big\|a^* - f\big\|_{\mathcal{D}_x}^2$$

by the minimality of a^*. Rearranging the inequality yields

$$\big\|(a^* - f) + \lambda a\big\|_{\mathcal{D}_x}^2 - \big\|a^* - f\big\|_{\mathcal{D}_x}^2 = 2\big\langle a^* - f, \lambda a\big\rangle_{\mathcal{D}_x} + \big\|\lambda a\big\|_{\mathcal{D}_x}^2 \geq 0,$$

giving rise to the definition of the non-negative function

$$\phi : \mathbb{R} \to \mathbb{R} : \lambda \mapsto 2\lambda\big\langle a^* - f, a\big\rangle_{\mathcal{D}_x} + \lambda^2\big\|a\big\|_{\mathcal{D}_x}^2.$$

Under the assumption that $\big\langle a^* - f, a\big\rangle_{\mathcal{D}_x} \neq 0$ and $\|a\|_{\mathcal{D}_x}^2 \neq 0$, we derive

$$\phi\Big(-\frac{\big\langle a^* - f, a\big\rangle_{\mathcal{D}_x}}{\|a\|_{\mathcal{D}_x}^2}\Big) = -2 \cdot \frac{\big\langle a^* - f, a\big\rangle_{\mathcal{D}_x}^2}{\|a\|_{\mathcal{D}_x}^2} + \frac{\big\langle a^* - f, a\big\rangle_{\mathcal{D}_x}^2}{\|a\|_{\mathcal{D}_x}^2} = -\frac{\big\langle a^* - f, a\big\rangle_{\mathcal{D}_x}^2}{\|a\|_{\mathcal{D}_x}^2} < 0$$

and under the assumption that $\big\langle a^* - f, a\big\rangle_{\mathcal{D}_x} \neq 0$ and $\|a\|_{\mathcal{D}_x}^2 = 0$, we derive

$$\phi\Big(-\big\langle a^* - f, a\big\rangle_{\mathcal{D}_x}\Big) = -2 \cdot \big\langle a^* - f, a\big\rangle_{\mathcal{D}_x}^2 < 0,$$

which is a contradiction to $\phi(\lambda) \geq 0$ for every $\lambda \in \mathbb{R}$. Thus $\big\langle f - a^*, a\big\rangle_{\mathcal{D}_x} = 0$. □

Finally, the concept of orthogonal projections enables us to represent any functional $\mu \in \mathcal{F}^*$ based on a representer $d \in \mathcal{F}$.

Lemma 3.6. *The map*

$$J : \mathcal{F} \to \mathcal{F}^* : d \mapsto \langle d, \cdot\rangle_{\mathcal{D}_x}$$

is linear and surjective.

Proof. The linearity follows directly from the linearity of $\langle \cdot, \cdot\rangle_{\mathcal{D}_x}$. To prove that J is indeed surjective, we let $\mu \in \mathcal{F}^*$ and need to find some $d \in \mathcal{F}$ such that $\mu = J(d)$.

Case $\mu = 0$: We can choose $d = 0$, with $\mu = 0 = J(d)$.

Case $\mu \neq 0$: Due to the linearity of μ we can normalize and find some $e \in \mathcal{F}$ with $\mu[e] = 1$. Let $N := \mu^{-1}(0)$ denote the kernel of μ, which is non-empty, closed and convex since μ is linear and continuous. Thus by Lemma 3.4 there exists some $p_e \in N$, such that p_e is an orthogonal projection of e on N. Define $f := e - p_e$, then $f \notin N$ since

$$\mu[f] = \mu[e] - \mu[p_e] = \mu[e] = 1.$$

Based on the linearity of μ, we can derive for arbitrary $g \in \mathcal{F}$, that

$$\mu\big[g - \mu[g]f\big] = \mu[g] - \mu[g]\mu[f] = \mu[g] - \mu[g] \cdot 1 = 0,$$

which implies $h := g - \mu[g]f \in N$. Thus $\langle f, h\rangle_{\mathcal{D}_x} = 0$ by corollary 3.5 and we conclude

$$\langle f, g\rangle_{\mathcal{D}_x} = \langle f, h\rangle_{\mathcal{D}_x} + \langle f, \mu[g]f\rangle_{\mathcal{D}_x} = 0 + \mu[g] \cdot \langle f, f\rangle_{\mathcal{D}_x} = \mu[g] \cdot \|f\|^2_{\mathcal{D}_x}.$$

Now we can assume that $\|f\|^2_{\mathcal{D}_x} \neq 0$. Otherwise we would have $\langle f, g\rangle_{\mathcal{D}_x} = 0$ for every $g \in \mathcal{F}$ and therefore $f = 0$ which is a contradiction to $f \notin N$. Rearranging terms results in

$$\mu[g] = \big\langle f/\|f\|^2_{\mathcal{D}_x}, g\big\rangle_{\mathcal{D}_x} = J\big(f/\|f\|^2_{\mathcal{D}_x}\big)[g].$$

Hence we have found $d = f/\|f\|^2_{\mathcal{D}_x}$ with $\mu = J(d)$. This shows surjectivity. □

Thus for each functional $\mu \in \mathcal{F}^*$ there exists $d \in \mathcal{F}$ such that $\mu = \langle d, \cdot\rangle_{\mathcal{D}_x}$. In other words

$$\mathcal{F}^* = \Big\{\mu : \mathcal{F} \to \mathbb{R} : f \mapsto \langle d, f\rangle_{\mathcal{D}_x} \mid d \in \mathcal{F}\Big\}.$$

Lemma 3.6 is a weakening of the Riesz representation theorem stating that any Hilbert space $\mathcal{H}$ is isometric and isomorphic to its dual space $\mathcal{H}^*$ via the map $J : \mathcal{H} \to \mathcal{H}^* : x \mapsto \langle x, \cdot\rangle_{\mathcal{H}}$.

Since $\langle \cdot, \cdot\rangle_{\mathcal{D}_x}$ is only positive semidefinite and not positive definite we lose the property of J being isometric and therefore injective. Again, this is just a side note, since surjectivity will be sufficient enough for the further theory. Note that, based on the definition of $\mathcal{D}_x$, J maps two elements of $\mathcal{F}$ to the same functional, if and only if they are equal on the data.

The representation theorem has great importance for the functional gradient descent method. Another crucial ingredient is the definition of functional derivatives and functional gradient.

