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 $n_x \in \mathbb{N}$, $\mathsf{w} \in \mathbb{R}^{n_x}$, $\mathsf{b} \in \mathbb{R}$ and $\sigma : \mathbb{R} \to \mathbb{R}$. A neuron is defined as*

$$\nu : \mathbb{R}^{n_x} \to \mathbb{R} : x \mapsto \sigma\Big(\mathsf{w}^\mathsf{T} x + \mathsf{b}\Big).$$

In this notion σ is called the activation function, w the weights and b the bias of ν.

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

Definition 2.2. *Let $n_x, n_y \in \mathbb{N}$, $W = [\mathsf{w}_1, \dots, \mathsf{w}_{n_y}] \in \mathbb{R}^{n_x \times n_y}$, $\mathsf{b} \in \mathbb{R}^{n_y}$ and $\sigma : \mathbb{R} \to \mathbb{R}$. Given neurons ν_i with weights w_i, bias b_i and activation function σ for $i = 1, \dots, n_y$, we define a layer as*

$$f : \mathbb{R}^{n_x} \to \mathbb{R}^{n_y} : x \mapsto \begin{bmatrix} \nu_1(x) \\ \vdots \\ \nu_{n_y}(x) \end{bmatrix} = \sigma\Big(W^\mathsf{T} x + \mathsf{b}\Big),$$

where the activation function σ is applied element-wise.

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

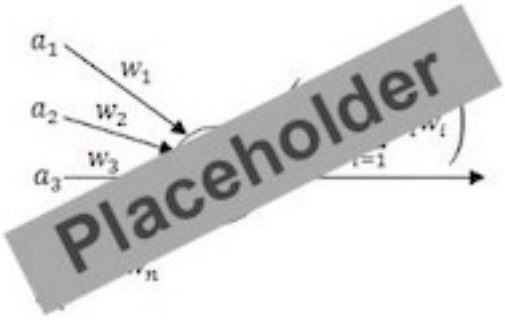

Figure 1: Illustration of a neuron as graph.

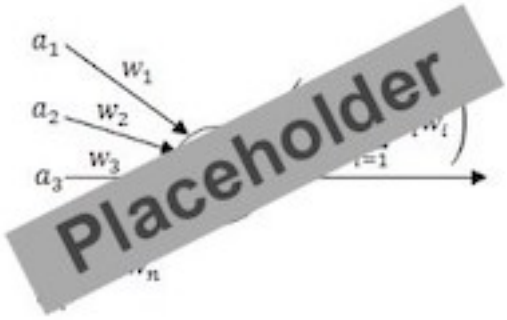

Figure 2: Illustration of a layer as graph.

In the following we will no longer distinguish between weights and biases and simply refer to them as parameters. 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 $n_0, \dots, n_l \in \mathbb{N}$. A neural network is defined as a function*

$$f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l} : x \mapsto f^{(l)} \circ f^{(l-1)} \circ \cdots \circ f^{(1)}(x),$$

where $f^{(i)}$ for $i = 1, \dots, l$ is a layer with input dimension n_{i-1} and output dimension n_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.

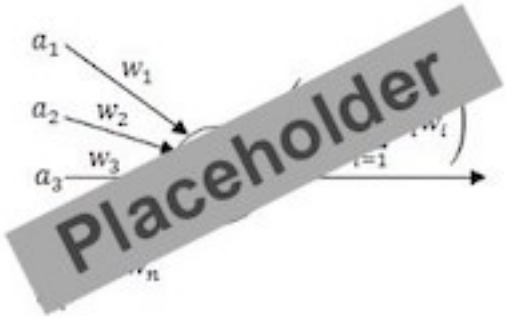

Figure 3: Illustration of a neural network as graph.

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 parameters and then iteratively adjust the parameters in order to approximate the unknown target function. In the notion of definition 2.3 one can determine the total number parameters in f as

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

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

$$\tilde{f} : \mathbb{R}^{n_0} \times \mathbb{R}^n \to \mathbb{R}^{n_l} : (x, w) \mapsto \tilde{f}(x, w),$$

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 parameters. 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, $n_0 \in \mathbb{N}$ will denote the dimension of the input space, $n_l \in \mathbb{N}$ will denote the dimension of the output space and $n \in \mathbb{N}$ will denote the number of adjustable parameters.