Definition 3.7. *Let $\phi, f_0 \in \mathcal{F}$. A functional $\mu : \mathcal{F} \to \mathbb{R}$ is said to be differentiable at the point f_0 in direction ϕ, if*

$$\int \frac{\partial \mu}{\partial f_0(x)}(x)\phi(x)\ dx = \lim_{\epsilon \to 0} \frac{\mu[f_0 + \epsilon\phi] - \mu[f_0]}{\epsilon}$$

exists. The functional gradient of μ at f_0 is defined as

$$\nabla\mu|_{f_0} : \mathbb{R}^d \to \mathbb{R}^p : x \mapsto \frac{\partial \mu}{\partial f_0(x)}(x).$$

If the limit exists for every $\phi \in \mathcal{F}$ we define the functional derivative of μ at f_0 as

$$\partial_f\mu|_{f_0} : \mathcal{F} \to \mathbb{R} : \phi \mapsto \frac{\partial}{\partial \epsilon}\Big[\mu[f_0 + \epsilon\phi]\Big]_{\epsilon=0}.$$

Based on these definitions, we can iteratively minimize functionals $\mu : \mathcal{F} \to \mathbb{R}$ in a similar manner to the gradient descent method.
With a slight change in notation, the same argumentation that was used for convergence of gradient descent on functions, can be applied to the functional gradient descent method.

Let $\mu : \mathcal{F} \to \mathbb{R}$ be differentiable. Functional gradient descent proceeds as follows:

1) Initialize $k = 0$, choose $\epsilon \geq 0 \in \mathbb{R}$ and pick some random $f_0 \in \mathcal{F}$.

2) WHILE $\left\|\nabla\mu|_{f_k}\right\|_{\mathcal{D}_x} > \epsilon$ DO:

3) Choose a step size $\alpha_k > 0 \in \mathbb{R}$.

4) Set $f_{k+1} = f_k - \alpha_k \cdot \nabla\mu|_{f_k}$ and $k = k + 1$.

Until now we have a theoretical understanding of functional gradient descent in the function space $\mathcal{F}$. However there is not yet a connection to gradient descent in the parameter space. In theory, functional gradient descent is an excellent tool to guarantee convergence but in practice we need the explicit parameter vector $w \in \mathbb{R}^n$ to represent the optimal function due to functional gradient descent. This relationship between functional gradient descent and parameter gradient descent can be established by so called kernel gradient descent.

3.2 Kernel Gradient

We start by recalling the basic definition of one-dimensional kernels.

Definition 3.8. *Let X be an arbitrary set. A one-dimensional kernel over X is a map*

$$K : X \times X \to \mathbb{R} : (x, x') \mapsto K(x, x').$$

A kernel K is said to be positive definite symmetric if for any $m \in \mathbb{N}$ and $x \in X^m$, the matrix with entries $M_{i,j} = K(x_i, x_j)$ for $i, j = 1, \ldots, m$ is symmetric and positive semidefinite.

This thesis will work with kernels over the input space $\mathbb{R}^d$, not only with one-dimensional but with multi-dimensional kernels, which motivates the following definition.

Definition 3.9. *A multi-dimensional kernel over $\mathbb{R}^d$ is a function*

$$K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{p\times p} : (x, x') \mapsto K(x, x'),$$

such that $K(x, x') = K(x', x)^T$ for any $x, x' \in \mathbb{R}^d$.

Based on the marginal distribution $\mathcal{D}_x$, kernels induce symmetric bilinear forms.

Lemma 3.10. *A kernel $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{p\times p}$ induces the symmetric bilinear form*

$$\langle\cdot,\cdot\rangle_K : \mathcal{F} \times \mathcal{F} \to \mathbb{R} : (f, g) \mapsto \langle f, g\rangle_K := \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[f(x)^T K(x, x')g(x')\Big],$$

where $x, x' \in \mathbb{R}^d$ are drawn independently according to $\mathcal{D}_x$.

Proof. Let $f_1, f_2, f, g \in \mathcal{F}$ and $\lambda \in \mathbb{R}$. $\langle\cdot,\cdot\rangle_K$ is indeed symmetric and bilinear since it suffices

(i) $\langle f_1 + f_2, g\rangle_K = \langle f_1, g\rangle_K + \langle f_2, g\rangle_K$:

$$\begin{aligned}\langle f_1 + f_2, g\rangle_K &= \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[\big(f_1(x) + f_2(x)\big)^T K(x,x')g(x')\Big] \\ &= \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[f_1(x)^T K(x,x')g(x') + f_2(x)^T K(x,x')g(x')\Big] \\ &= \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[f_1(x)^T K(x,x')g(x')\Big] + \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[f_2(x)^T K(x,x')g(x')\Big] \\ &= \langle f_1, g\rangle_K + \langle f_2, g\rangle_K,\end{aligned}$$

(ii) $\langle \lambda f, g\rangle_K = \lambda\langle f, g\rangle_K$:

$$\begin{aligned}\langle \lambda f, g\rangle_K &= \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[\lambda f(x)^T K(x,x')g(x')\Big] \\ &= \lambda \cdot \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[f(x)^T K(x,x')g(x')\Big] = \lambda\langle f, g\rangle_K,\end{aligned}$$

(iii) $\langle f, g\rangle_K = \langle g, f\rangle_K$:

$$\begin{aligned}\langle f, g\rangle_K &= \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[f(x)^T K(x,x')g(x')\Big] \\ &= \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[f(x)^T K(x',x)^T g(x')\Big] \\ &= \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[g(x')^T K(x',x)f(x)\Big] = \langle g, f\rangle_K.\end{aligned}$$

In contrast to $\langle\cdot,\cdot\rangle_{\mathcal{D}_x}$, the bilinear map $\langle\cdot,\cdot\rangle_K$ is not necessarily positive semidefinite. □

Using these bilinear forms, one can generalize the definition of positive definite kernels.

Definition 3.11. *A kernel $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{p\times p}$ is called positive definite with respect to the seminorm $\|\cdot\|_{\mathcal{D}_x}$ if for every $f \in \mathcal{F}$ it holds, that $\|f\|_{\mathcal{D}_x} > 0$ implies $\langle f, f\rangle_K > 0$.*

We can use positive definite kernels to equip $\mathcal{F}$ with another seminorm.

Lemma 3.12. *Given a kernel $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{p\times p}$, that is positive definite with respect to the seminorm $\|\cdot\|_{\mathcal{D}_x}$ we can use the bilinear form $\langle\cdot,\cdot\rangle_K$ to define another seminorm*

$$\|\cdot\|_K : \mathcal{F} \to \mathbb{R} : f \mapsto \sqrt{\langle\cdot,\cdot\rangle_K}.$$

Proof. Since $\langle\cdot,\cdot\rangle_K$ is symmetric by lemma 3.10 it remains to show that $\langle\cdot,\cdot\rangle_K$ is positive semidefinite. For this we let $f \in \mathcal{F}$ be arbitrary and investigate the following cases.

Case $\|f\|_{\mathcal{D}_x} > 0$: Since K is positive definite with respect to $\|\cdot\|_{\mathcal{D}_x}$, we derive $\langle f, f\rangle_K > 0$.

Case $\|f\|_{\mathcal{D}_x} = 0$: By the definition of $\langle\cdot,\cdot\rangle_{\mathcal{D}_x}$, we derive

$$\big\|f\big\|_{\mathcal{D}_x}^2 = \langle f, f\rangle_{\mathcal{D}_x} = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[f(x)^T f(x)\Big] = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[\sum_{i=1}^{d} f_i(x)^2\Big] = 0.$$

Thus for every $x \sim \mathcal{D}_x$ it holds that $f(x) = 0$. Therefore we conclude

$$\langle f, f\rangle_K = \mathop{\mathbb{E}}_{x,x'\sim\mathcal{D}_x}\Big[f(x)^T K(x,x')f(x')\Big] = 0.$$

In total we have shown that $\langle f, f\rangle_K > 0$ for every $f \in \mathcal{F}$, such that $\|\cdot\|_K$ is a seminorm. □

This concludes the necessary knowledge on kernels. Next we will introduce the kernel gradient of a functional, which is based on the partial application of kernels.

Definition 3.13. *Let $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{p\times p}$ be a kernel. For $j = 1, \dots, p$ one defines the partial application of K as the function*

$$K_{\cdot,j} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^p : (x, x') \mapsto K(x, x')_{\cdot,j},$$

where $K(x, x')_{\cdot,j}$ denotes the j-th column of the $p \times p$ matrix $K(x, x')$.

Fixing one argument of $K_{\cdot,j}$ results in a function in $\mathcal{F}$. This enables the following definition.

Definition 3.14. *Given a kernel $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{p\times p}$, we define the map*

$$\Phi_K : \mathcal{F}^* \to \mathcal{F} : \mu \mapsto f_\mu,$$

mapping a dual element $\mu \in \mathcal{F}$ to the function

$$f_\mu : \mathbb{R}^d \to \mathbb{R}^p : x \mapsto \begin{bmatrix} \mu[K_{\cdot,1}(\cdot, x)] \\ \vdots \\ \mu[K_{\cdot,p}(\cdot, x)] \end{bmatrix}.$$

Based on the definition of Φ_K we can define the kernel gradient of a functional.

Definition 3.15. *Let $f_0 \in \mathcal{F}$, $\mu : \mathcal{F} \to \mathbb{R}$ be a differentiable functional with $\partial_f \mu|_{f_0} \in \mathcal{F}^*$ and K be a kernel. The kernel gradient of μ at f_0 with respect to K is defined as*

$$\nabla_K \mu|_{f_0} : \mathbb{R}^d \to \mathbb{R}^p : x \mapsto \Phi_K\Big(\partial_f \mu|_{f_0}\Big)(x).$$

Using this definition, we can formulate the kernel gradient descent method.

Let K be a kernel and $\mu : \mathcal{F} \to \mathbb{R}$ be a differentiable functional with $\partial_f \mu|_{f_0} \in \mathcal{F}^*$. Kernel gradient descent with respect to the kernel K proceeds as follows:

1) Initialize $k = 0$, choose $\epsilon \geq 0 \in \mathbb{R}$ and pick some random $f_0 \in \mathcal{F}$.

2) WHILE $\big\|\nabla_K \mu|_{f_k}\big\|_{\mathcal{D}_x} > \epsilon$ DO:

   3) Choose a step size $\alpha_k > 0 \in \mathbb{R}$.

   4) Set $f_{k+1} = f_k - \alpha_k \cdot \nabla_K \mu|_{f_k}$ and $k = k + 1$.

In the following, we will apply kernel gradient descent to minimize the cost C. During this we are especially interested in the evolution of the parameters defining the sequence $(f_k)_{k\in\mathbb{N}_0}$.

With the choice of a differentiable loss function, the cost functional $C : \mathcal{F} \to \mathbb{R}$ is differentiable as well, such that the functional derivative $\partial_f C|_{f_0}$ is well-defined for every $f_0 \in \mathcal{F}$.

In the following we will make the restriction, that the marginal distribution $\mathcal{D}_x$ on the input space $\mathbb{R}^d$ is given by the empirical distribution on a finite subset of $\mathbb{R}^d$. This means there exist $m \in \mathbb{N}$ and $x_1, \dots, x_m \in \mathbb{R}^d$ such that

$$\mathcal{D}_x = \frac{1}{m} \sum_{i=1}^{m} \delta_{x_i},$$

where δ_{x_i} denotes the Dirac measure in x_i for $i = 1, \dots, m$. Under this assumption, the cost C depends only on the values of f at a finite dataset. Thus we conclude, that $\partial_f C|_{f_0}$ is linear and continuous and therefore $\partial_f C|_{f_0} \in \mathcal{F}^*$ for every $f_0 \in \mathcal{F}$. Hence by Lemma 3.6 there exists a representer $d|_{f_0} \in \mathcal{F}$, such that the functional derivative of C can be written as

$$\partial_f C|_{f_0} = \langle d|_{f_0}, \cdot \rangle_{\mathcal{D}_x}.$$

Next we will use this restriction on the distribution $\mathcal{D}_x$, to simplify the notation of Φ_K.

Lemma 3.16. *Let $\mu = \langle d, \cdot \rangle_{\mathcal{D}_x} \in \mathcal{F}^*$ and $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{p \times p}$ be a kernel, then*

$$\Phi_K(\mu) = \frac{1}{m} \sum_{i=1}^{m} K(\cdot, x_i) d(x_i),$$

where $x_1, \dots, x_m \in \mathbb{R}^d$ denotes the finite support of the empiricial distribution $\mathcal{D}_x$.

Proof. Let $x \in \mathbb{R}^d$ be arbitrary. By the definition of Φ_K we have

$$\Phi_K\big(\mu\big)(x) = \Phi_K\Big(\langle d, \cdot \rangle_{\mathcal{D}_x}\Big)(x) = \begin{bmatrix} \mu[K_{\cdot,1}(\cdot, x)] \\ \vdots \\ \mu[K_{\cdot,p}(\cdot, x)] \end{bmatrix} = \begin{bmatrix} \big\langle d, K_{\cdot,1}(\cdot, x) \big\rangle_{\mathcal{D}_x} \\ \vdots \\ \big\langle d, K_{\cdot,p}(\cdot, x) \big\rangle_{\mathcal{D}_x} \end{bmatrix}.$$

Recalling the definition of $\langle \cdot, \cdot \rangle_{\mathcal{D}_x}$, it holds that

$$\big\langle d, K_{\cdot,j}(\cdot, x) \big\rangle_{\mathcal{D}_x} = \mathop{\mathbb{E}}_{x' \sim \mathcal{D}_x} \Big[d(x')^T K_{\cdot,j}(x', x) \Big]$$

for $j = 1, \dots, p$. Thus we derive

$$\Phi_K\big(\mu\big)(x) = \begin{bmatrix} \big\langle d, K_{\cdot,1}(\cdot, x) \big\rangle_{\mathcal{D}_x} \\ \vdots \\ \big\langle d, K_{\cdot,p}(\cdot, x) \big\rangle_{\mathcal{D}_x} \end{bmatrix} = \mathop{\mathbb{E}}_{x' \sim \mathcal{D}_x} \begin{bmatrix} d(x')^T K_{\cdot,1}(x', x) \\ \vdots \\ d(x')^T K_{\cdot,p}(x', x) \end{bmatrix} = \mathop{\mathbb{E}}_{x' \sim \mathcal{D}_x} \Big[d(x')^T K(x', x) \Big]^T.$$