2.2 Training in the Parameter Space

Training a neural network refers to finding the optimal parameters, given a fixed neural network architecture $\tilde{f} : \mathbb{R}^{n_0} \times \mathbb{R}^n \to \mathbb{R}^{n_l}$. In other words, finding a function $f \in \mathcal{F}|_{\tilde{f}}$, where

$$\mathcal{F}|_{\tilde{f}} \coloneqq \Big\{ f_w : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l} : x \mapsto \tilde{f}(x, w) \mid w \in \mathbb{R}^n \Big\}$$

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

$$\ell : \mathbb{R}^{n_l} \times \mathbb{R}^{n_l} \to \mathbb{R}_+ : \big(f(x, w), y\big) \mapsto \ell\big(f(x, w), y\big),$$

that should be chosen in such a way, that for some input $x \in \mathbb{R}^{n_0}$ with target output $y \in \mathbb{R}^{n_l}$ 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 $\mathbb{R}^{n_0} \times \mathbb{R}^{n_l}$, describing the correlation between inputs $x \in \mathbb{R}^{n_0}$ and target values $y \in \mathbb{R}^{n_l}$, we call a parameter vector $w_* \in \mathbb{R}^n$ to be optimal, if

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

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

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

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 a set of $m \in \mathbb{N}$ labeled samples

$$\Big\{ (x_i, y_i) \in \mathbb{R}^{n_0} \times \mathbb{R}^{n_l} \mid i = 1, \dots, m \Big\},$$

which is called training data. Instead of working with the generalization error, the training data enables us to minimize the empirical error function

$$E : \mathbb{R}^n \to \mathbb{R}_+ : w \mapsto \frac{1}{m} \sum_{i=1}^{m} \ell\big(f(x_i, w), y_i\big) + \lambda \|w\|_2^2,$$

where $\lambda \geqslant 0$ 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 parameter vector w that minimizes the empirical error function is usually hard to solve as the error may have highly nonlinear dependencies on the parameters. 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*

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

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, which was first raised by Cauchy in [2].

Algorithm 1 Gradient Descent

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

2: Initialize $k \leftarrow 0$.

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

4: Choose a step size $\eta_k > 0$.

5: Set $x_{k+1} \leftarrow x_k - \eta_k \cdot \nabla f(x_k)$ and $k \leftarrow k + 1$.

6: **end while**

We will see, that in the case where f is bounded from below and ∇f is Lipschitz continuous, we can guarantee the convergence 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 $L \geqslant 0$, such that*

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

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 $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable. A stationary point of f is an element of*

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

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*

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

To prove convergence of the gradient descent algorithm for bounded f with Lipschitz continuous gradient, we need the following result which is formulated as lemma 1.2.3 in [3].

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

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

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

$$\psi : \mathbb{R} \to \mathbb{R} : \lambda \mapsto f(z_\lambda).$$

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

$$\psi' : \mathbb{R} \to \mathbb{R} : \lambda \mapsto (x' - x)^\mathsf{T} \cdot \nabla f(z_\lambda).$$

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

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

This equality can be rewritten as

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

Using the Cauchy-Schwarz inequality we derive

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

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

$$\left\| \nabla f(z_\lambda) - \nabla f(x) \right\|_2 \leqslant L \cdot \left\| z_\lambda - x \right\|_2 = L \cdot \left\| \lambda(x' - x) \right\|_2 = L\lambda \cdot \left\| x' - x \right\|_2$$

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

$$\left| f(x') - f(x) - \left\langle \nabla f(x), x' - x \right\rangle \right| \leqslant \int_0^1 L\lambda \cdot \left\| x' - x \right\|_2^2 d\lambda = \frac{L}{2}\|x' - x\|_2^2.$$

Rearranging the inequality finally yields

$$f(x') \leqslant f(x) + \left\langle \nabla f(x), x' - x \right\rangle + \frac{L}{2}\|x' - x\|_2^2.$$

This completes the proof, since x, x' 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 $\eta_k = 1/L$ produces a sequence $(x_k)_{k=0}^{\infty}$ in $\mathbb{R}^n$, such that*

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

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

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

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

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

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

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

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

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

Using $f(x_*) \leqslant f(x_k)$ for every $k \in \mathbb{N}_0$, we conclude

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

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

The theorem can be found as equation (1.2.15) in the beginning of section 1.2.3 of [3]. 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 $x \in \mathbb{R}^n$ with $\nabla f(x) = 0$. 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*

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

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*

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

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

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

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. Mathematically this can be formulated as follows.

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

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

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

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

which is non-negative on the interval $[0,1]$ by convexity of f. Denoting the derivative as

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

and using the non-negativity of ψ together with $\psi(0) = 0$, we observe that

$$\psi'(0) = f(x') - f(x) - (x' - x)^{\mathsf{T}} \cdot \nabla f(x) \geqslant 0.$$

Since x was chosen such that $\nabla f(x) = 0$, we can rearrange terms to conclude

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

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 empirical error function used for training neural networks, but we will introduce a theoretical approach to prevent this problem in section 3. Next we will provide further iterative optimization methods.