With $K(x', x)^T = K(x, x')$ for every $x, x' \in \mathbb{R}^d$ by definition, we conclude

$$\Phi_K\big(\mu\big)(x) = \mathop{\mathbb{E}}_{x' \sim \mathcal{D}_x} \Big[d(x')^T K(x', x) \Big]^T = \mathop{\mathbb{E}}_{x' \sim \mathcal{D}_x} \Big[K(x', x)^T d(x') \Big] = \mathop{\mathbb{E}}_{x' \sim \mathcal{D}_x} \Big[K(x, x') d(x') \Big].$$

The last step consists of using the assumption, that $\mathcal{D}_x$ is the empirical distribution on a finite dataset $x_1, \dots, x_m \in \mathbb{R}^d$. This enables us to derive

$$\Phi_K\big(\mu\big)(x) = \mathop{\mathbb{E}}_{x' \sim \mathcal{D}_x} \Big[K(x, x') d(x') \Big] = \frac{1}{m} \sum_{i=1}^{m} K(x, x_i) d(x_i).$$

This proves the statement of the lemma. □

By lemma 3.16 we can write the kernel gradient of C at f_0 with respect to some kernel K as

$$\nabla_K C|_{f_0}(x) = \Phi_K\Big(\langle d|_{f_0}, \cdot\rangle_{\mathcal{D}_x}\Big)(x) = \frac{1}{m}\sum_{i=1}^{m} K(x, x_i) d|_{f_0}(x_i).$$

Due to the definition of the kernel, $\nabla_K C|_{f_0}$ has the advantage of being well-defined on the whole input space $\mathbb{R}^d$, where as the partial derivative $\partial_f C|_{f_0} = \langle d|_{f_0}, \cdot\rangle_{\mathcal{D}_x}$ is based on the bilinear form $\langle\cdot,\cdot\rangle_{\mathcal{D}_x}$ and therefore only defined on the finite dataset.

Definition 3.17. *Let K be a kernel. A time dependent function $f : \mathbb{R} \to \mathcal{F}$ is said to follow the kernel gradient descent on C with respect to K if it satisfies the differential equation*

$$\partial_t f(t) = -\nabla_K C|_{f(t)}.$$

We can think of such a function as a continuous extension to the sequence produced by kernel gradient descent on C, which can be used to investigate the change in C.

Lemma 3.18. *Let K be a positive definite kernel with respect to $\|\cdot\|_{\mathcal{D}_x}$ and $f : \mathbb{R} \to \mathcal{F}$ be a time dependent function, that follows the kernel gradient descent on C with respect to K, then the evolution of $C \circ f$ during kernel gradient descent is expressed by*

$$\partial_t C|_{f(t)} = -\big\|d|_{f(t)}\big\|_K^2,$$

where $d|_{f(t)} \in \mathcal{F}$ denotes the representer of $\partial_f C|_{f(t)} = \big\langle d|_{f(t)}, \cdot\big\rangle_{\mathcal{D}_x} \in \mathcal{F}^*$ for $t \in \mathbb{R}$.

Proof. Using the notation $C|_{f(t)} = C\big[f(t)\big]$, we derive

$$\partial_t C|_{f(t)} = \partial_f C|_{f(t)}\big[\partial_t f(t)\big] = \big\langle d|_{f(t)}, \cdot\big\rangle_{\mathcal{D}_x}\big[\partial_t f(t)\big] = \big\langle d|_{f(t)}, \partial_t f(t)\big\rangle_{\mathcal{D}_x}.$$

Since f follows the kernel gradient descent on C with respect to K this can be rewritten as

$$\partial_t C|_{f(t)} = \big\langle d|_{f(t)}, \partial_t f(t)\big\rangle_{\mathcal{D}_x} = \big\langle d|_{f(t)}, -\nabla_K C|_{f(t)}\big\rangle_{\mathcal{D}_x}.$$

Recalling the definition of $\nabla_K C|_{f(t)}$, which was given by

$$\nabla_K C|_{f(t)} = \frac{1}{m}\sum_{i=1}^{m} K(\cdot, x_i) d|_{f(t)}(x_i),$$

we derive by the linearity of $\langle\cdot,\cdot\rangle_{\mathcal{D}_x}$, that

$$\begin{aligned}
\big\langle d|_{f(t)}, -\nabla_K C|_{f(t)}\big\rangle_{\mathcal{D}_x} &= -\frac{1}{m}\sum_{i=1}^{m} \big\langle d|_{f(t)}, K(\cdot, x_i) d|_{f(t)}(x_i)\big\rangle_{\mathcal{D}_x} \\
&= -\frac{1}{m}\sum_{i=1}^{m} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[d|_{f(t)}(x)^T K(x, x_i) d|_{f(t)}(x_i)\Big] \\
&= -\mathop{\mathbb{E}}_{x'\sim\mathcal{D}_x}\mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[d|_{f(t)}(x)^T K(x, x') d|_{f(t)}(x')\Big] = -\big\langle d|_{f(t)}, d|_{f(t)}\big\rangle_K.
\end{aligned}$$

Finally we conclude, that

$$\partial_t C|_{f(t)} = \big\langle d|_{f(t)}, -\nabla_K C|_{f(t)}\big\rangle_{\mathcal{D}_x} = -\big\langle d|_{f(t)}, d|_{f(t)}\big\rangle_K = -\big\|d|_{f(t)}\big\|_K^2.$$

This completes the proof. □

The lemma guarantees the convergence of C to a critical point, as $\partial_t C|_{f(t)} \leq 0$. In the case where C is convex and bounded from below, this implies the convergence of $f(t)$ to a global minimum $f^* \in \mathcal{F}$ for $t \to \infty$.

3.3 Kernel Approximation

This subsection illustrates the relationship between gradient descent in the parameter space and kernel gradient descent in the function space under the assumption, that the objective function depends linear on its parameters. The example given in here, will be generalized later on, leading to the definition of the neural tangent kernel.

To start things off, we let $\mathcal{D}_{\mathcal{F}}$ be any probability distribution on the function space $\mathcal{F}$ and construct a kernel K, such that for $x, x' \in \mathbb{R}^d$ the image $K(x, x')$ is a $p \times p$ matrix defined by

$$K_{i,j}(x, x') := \mathop{\mathbb{E}}_{f \sim \mathcal{D}_{\mathcal{F}}} \Big[f_i(x) \cdot f_j(x) \Big], \quad i, j = 1, \dots, p.$$

We will see, that this kernel can be approximated by random functions. For this we choose some $n \in \mathbb{N}$ and draw n random functions $f^{(k)} : \mathbb{R}^d \to \mathbb{R}^p$ for $k = 1, \dots, n$ independently from the distribution $\mathcal{D}_{\mathcal{F}}$. Using these, we can define the linear map

$$F^{lin} : \mathbb{R}^n \to \mathcal{F} : w \mapsto f_w := \frac{1}{\sqrt{n}} \sum_{k=1}^{n} w_k f^{(k)},$$

whose partial derivatives for $k = 1, \dots, n$ are given by

$$\partial_{w_k} F^{lin} : \mathbb{R}^n \to \mathcal{F} : w \mapsto \frac{1}{\sqrt{n}} f^{(k)}.$$

Based on F^{lin} we can define another kernel, that will be used for the approximation of K.

Definition 3.19. *For $x \in \mathbb{R}^p$ we define the symmetric matrix $x \otimes x := M \in \mathbb{R}^{p \times p}$, where*

$$M_{i,j} := x_i \cdot x_j, \quad i, j = 1, \dots, p.$$

Definition 3.20. *Given n random functions $f^{(k)} : \mathbb{R}^d \to \mathbb{R}^p$ for $k = 1, \dots, n$ drawn i.i.d. according to some distribution $\mathcal{D}_{\mathcal{F}}$ on the function space $\mathcal{F}$, the tangent kernel is defined as*

$$\tilde{K} := \sum_{k=1}^{n} \partial_{w_k} F^{lin}(w) \otimes \partial_{w_k} F^{lin}(w) = \frac{1}{n} \sum_{k=1}^{n} f^{(k)} \otimes f^{(k)}.$$

In other words, $\tilde{K}$ maps a tuple $(x, x') \in \mathbb{R}^d \times \mathbb{R}^d$ to the matrix $\tilde{K} \in \mathbb{R}^{p \times p}$ with entries

$$\tilde{K}_{i,j}(x, x') = \frac{1}{n} \sum_{k=1}^{n} f_i^{(k)}(x) \cdot f_j^{(k)}(x'), \quad i, j = 1, \dots, p.$$

Similar to the definition of time dependent functions, that follow the kernel gradient descent on C with respect to some kernel K we can define time dependent functions, that follow the gradient descent on $C \circ F^{lin}$. This is done by the following definition.

Definition 3.21. *A time dependent function $w : \mathbb{R} \to \mathbb{R}^n$ is said to follow the gradient descent on $C \circ F^{lin}$, if it satisfies the differential equation*

$$\partial_t w(t) = -\nabla \big(C \circ F^{lin} \big)(w(t)) = -\nabla C|_{f_{w(t)}},$$

where $-\nabla C|_{f_{w(t)}}$ denotes the gradient of $C \circ F^{lin}$ at $w(t)$.

Again, we can think of such a function as a continuous extension to the sequence produced by gradient descent on $C \circ F^{lin}$.

Lemma 3.22. *Let $w : \mathbb{R} \to \mathbb{R}^n$ be a time dependent function, that follows the gradient descent on $C \circ F^{lin}$. For $k = 1, \dots, n$, the change in w_k during gradient descent is given by*

$$\partial_t w_k(t) = -\frac{1}{\sqrt{n}} \big\langle d|_{f_{w(t)}}, f^{(k)} \big\rangle_{\mathcal{D}_x},$$

where $d|_{f_{w(t)}} \in \mathcal{F}$ denotes the representer of $\partial_f C|_{f_{w(t)}} = \big\langle d|_{f_{w(t)}}, \cdot \big\rangle_{\mathcal{D}_x} \in \mathcal{F}^*$ for $t \in \mathbb{R}$.

Proof. Using the notation in definition 3.21, we derive

$$\partial_t w_k(t) = -\partial_{w_k}\big(C \circ F\big)(w(t)) = \partial_f C|_{f_{w(t)}} \Big[-\frac{1}{\sqrt{n}} f^{(k)} \Big] = -\frac{1}{\sqrt{n}} \big\langle d|_{f_{w(t)}}, f^{(k)} \big\rangle_{\mathcal{D}_x}$$

for $k = 1, \dots, n$. This completes the proof. □

Based on this lemma we can establish a relationship between time dependent functions that follow the gradient descent on $C \circ F^{lin}$ and time dependent functions that follow the kernel gradient descent on C with respect to the tangent kernel.

Lemma 3.23. *Let $w : \mathbb{R} \to \mathbb{R}^n$ be a time dependent function, that follows the gradient descent on $C \circ F^{lin}$, then the time dependent function $F^{lin} \circ w : \mathbb{R} \to \mathcal{F} : t \mapsto f_{w(t)}$ follows the kernel gradient descent on C with respect to the tangent kernel $\tilde{K}$. Formally this is*

$$\partial_t\big(F^{lin} \circ w\big)(t) = \partial_t f_{w(t)} = -\nabla_{\tilde{K}} C|_{f_{w(t)}}.$$

Proof. By lemma 3.22 and the definition of F^{lin} it holds, that

$$\partial_t f_{w(t)} = \frac{1}{\sqrt{n}} \sum_{k=1}^{n} \partial_t w_k(t) \cdot f^{(k)} = -\frac{1}{n} \sum_{k=1}^{n} \big\langle d|_{f_{w(t)}}, f^{(k)} \big\rangle_{\mathcal{D}_x} \cdot f^{(k)}.$$

Furthermore, we can write the kernel gradient of C at $f_{w(t)}$ with respect to $\tilde{K}$ as

$$\begin{aligned}
\nabla_{\tilde{K}} C|_{f_{w(t)}} &= \frac{1}{m} \sum_{i=1}^{m} K(\cdot, x_i) \cdot d|_{f_{w(t)}}(x_i) \\
&= \frac{1}{m} \sum_{i=1}^{m} \left[\frac{1}{n} \sum_{k=1}^{n} f^{(k)} \otimes f^{(k)} \right] (\cdot, x_i) \cdot d|_{f_{w(t)}}(x_i) \\
&= \frac{1}{n} \sum_{k=1}^{n} \left[\frac{1}{m} \sum_{i=1}^{m} f^{(k)}(x_i)^T d|_{f_{w(t)}}(x_i) \right] \cdot f^{(k)} \\
&= \frac{1}{n} \sum_{k=1}^{n} \mathop{\mathbb{E}}_{x \sim \mathcal{D}_x} \left[f^{(k)}(x)^T d|_{f_{w(t)}}(x) \right] \cdot f^{(k)} \\
&= \frac{1}{n} \sum_{k=1}^{n} \big\langle f^{(k)}, d|_{f_{w(t)}} \big\rangle_{\mathcal{D}_x} \cdot f^{(k)}.
\end{aligned}$$