The gradient descent method is based on the first order Taylor expansion. Intuitively, inclusion of second order information should speed up the minimization process. This idea lead to the famous Newton's method, which requires the objective function to be twice differentiable. Furthermore, to apply Newton's method, we have to compute the Hessian matrix, leading to way more computation effort needed in comparison to gradient descent. In order to reduce this additional effort, one can approximate the inverse Hessian matrix instead of computing it exactly. One popular method, that pursues this approach is the following from [4].

Algorithm 2 Broyden-Fletcher-Goldfarb-Shanno (BFGS)

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

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

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

4: Choose a step size $\eta_k > 0$.

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

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

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

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

9: **end while**

With the initialization $B_0 = I_n$, the method starts with a simple gradient descent update and then iteratively minimizes the objective function by inverse Hessian matrix approximations.

Unfortunately, Hessian matrix computation or Hessian matrix approximation is infeasible for deep neural networks due to the large number of parameters. However, later on we will introduce an approach to train neural networks in tiny subspaces, such that second order methods become applicable.

At this stage we are restricted to training neural networks via gradient descent on the empirical error function. Recalling its definition

$$E : \mathbb{R}^n \to \mathbb{R}_+ : w \mapsto \frac{1}{m} \sum_{i=1}^{m} \ell\big(f(x_i, w), y_i\big) + \lambda \|w\|_2^2,$$

we observe, that the computation effort needed to compute the gradient, depends on the number of training samples $m \in \mathbb{N}$. 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 in a reasonable period of time. However, there exist several other optimization methods, that apply the same concept of minimizing a function along the slope of its surface. These algorithms make a trade-off between the accuracy and the time needed for gradient computations. In the following we will introduce two of them.

Stochastic gradient descent is a variation of gradient descent, that aims to approximate the gradient of the objective function instead of computing it exactly. 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 $\mathcal{I}_k \subset \{1, \dots, m\}$ of size $|\mathcal{I}_k| = b \in \mathbb{N}$ and uses

$$\nabla E(w_k, \mathcal{I}_k) \coloneqq \frac{1}{b} \sum_{i \in \mathcal{I}_k} \ell\big(f(x_i, w_k), y_i\big) + \lambda \|w_k\|_2^2$$

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. Formally stochastic gradient descent proceeds as follows.

Algorithm 3 Stochastic Gradient Descent (SGD)

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

2: Requires a batch size $b \in \mathbb{N}$ and a number of epochs $K \in \mathbb{N}$.

3: **for** $k = 1, \dots, \frac{Km}{b}$ **do**:

4: Draw a random subset $\mathcal{I}_k \subset \{1, \dots, m\}$ of size $|\mathcal{I}_k| = b$.

5: Choose a step size $\eta_k > 0$.

6: Set $w_{k+1} \leftarrow w_k - \eta_k \cdot \nabla E(w_k, \mathcal{I}_k)$.

7: **end for**

In here, an epoch describes the use of a number of samples for parameter updates, that equals the size of the whole training dataset. Thus on average each individual training sample will contribute K times to the optimization process. In comparison to this, the classical gradient descent method uses the data of one epoch per iteration. A theoretical convergence analysis of stochastic gradient descent can be found in [5], but would exceed the scope of this thesis.

So far we have not discussed how to choose the step sizes η_k. To simplify things, in practice one usually chooses a step size $\eta > 0$, 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 stochastic gradient descent algorithm by including past gradients for the current parameter update is called adaptive moment estimation and was first raised in [6].

Algorithm 4 Adaptive Moment Estimation (Adam)

Requires differentiable empirical error function $E : \mathbb{R}^n \to \mathbb{R}$ and random $w_0 \in \mathbb{R}^n$.
Requires constant step size $\eta > 0 \in \mathbb{R}$, batch size $b \in \mathbb{N}$ and number of epochs $K \in \mathbb{N}$.
Requires momentum factors $\beta_1, \beta_2 \in [0, 1)$ and a scalar $\epsilon > 0$.
for $k = 1, \ldots, \frac{Km}{b}$ **do**:
    Draw a random subset $\mathcal{I}_k \subset \{1, \ldots, m\}$ of size $|\mathcal{I}_k| = b$.
    Set $m_k \leftarrow \beta_1 m_{k-1} + (1 - \beta_1) \cdot \nabla E(w_k, \mathcal{I}_k)$ and let $\hat{m}_k \coloneqq \frac{m_k}{1-\beta_1^k}$.
    Set $v_k \leftarrow \beta_2 v_{k-1} + (1 - \beta_2) \cdot \nabla E(w_k, \mathcal{I}_k)^2$ and let $\hat{v}_k \coloneqq \frac{v_k}{1-\beta_2^k}$.
    Set $w_{k+1} \leftarrow w_k - \eta \cdot \frac{\hat{m}_k}{\sqrt{\hat{v}_k}+\epsilon}$.
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.

This completes the necessary basics on notation and training of neural networks. Next we will investigate certain approaches on how to improve the training of neural networks in order prevent the curse of non-convexity and reduce the computation and time effort needed.

Exploration: Further second order method
Optional: Backpropagation

3 Training in the Function Space

This section studies the training of neural networks in function space, instead of the parameter space $\mathbb{R}^n$. We will start with a motivation of this approach and introduce the concept of functional derivatives afterwards. Based on the concept of reproducing kernel Hilbert spaces, we can introduce the kernel gradient descent algorithm, which is a generalization of gradient descent for functionals. Finally, this algorithm leads to the definition of a special kernel, the neural tangent kernel, that is of great importance for the further theory. The results in here are mainly based on the work of Arthur Jacot, Franck Gabriel and Clément Hongler in [7].

3.1 Motivation

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

$$E : \mathbb{R}^n \to \mathbb{R}_+ : w \mapsto \frac{1}{m} \sum_{i=1}^{m} \ell\big(f(x_i, w), y_i\big) + \lambda \|w\|_2^2$$

on labeled training data $\{(x_i, y_i)\}_{i=1}^m \subset \mathbb{R}^{n_0} \times \mathbb{R}^{n_l}$. Due to the non-convexity of the error function E, it is in general hard to find a global minimum. However, for theoretical analysis, this problem can be prevented by moving from the parameter space $\mathbb{R}^n$ to the function space

$$\mathcal{F} \coloneqq \Big\{ f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l} \Big\}.$$

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

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

associating some cost to each element of the function space $\mathcal{F}$. The interpretation of ℓ as a convex function in f provides a convex cost C, since by lemma 2.12 the sum of convex functions is convex again. This is a huge advantage in comparison to the parameter optimization approach, where the convexity does not translates to E due to the composition of ℓ with f.

In the following we will introduce the kernel gradient descent algorithm, which is a generalization of gradient descent from the parameter space $\mathbb{R}^n$ to the function space $\mathcal{F}$. Since we can ensure convexity of the cost C in the function space, convergence of kernel gradient descent to a global minimum will be guaranteed.

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 kernel gradient descent to get a better understanding of gradient descent in the parameter space. That is why we aim to investigate the relationship between the two optimization approaches.

3.2 Functional Derivative

To start things off, we need the definition of functionals, mapping each element of the function space $\mathcal{F}$ to a real value. The cost C is an example of such a functional.

Definition 3.1. *A functional on the vector space $\mathcal{F}$ is defined as a mapping*

$$\mu : \mathcal{F} \to \mathbb{R} : f \mapsto \mu[f].$$

Minimizing functionals via the same approach in gradient descent, requires a translation of the well known concepts of differentiability and gradients from the parameter space $\mathbb{R}^n$ to the function space $\mathcal{F}$. The following definition is based on Appendix A of [8].

Definition 3.2. *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*

$$D_\phi \mu[f_0] \coloneqq \lim_{\epsilon \to 0} \frac{1}{\epsilon}\Big(\mu[f_0 + \epsilon\phi] - \mu[f_0]\Big) = \frac{\partial}{\partial \epsilon}\Big[\mu[f_0 + \epsilon\phi]\Big]_{\epsilon=0}$$

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

$$\partial_f \mu|_{f_0} : \mathcal{F} \to \mathbb{R} : \phi \mapsto D_\phi \mu[f_0]$$

and call μ differentiable at f_0.

This is similar to the definition of directional derivatives in $\mathbb{R}^{n_0}$. Likewise we can motivate the functional gradient in $\mathcal{F}$ based on the properties of the classical gradient in $\mathbb{R}^{n_0}$. To clarify this, we let $f \in \mathcal{F}$ and $x, v \in \mathbb{R}^{n_0}$, such that $\|v\|_2 = 1$ and consider the directional derivative of f in x along the direction v, which is given by

$$D_v f(x) \coloneqq \lim_{\epsilon \to 0} \frac{1}{\epsilon}\Big(f(x + \epsilon v) - f(x)\Big).$$

By Taylor's Theorem, we have for any $\epsilon > 0$, that

$$f(x + \epsilon v) = f(x) + \nabla f(x)^\mathsf{T}\big((x + \epsilon v) - x\big) + \mathcal{O}(\epsilon^2),$$

which can be rewritten as

$$f(x + \epsilon v) - f(x) = \epsilon\big\langle \nabla f(x), v\big\rangle + \mathcal{O}(\epsilon^2).$$

Dividing both sides by ϵ and taking the limit $\epsilon \to 0$ yields

$$D_v f(x) = \lim_{\epsilon \to 0} \frac{1}{\epsilon}\Big(f(x + \epsilon v) - f(x)\Big) = \big\langle \nabla f(x), v\big\rangle.$$