Comparison of the previous two expressions provides

$$\partial_t f_{w(t)} = -\frac{1}{n}\sum_{k=1}^{n} \langle d|_{f_{w(t)}}, f^{(k)} \rangle_{\mathcal{D}_x} \cdot f^{(k)} = -\nabla_{\tilde{K}} C|_{f_{w(t)}}.$$

This proves, that $F^{lin} \circ w$ follows the kernel gradient descent on C with respect to $\tilde{K}$. □

Thus kernel gradient descent on C with respect to the tangent kernel $\tilde{K}$ is equivalent to gradient descent on $C \circ F^{lin}$. For $i, j = 1, \dots, p$ the law of large numbers yields

$$\forall x, x' \in \mathbb{R}^d : \lim_{n\to\infty} \tilde{K}_{i,j}(x, x') = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} f_i^{(k)}(x) \cdot f_j^{(k)}(x') = \mathop{\mathbb{E}}_{f\sim\mathcal{D}_{\mathcal{F}}}\Big[f_i(x) \cdot f_j(x)\Big].$$

Hence the random tangent kernel $\tilde{K}$ is an approximation of the constant limiting kernel K.

3.4 Neural Tangent Kernel

The training of neural networks behaves quite similar to the kernel gradient descent with respect to the tangent kernel. However, in contrast to the situation in section 3.3, we consider the realization function

$$F : \mathbb{R}^n \to \tilde{\mathcal{F}} : w \mapsto f(x, w),$$

which is no longer linear due to the nested construction of neural networks. Therefore, the partial derivatives $\partial_{w_k} F$ are no longer independent of the parameters w, which motivates the definition of a new kernel, specifically related to neural networks.

Definition 3.24. *Given n random functions $f^{(k)} : \mathbb{R}^d \to \mathbb{R}^p$ for $k = 1, \dots, n$ drawn i.i.d. according to some distribution $\mathcal{D}_{\tilde{\mathcal{F}}}$ on $\tilde{\mathcal{F}}$, the neural tangent kernel is defined as*

$$\Theta_w := \sum_{k=1}^{n} \partial_{w_k} F(w) \otimes \partial_{w_k} F(w).$$

Thus, in contrast to the tangent kernel, which is constant during training since it is independent on the parameters w, the neural tangent kernel is random at initialization and varies during training. Therefore the relationship between gradient descent in the parameter space and kernel gradient descent with respect to the neural tangent kernel is more complex.

We will see, that in the infinite-width limit, that is the number of neurons per layer tends to infinity, the neural tangent kernel becomes deterministic at initialization and stays constant during training.

In order to simplify the further notation, we have to make some adjustments to the notion of neural networks in section 2.1. We still consider neural networks for $l \in \mathbb{N}$ and $d_0, \dots, d_l \in \mathbb{N}$ as functions

$$f : \mathbb{R}^{d_0} \to \mathbb{R}^{d_l} : x \mapsto f_l \circ f_{l-1} \circ \cdots \circ f_1(x),$$

where each f_i for $i = 1, \dots, l$ is a layer with input dimension d_{i-1} and output dimension d_i. However, in contrast to the notation in section 2.1 we denote these layers as functions

$$f_i : \mathbb{R}^{d_{i-1}} \to \mathbb{R}^{d_i} : x \mapsto \sigma\Big(\frac{1}{\sqrt{d_{i-1}}} W^{(i-1)} x + \beta b^{(i-1)}\Big),$$

where the activation function σ is applied entrywise. This notation arises by formulating definition 2.2 in terms of vectors and matrices, and introducing the additional factors $1/\sqrt{d_{i-1}}$ and $\beta > 0$. This procedure is important for later results.

Definition 3.25. *Let $T \subset \mathbb{R}$ be an index set. A family of random variables $\{X_t \mid t \in T\}$ is called a Gaussian process, if and only if for every finite set $\{t_1, \dots, t_k\} \subset T$, the random variable $(X_{t_1}, \dots, X_{t_k})$ is multivariate Gaussian distributed.*

Lemma 3.26. *Let $f : \mathbb{R}^{d_0} \to \mathbb{R}^{d_l}$ be a neural network of depth l, with a Lipschitz continuous non-linear activation function $\sigma : \mathbb{R} \to \mathbb{R}$ at each neuron. In the limit as $d_1, \dots, d_{l-1} \to \infty$ sequentially, the output functions $f_i(x, w)$ for $i = 1, \dots, d_l$, tend to i.i.d. centered Gaussian processes of covariance $\Sigma^{(l)}$, where $\Sigma^{(l)}$ is defined recursively by*

$$\Sigma^{(l)}(x, x') = \frac{1}{d_0} x^T x' + \beta^2$$
$$\Sigma^{(l+1)}(x, x') = \mathop{\mathbb{E}}_{f \sim \mathcal{N}(0, \Sigma^{(l)})} \Big[\sigma\big(f(x)\big) \sigma\big(f(x')\big) \Big] + \beta^2,$$

taking the expectation with respect to a centered Gaussian process f of covariance $\Sigma^{(l)}$.

Proof. **TO DO** □

Theorem 3.27. *Let $f : \mathbb{R}^{d_0} \to \mathbb{R}^{d_l}$ be a neural network of depth l, with a Lipschitz continuous non-linear activation function $\sigma : \mathbb{R} \to \mathbb{R}$ at each neuron. In the limit as $d_1, \dots, d_{l-1} \to \infty$ sequentially, the neural tangent kernel Θ_w converges in probability to a deterministic limiting kernel, this is*

$$\Theta_w \to \Theta_\infty \otimes I_{d_0}.$$

The scalar kernel $\Theta_\infty : \mathbb{R}^{d_0} \times \mathbb{R}^{d_0} \to \mathbb{R}$ is defined recursively by

$$\Theta_\infty^{(1)}(x, x') = \Sigma^{(1)}(x, x')$$
$$\Theta_\infty^{(l+1)}(x, x') = \Theta_\infty^{(l)}(x, x') \dot{\Sigma}^{(l+1)}(x, x') + \Sigma^{(l+1)}(x, x'),$$

where

$$\dot{\Sigma}^{(l+1)}(x, x') := \mathop{\mathbb{E}}_{f \sim \mathcal{N}(0, \Sigma^{(l)})} \Big[\sigma'\big(f(x)\big) \sigma'\big(f(x')\big) \Big],$$

taking the expectation with respect to a centered Gaussian process f of covariance $\Sigma^{(l)}$.