This gives us an approach of how to define the functional gradient in $\mathcal{F}$. Roughly speaking we could translate this concept and think of an element $\nabla \mu|_{f_0} \in \mathcal{F}$ as the functional gradient at $f_0 \in \mathcal{F}$ of some differentiable $\mu : \mathcal{F} \to \mathbb{R}$, if it satisfies

$$\forall \phi \in \mathcal{F} : D_\phi \mu[f_0] = \big\langle \nabla \mu|_{f_0}, \phi\big\rangle_{\mathcal{F}},$$

for a reasonable inner product $\langle \cdot, \cdot \rangle_{\mathcal{F}}$. It will turn out, that this is completely possible. We will start with the definition of an inner product and show the existence of functional gradients afterwards. The following lemma is based on [7].

Lemma 3.3. *Given a fixed probability distribution $\mathcal{D}_x$ on the input space $\mathbb{R}^{n_0}$, 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

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

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

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

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

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

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

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

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

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

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

Note that $\langle\cdot,\cdot\rangle_{\mathcal{D}_x}$ is not strictly positive definite. For example, in the case where $\mathcal{D}_x$ is a discrete distribution, any $f \in \mathcal{F}$ with $f(\operatorname{supp}(\mathcal{D}_x)) = 0$, satisfies $\langle f, f\rangle_{\mathcal{D}_x} = 0$. Since f could take arbitrary values for $x \notin \operatorname{supp}(\mathcal{D}_x)$, we can not conclude strict positive definiteness. However, in our setting, we only care about data points with a positive probability measure in terms of $\mathcal{D}_x$. That is why we can define an equivalence relation on $\mathcal{F}$, such that

$$\forall f, g \in \mathcal{F} \;:\; f \sim g \;\Leftrightarrow\; \langle f, \cdot\rangle_{\mathcal{D}_x} = \langle g, \cdot\rangle_{\mathcal{D}_x}.$$

Hence two elements are equivalent in terms of $\sim$, if and only if they are equal on the data. Roughly speaking, the restriction of $\langle\cdot,\cdot\rangle_{\mathcal{D}_x}$ to $\mathcal{F}/\!\sim$ is strictly positive definite, so

$$\langle\cdot,\cdot\rangle_{\mathcal{F}/\sim} : F/\!\sim \times F/\!\sim \to \mathbb{R} : \big([f], [g]\big) \mapsto \langle f, g\rangle_{\mathcal{D}_x}$$

is an inner product. We can prove completeness of $\mathcal{F}$ with respect to $\|\cdot\|_{\mathcal{D}_x}$ and therefore completeness of $\mathcal{F}/\!\sim$ with respect to $\|\cdot\|_{\mathcal{F}/\sim}$, such that $\big(F/\!\sim, \langle\cdot,\cdot\rangle_{\mathcal{F}/\sim}\big)$ is a Hilbert space. This is done by the following lemma.

Corollary 3.4. *The seminorm $\|\cdot\|_{\mathcal{D}_x}$ induces the pseudometric*

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

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

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

(i) $d_{\mathcal{F}}(f, f) = 0$:

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

(ii) $d_{\mathcal{F}}(f, g) = d_{\mathcal{F}}(g, f)$:

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

(iii) $d_{\mathcal{F}}(f, h) \leqslant d_{\mathcal{F}}(f, g) + d_{\mathcal{F}}(g, h)$:

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

The upper equations follow directly from the properties of the seminorm $\|\cdot\|_{\mathcal{D}_x}$.

It remains to show the completeness of $\mathcal{F}$, which arises from the fact, that every Cauchy sequence in $\mathbb{R}$ converges. To clarify this, we let $(f_n)_{n=1}^{\infty}$ denote a Cauchy sequence in $\mathcal{F}$, so

$$\forall \epsilon > 0 : \exists N \in \mathbb{N} : \forall m, n > N : d_{\mathcal{F}}(f_n, f_m)^2 < \epsilon. \tag{$*$}$$

Recalling the definition of $\|\cdot\|_{\mathcal{D}_x}$, we have

$$d_{\mathcal{F}}(f_n, f_m)^2 = \big\|f_n - f_m\big\|_{\mathcal{D}_x}^2 = \mathop{\mathbb{E}}_{x \sim \mathcal{D}_x}\Big[(f_n - f_m)(x)^{\mathsf{T}}(f_n - f_m)(x)\Big],$$

such that the inequality in $(*)$ translates to

$$d_{\mathcal{F}}(f_n, f_m)^2 = \mathop{\mathbb{E}}_{x \sim \mathcal{D}_x}\Big[\sum_{i=1}^{n_l}(f_n - f_m)_i(x)^2\Big] = \mathop{\mathbb{E}}_{x \sim \mathcal{D}_x}\Big[\sum_{i=1}^{n_l}\big((f_n)_i(x) - (f_m)_i(x)\big)^2\Big] < \epsilon.$$