Proof. **TO DO** □

Theorem 3.28. *Assume that $\sigma : \mathbb{R} \to \mathbb{R}$ is a Lipschitz continuous and twice differentiable non-linear function with bounded second derivative. For any T such that the integral $\int_0^T \|d_t\|_{\mathcal{D}_x} \, dt$ stays stochastically bounded, as $d_1, \dots, d_{l-1} \to \infty$ sequentially, we have, uniformly for $t \in [0, T]$, that*

$$\Theta_w \to \Theta_\infty \otimes I_{d_l}.$$

As a consequence, in this limit, the dynamics of f_w is described by the differential equation

$$\partial_t f_{w(t)} = \Phi_{\Theta_\infty \otimes I_{d_l}} \Big(\big\langle d_t, \cdot \big\rangle_{\mathcal{D}_x} \Big).$$

Proof. **TO DO** □

Theorem 3.29. *Theorem 3.7 from [3]*

Proof. **TO DO** □

TO DO - Section 3.2
LEONARDO: Why is $\partial_f C|_{f_0}$ linear and continuous?
LEONARDO: Why does the first equality in proof of lemma 3.20 holds?

TO DO - Section 3.3
LEONARDO: Why does the first equality in proof of lemma 3.24 holds?
LEONARDO: Is the equivalence between the gradient methods clear?

TO DO
References (E-Mail + 3.7 wikipedia references)
Notation (double variables)
Notation 3.7: d instead of μ

TO DO MORE DETAILED
Completeness argument
Critical point of functionals
Convergence of kernel gradient descent
More general words on why we need theorems and definitions

TO DO POTENTIALLY
Proofs 3.29 - 3.31
Backpropagation
Convergence of functional gradient descent

TIPS
libgen.rs for sources

4 Dynamic Linear Dimensionality Reduction

5 Numerical Experiments

6 Conclusion

In conclusion...

7 Appendix

Definition 7.1. *A real vector space $\mathcal{H}$ is called a Hilbert space, if there exists an inner product $\langle\cdot,\cdot\rangle : \mathcal{H}\times\mathcal{H}\to\mathbb{R}$, such that H is complete with respect to the norm $\|\cdot\| := \sqrt{\langle\cdot,\cdot\rangle}$.*

Theorem 7.2. *Every Hilbert space $\mathcal{H}$ is isometric and isomorphic to its dual space $\mathcal{H}^*$ via*

$$J : \mathcal{H}\to\mathcal{H}^* : x\mapsto\langle x,\cdot\rangle.$$

Therefore each $\hat{x}\in\mathcal{H}^*$ can be represented as $\hat{x} = J(x)$ for some $x\in\mathcal{H}$.

Proof. Let $f, g\in\mathcal{F}$ and $\mu := J(f)$. By the Cauchy-Schwarz inequality it holds that

$$\big|\mu[g]\big| = \langle f, g\rangle \le \big\|f\big\|_{\mathcal{D}_x}\big\|g\big\|_{\mathcal{D}_x} \implies \big\|\mu\big\|_{\mathcal{F}^*} \le \big\|f\big\|_{D_x}.$$

Thus $\mu\in\mathcal{F}^*$ and therefore J is well defined. Furthermore it holds that

$$\big|\mu[f]\big| = \langle f, f\rangle = \big\|f\big\|^2_{\mathcal{D}_x} \implies \big\|\mu\big\|_{\mathcal{F}^*} \ge \big\|f\big\|_{D_x}.$$

The combination of both inequalities yields

$$\big\|J(f)\big\|_{\mathcal{F}^*} = \big\|\mu\big\|_{\mathcal{F}^*} = \big\|f\big\|_{D_x},$$

proving that J is indeed an isometry between $\mathcal{F}$ and its dual $\mathcal{F}^*$. Therefore J is injective and it only remains to prove the surjectivity. □

Corollary 7.3. *Let $f : \mathbb{R}^n\to\mathbb{R}$ be differentiable with global minimizer $x_*\in\mathbb{R}^n$, such that the gradient ∇f is Lipschitz continuous with Lipschitz constant $L > 0$, then*

$$\forall x\in\mathbb{R}^n : f(x) - f(x_*) \ge \frac{1}{2L}\big\|\nabla f(x)\big\|_2^2.$$

Proof. Let $x\in\mathbb{R}^n$ be arbitrary. Using $y = x - 1/L\cdot\nabla f(x)$ in lemma 2.8 yields

$$\begin{aligned} f(y) &\le f(x) + \big\langle\nabla f(x), -\frac{1}{L}\nabla f(x)\big\rangle + \frac{L}{2}\big\|\frac{1}{L}\nabla f(x)\big\|_2^2 \\ &= f(x) - \frac{1}{L}\big\|\nabla f(x)\big\|_2^2 + \frac{1}{2L}\big\|\nabla f(x)\big\|_2^2 \\ &= f(x) - \frac{1}{2L}\big\|\nabla f(x)\big\|_2^2. \end{aligned}$$

Taking the infimum on the lefthand side results in

$$f(x_*) = \inf_{y\in\mathbb{R}^n} f(y) \le f(x) - \frac{1}{2L}\big\|\nabla f(x)\big\|_2^2.$$

Rearranging the inequality completes the proof. □

Lemma 7.4. *Let $f : \mathbb{R}^n\to\mathbb{R}$ be differentiable, such that the gradient ∇f is Lipschitz continuous with Lipschitz constant $L > 0$, then*

$$\forall x, y\in\mathbb{R}^n : \big\langle\nabla f(y) - \nabla f(x), y - x\big\rangle \ge \frac{1}{L}\big\|\nabla f(y) - \nabla f(x)\big\|_2^2.$$

Proof. Let $x, y \in \mathbb{R}^n$ be arbitrary. Define the differentiable functions

$$g_x : \mathbb{R}^n \to \mathbb{R} : z \mapsto f(z) - \big\langle \nabla f(x), z \big\rangle,$$
$$g_y : \mathbb{R}^n \to \mathbb{R} : z \mapsto f(z) - \big\langle \nabla f(y), z \big\rangle,$$

then g_x is minimized by x and g_y is minimized by y since the gradients are given by

$$\nabla g_x(z) = \nabla f(z) - \nabla f(x),$$
$$\nabla g_y(z) = \nabla f(z) - \nabla f(y).$$

Thus we can apply corollary 7.3 to g_x and derive

$$f(y) - f(x) - \big\langle \nabla f(x), y - x \big\rangle = g_x(y) - g_x(x) \geq \frac{1}{2L} \big\| \nabla g_x(y) \big\|_2^2.$$

Similarly, applying corollary 7.3 to g_y results in

$$f(x) - f(y) - \big\langle \nabla f(y), x - y \big\rangle = g_y(x) - g_y(y) \geq \frac{1}{2L} \big\| \nabla g_y(x) \big\|_2^2.$$

Since $\big\| \nabla g_x(y) \big\|_2^2 = \big\| \nabla f(y) - \nabla f(x) \big\|_2^2 = \big\| \nabla g_y(x) \big\|_2^2$, summation of the inequalities yields

$$\big\langle \nabla f(y) - \nabla f(x), y - x \big\rangle \geq \frac{1}{L} \big\| \nabla f(y) - \nabla f(x) \big\|_2^2,$$

where we have used $-\big\langle \nabla f(y), x - y \big\rangle = \big\langle \nabla f(y), y - x \big\rangle$. This completes the proof. □