Since all of the summands are non-negative, we observe for any $x \sim \mathcal{D}_x$, that

$$\forall \epsilon > 0 : \exists N \in \mathbb{N} : \forall m, n > N : \big|(f_n)_i(x) - (f_m)_i(x)\big| < \epsilon.$$

Hence $((f_n)_i(x))_{n=1}^{\infty}$ is a Cauchy sequence in $\mathbb{R}$ and therefore converges to a limit

$$\hat{f}_i(x) \coloneqq \lim_{n \to \infty} (f_n)_i(x) \in \mathbb{R}.$$

Equivalently this can be formulated as

$$\forall \epsilon > 0 : \exists N \in \mathbb{N} : \forall n > N : \big|(f_n)_i(x) - \hat{f}_i(x)\big| < \epsilon.$$

Since $\mathcal{F}$ contains all functions from $\mathbb{R}^{n_0}$ to $\mathbb{R}^{n_l}$, we can choose some $f_{lim} \in \mathcal{F}$, that satifies

$$\forall x \sim \mathcal{D}_x : f_{lim}(x) = \left[\hat{f}_1(x), \dots, \hat{f}_{n_l}(x)\right]^\mathsf{T}.$$

Again, using the definition of $\|\cdot\|_{\mathcal{D}_x}$, we observe for $n \in \mathbb{N}$, that

$$d_\mathcal{F}(f_n, f_{lim})^2 = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[\sum_{i=1}^{n_l}(f_n - f_{lim})_i(x)^2\Big] = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_x}\Big[\sum_{i=1}^{n_l}\big((f_n)_i(x) - (f_{lim})_i(x)\big)^2\Big].$$

Since for any $x \sim \mathcal{D}_x$ each sequence $((f_n)_i(x))_{n=1}^\infty$ converges, we finally conclude, that

$$\forall \epsilon > 0 : \exists N \in \mathbb{N} : \forall n > N : d_\mathcal{F}(f_n, f_{lim})^2 < \epsilon.$$

Hence the Cauchy sequence $(f_n)_{n=1}^\infty$ has a limit in $\mathcal{F}$, which makes $(\mathcal{F}, d_\mathcal{F})$ complete. □

Note that since $(\mathcal{F}, d_\mathcal{F})$ is only a pseudometric space and not a metric space, the limit f_{lim} constructed in the proof is not necessarily unique. Essentially any $f \in \mathcal{F}$, that suffices

$$\forall x \sim \mathcal{D}_x : f(x) = \left[\hat{f}_1(x), \dots, \hat{f}_{n_l}(x)\right]^\mathsf{T}$$

represents a limit function of the Cauchy sequence. This is due to the limit being measured in terms of the seminorm $\|\cdot\|_{\mathcal{D}_x}$, which depends only on the data according to $\mathcal{D}_x$. Namely it is no matter which values the limit function takes on points $x \notin \operatorname{supp}(\mathcal{D}_x)$. However, in the Hilbert space $\mathcal{F}/\!\sim$, the limit functions form an equivalence class, such that the limit in $\mathcal{F}/\!\sim$ is uniquely determined by its equivalence class.

Based on the notion of $\langle\cdot,\cdot\rangle_{\mathcal{D}_x}$, we can think of an element $\nabla\mu|_{f_0} \in \mathcal{F}$ as the functional gradient at $f_0 \in \mathcal{F}$ of some differentiable $\mu : \mathcal{F} \to \mathbb{R}$, if it satisfies

$$\forall \phi \in \mathcal{F} : D_\phi\mu[f_0] = \big\langle \nabla\mu|_{f_0}, \phi\big\rangle_{\mathcal{D}_x}.$$

Note that, due to the definition of $\langle\cdot,\cdot\rangle_{\mathcal{D}_x}$, the functional gradient is not unique and depends only on the support of $\mathcal{D}_x$. Essentially we could think of any other element $d \in \mathcal{F}$, that is equivalent to $\nabla\mu|_{f_0}$ in terms of $\sim$, as the functional gradient as well. We will discuss and address this problem later on, but first let us prove the existence of functional gradients.

We will see, that a functional gradient exists, if $\partial_f\mu|_{f_0}$ is linear and continuous. For this we denote the class of linear and continuous functionals on $\mathcal{F}$, namely the dual space of $\mathcal{F}$.

Definition 3.5. *Given a real vector space $\mathcal{F}$, we define the dual space*

$$\mathcal{F}^* \coloneqq \Big\{\mu : \mathcal{F} \to \mathbb{R} \mid \mu \textit{ linear and continuous}\Big\}.$$

To prove the existence of functional gradients, we will construct a surjective mapping

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

such that we can conclude the existence based on $\partial_f\mu|_{f_0} \in \mathcal{F}^*$. For the construction of J we have to translate some well known results from functional analysis to our setting.

Based on the completeness of $(\mathcal{F}, d_{\mathcal{F}})$, we can introduce the concept of orthogonal projections, which is necessary for our goal of establishing a mapping between $\mathcal{F}$ and its dual space $\mathcal{F}^*$. The following lemma is a translation of Theorem 3.3.1 in [9] to our case, where we have a symmetric positive semidefinite bilinear form $\langle \cdot, \cdot \rangle_{\mathcal{D}_x}$ instead of an inner product.

Lemma 3.6. *Let $\mathcal{A} \subseteq \mathcal{F}$ be non-empty, closed and convex. For every $f \in \mathcal{F}$ there exists some $a^* \in \mathcal{A}$ such that*

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

Each $a^* \in \mathcal{A}$ that suffices this property, is called an orthogonal projection of f on $\mathcal{A}$.

Proof. By definition of the infimum there exists a sequence $(a_n)_{n=1}^{\infty}$ in $\mathcal{A}$, such that

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

We show that $(a_n)_{n=1}^{\infty}$ is a Cauchy sequence. For this we let $n, m \in \mathbb{N}$ and observe that

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

Furthermore we have

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

Combining both expressions yields

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

By the definition of δ we observe that

$$\forall \epsilon > 0 : \exists N \in \mathbb{N} : \forall n > N : \|f - a_n\|_{\mathcal{D}_x}^2 < \delta + \epsilon.$$

Furthermore, since $\mathcal{A}$ is convex, $\frac{1}{2}(a_n + a_m) \in \mathcal{A}$ and we derive

$$\|f - \frac{1}{2}(a_n + a_m)\|_{\mathcal{D}_x}^2 \geqslant \inf_{a \in \mathcal{A}} \|f - a\|_{\mathcal{D}_x}^2 = \delta.$$

Using these two observations, we conclude that

$$\forall \epsilon > 0 : \exists N \in \mathbb{N} : \forall n > N : \|a_n - a_m\|_{\mathcal{D}_x}^2 \leqslant 2(\delta + \epsilon) + 2(\delta + \epsilon) - 4\delta = 4\epsilon$$

Thus $(a_n)_{n=1}^{\infty}$ is a Cauchy sequence. Since $\mathcal{F}$ is complete and $\mathcal{A}$ is closed, $(a_n)_{n=1}^{\infty}$ converges to some $a^* \in \mathcal{A}$ with $\|f - a^*\|_{\mathcal{D}_x}^2 = \delta$, which proofs the lemma. □

Note that since $\|\cdot\|_{\mathcal{D}_x}$ is only a seminorm and not a norm, the limit of $(a_n)_{n=1}^{\infty}$ is not unique. Essentially there can exist multiple projections from f onto the subset $\mathcal{A}$. Nevertheless, these orthogonal projections form an equivalence, such that we have uniqueness in $\mathcal{F}/\sim$.

Lemma 3.6 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, because $\langle \cdot, \cdot \rangle_{\mathcal{D}_x}$ is just positive semidefinite. However this is just a side note, since the existence of an orthogonal projection suffices for the further theory. Next we infer a useful corollary, which is notated as lemma 3.3.2 in [9].

Corollary 3.7. *Let $f \in \mathcal{F}$ and $\mathcal{A} \subseteq \mathcal{F}$ 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, $a^* + \lambda a \in \mathcal{A}$ and we derive

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

Rearranging the inequality yields

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

giving rise to the definition of the non-negative function

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

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

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

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

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

which is a contradiction to $\psi(\lambda) \geqslant 0$ 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 construct any $\mu \in \mathcal{F}^*$ based on a representer $d \in \mathcal{F}$. The following lemma is a translation of theorem 3.8.1 in [9] to our setting.

Theorem 3.8. *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 $N \coloneqq \mu^{-1}(0)$ denote the kernel of μ, which is non-empty, closed and convex since μ is linear and continuous. Thus by Lemma 3.6 there exists some $p_e \in N$, such that p_e is an orthogonal projection of e on N. Define $f \coloneqq e - p_e$, then $f \notin N$ since

$$\mu[f] = \mu[e] - \mu[p_e] = \mu[e] - 0 = 1.$$

Thus we can choose any $g \in \mathcal{F}$, define $h \coloneqq g - \mu[g]f$ and use the linearity of μ to derive

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

This implies $h \in N$, so that $\langle f, h\rangle_{\mathcal{D}_x} = \langle e - p_e, h\rangle_{\mathcal{D}_x} = 0$ by corollary 3.7 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