Theorem 7.5. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex and differentiable. Under the assumption that ∇f is Lipschitz continuous with Lipschitz constant $L > 0$, the gradient descent algorithm with $\epsilon = 0$ and $\alpha_k = 1/L$ for every $k \in \mathbb{N}_0$ produces a sequence $(x_k)_{k \in \mathbb{N}_0} \in \mathbb{R}^n$, such that*

$$\forall k \in \mathbb{N} : f(x_k) - f(x_*) \leq \frac{2L \big\| x_0 - x_* \big\|_2^2}{k}.$$

Proof. Let $k \in \mathbb{N}$ be arbitrary. Applying lemma 7.4 to x_k and x_* provides

$$\big\langle \nabla f(x_k) - \nabla f(x_*), x_k - x_* \big\rangle \geq \frac{1}{L} \big\| \nabla f(x_k) - \nabla f(x_*) \big\|_2^2.$$

Using $\nabla f(x_*) = 0$ and the symmetry of the scalar product, this is equivalent to

$$\big\langle x_k - x_*, \nabla f(x_k) \big\rangle \geq \frac{1}{L} \big\| \nabla f(x_k) \big\|_2^2.$$

Together with $x_{k+1} = x_k - 1/L \cdot \nabla f(x_k)$ by gradient descent, this can be used to show

$$\begin{aligned} \big\| x_{k+1} - x_* \big\|_2^2 &= \big\| x_k - x_* - \frac{1}{L} \nabla f(x_k) \big\|_2^2 \\ &= \big\| x_k - x_* \big\|_2^2 - 2 \frac{1}{L} \big\langle x_k - x_*, \nabla f(x_k) \big\rangle + \frac{1}{L^2} \big\| \nabla f(x_k) \big\|_2^2 \\ &\leq \big\| x_k - x_* \big\|_2^2 - \frac{2}{L} \cdot \frac{1}{L} \big\| \nabla f(x_k) \big\|_2^2 + \frac{1}{L^2} \big\| \nabla f(x_k) \big\|_2^2 \\ &= \big\| x_k - x_* \big\|_2^2 - \frac{1}{L^2} \big\| \nabla f(x_k) \big\|_2^2, \end{aligned} \tag{1}$$

which implies $\|x_k - x_*\|_2^2$ is monotonically decreasing in k. Furthermore lemma 2.8 yields

$$f(x_{k+1}) \leq f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k \rangle + \frac{L}{2}\|x_{k+1} - x_k\|_2^2.$$

By the definition of gradient descent, we can plug in $x_{k+1} - x_k = -1/L \cdot \nabla f(x_k)$ to derive

$$\begin{aligned} f(x_{k+1}) &\leq f(x_k) + \langle \nabla f(x_k), -\frac{1}{L}\nabla f(x_k) \rangle + \frac{L}{2}\|\frac{1}{L}\nabla f(x_k)\|_2^2 \\ &= f(x_k) - \frac{1}{L}\|\nabla f(x_k)\|_2^2 + \frac{1}{2L}\|\nabla f(x_k)\|_2^2 \\ &= f(x_k) - \frac{1}{2L}\|\nabla f(x_k)\|_2^2. \end{aligned} \tag{2}$$

Hence the gradient descent algorithm with constant step size $\alpha_k = 1/L$ guarantees to make progress unless $\nabla f(x_k) \neq 0$. On top of this we can infer by the convexity of f, that

$$f(x_k) - f(x_*) \leq \langle \nabla f(x_k), x_k - x_* \rangle.$$

Using the Cauchy-Schwarz inequality and the monotonicity of $\|x_k - x_*\|_2^2$ by (1) yields

$$f(x_k) - f(x_*) \leq \langle \nabla f(x_k), x_k - x_* \rangle \leq \|\nabla f(x_k)\|_2 \cdot \|x_k - x_*\|_2 \leq \|\nabla f(x_k)\|_2 \cdot \|x_0 - x_*\|_2,$$

which is equivalent to

$$\|\nabla f(x_k)\|_2 \geq \frac{1}{\|x_0 - x_*\|_2}\big(f(x_k) - f(x_*)\big). \tag{3}$$

Inserting the bound (3) in (2) results in

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2L}\frac{1}{\|x_0 - x_*\|_2^2}\big(f(x_k) - f(x_*)\big)^2.$$

Subtracting $f(x_*)$ on both sides and using the notation $\Delta_k := f(x_k) - f(x_*)$ yields

$$\Delta_{k+1} \leq \Delta_k - \frac{1}{2L}\frac{1}{\|x_0 - x_*\|_2^2}\Delta_k^2 \;\Leftrightarrow\; \frac{1}{2L}\frac{1}{\|x_0 - x_*\|_2^2}\Delta_k^2 \leq \Delta_k - \Delta_{k+1}.$$

Since Δ_k is monotonically decreasing in k by (2), we derive $1 \leq \Delta_k/\Delta_{k+1}$. Expanding the inequality with the positive factor $1/\Delta_k\Delta_{k+1}$ therefore provides

$$\frac{1}{2L}\frac{1}{\|x_0 - x_*\|_2^2} \leq \frac{1}{2L}\frac{1}{\|x_0 - x_*\|_2^2} \cdot \frac{\Delta_k}{\Delta_{k+1}} \leq \frac{1}{\Delta_{k+1}} - \frac{1}{\Delta_k}.$$

Summing up both sides from $i = 0, \ldots, k-1$ yields

$$k \cdot \frac{1}{2L}\frac{1}{\|x_0 - x_*\|_2^2} \leq \sum_{i=0}^{k-1}\left[\frac{1}{\Delta_{k+1}} - \frac{1}{\Delta_k}\right] = \frac{1}{\Delta_k} - \frac{1}{\Delta_0} \leq \frac{1}{\Delta_k}.$$

Finally we can use $\Delta_k = f(x_k) - f(x_*)$ to conclude

$$\Delta_k = f(x_k) - f(x_*) \leq \frac{2L\|x_0 - x_*\|_2^2}{k}.$$

This completes the proof. □

Problem: Maxima in lemma 7.4

References

[1] Tao Li, Lei Tan, Qinghua Tao, Yipeng Liu, Xiaolin Huang. Low Dimensional Landscape Hypothesis is True: DNNs can be Trained in Tiny Subspaces. arXiv preprint arXiv:2103.11154, 2021.

[2] Arthur Jacot, Franck Gabriel, Clément Holger. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. Advances in Neural Information Processing Systems, volume 31, pages 8571-8580, 2018.

[3] Zhou Fan, Zhichao Wang. Spectra of the Conjugate Kernel and Neural Tangent Kernel for Linear-Width Neural Networks. Advances in Neural Information Processing Systems, volume 33, pages 7710-7721, 2020.