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

Hence we can conclude the existence of functional gradients, as long as the functional derivative $\partial_f \mu|_{f_0}$ is an element of $\mathcal{F}^*$. In other words, $\partial_f \mu|_{f_0}$ is linear and continuous. To guarantee this, for the rest of the thesis we make the restriction, that the marginal distribution $\mathcal{D}_x$ on the input space is given by the empirical distribution on a finite subset of $\mathbb{R}^{n_0}$. This means there exist $m \in \mathbb{N}$ and $x_1, \dots, x_m \in \mathbb{R}^{n_0}$, such that the distribution of inputs is given by

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

where δ_{x_i} denotes the Dirac measure in x_i. Under this assumption, $\langle \cdot, \cdot\rangle_{\mathcal{D}_x}$ depends only on the values of f at a finite dataset. This implies the following result from [7].

Lemma 3.9. *Let $\mu : \mathcal{F} \to \mathbb{R}$ be differentiable in $f_0 \in \mathcal{F}$, then we have $\partial_f \mu|_{f_0} \in \mathcal{F}^*$. In other words, the functional derivative $\partial_f \mu|_{f_0}$ is linear and continuous.*

Proof. The prove is based on the linear operator

$$T : \mathcal{F}/\!\sim \to \mathbb{R} : [\phi] \mapsto \partial_f \mu|_{f_0}[\phi].$$

To prove that T is indeed linear, we need to show that $\partial_f \mu|_{f_0}$ is linear. Based on the linearity of the total derivative, we observe for any $\phi_1, \phi_2 \in \mathcal{F}$, that

$$\begin{aligned}
\partial_f \mu|_{f_0}[\phi_1 + \phi_2] &= \frac{\partial}{\partial \epsilon}\Big[\mu[f_0 + \epsilon\phi_1 + \epsilon\phi_2]\Big]_{\epsilon=0} \\
&= \frac{\partial}{\partial \epsilon}\Big[\mu[f_0 + \epsilon\phi_1]\Big]_{\epsilon=0} + \frac{\partial}{\partial \epsilon}\Big[\mu[f_0 + \epsilon\phi_2]\Big]_{\epsilon=0} \\
&= \partial_f \mu|_{f_0}[\phi_1] + \partial_f \mu|_{f_0}[\phi_2].
\end{aligned}$$

Furthermore we have for $\phi \in \mathcal{F}$ and $\lambda \in \mathbb{R}$, that

$$\partial_f \mu|_{f_0}[\lambda\phi] = \frac{\partial}{\partial \epsilon}\Big[\mu[f_0 + \epsilon\lambda\phi]\Big]_{\epsilon=0} = \lambda \cdot \frac{\partial}{\partial \epsilon}\Big[\mu[f_0 + \epsilon\phi]\Big]_{\epsilon=0} = \lambda \cdot \partial_f \mu|_{f_0}[\phi].$$

Hence T is linear. Using the assumption, that $\mathcal{D}_x$ is the empirical distribution on a finite subset of $\mathbb{R}^{n_0}$, we conclude, that $\mathcal{F}/\!\sim$ is finite-dimensional. Thus there exists $m \coloneqq \dim(F/\!\sim)$ and a basis $[e_1], \dots, [e_m] \in \mathcal{F}/\!\sim$, such that for any $[f] = \sum_{i=1}^{m} \alpha_i [e_i] \in \mathcal{F}/\!\sim$ we observe, that

$$\big|T[f]\big| = \Big|\sum_{i=1}^{m} \alpha_i T[e_i]\Big| \leqslant \sum_{i=1}^{m} \big|\alpha_i\big| \cdot \big|T[e_i]\big| \leqslant \max_{j=1,\dots,m} |\alpha_j| \cdot \sum_{i=1}^{m} \big|T[e_i]\big|.$$

Since $\max_{j=1,\dots,m} |\alpha_j|$ defines a norm on $\mathcal{F}/\!\sim$, based on the equivalence of norms in finite-dimensional spaces, we conclude the existence of a constant $c_1 \geqslant 0$, such that

$$\forall \alpha_1, \dots, \alpha_m \in \mathbb{R} : \max_{j=1,\dots,m} |\alpha_j| \leqslant c_1 \Big\| \sum_{i=1}^{m} \alpha_i [e_i] \Big\|_{\mathcal{F}/\sim}.$$

Using this and another constant $c_2 \coloneqq c_1 \sum_{i=1}^{m} \big|T[e_i]\big|$, we derive

$$\forall [f] \in F/\!\sim : \big|T[f]\big| \leqslant \max_{j=1,\dots,m} |\alpha_j| \cdot \sum_{i=1}^{m} \big|T[e_i]\big| \leqslant c_2 \Big\| \sum_{i=1}^{m} \alpha_i [e_i] \Big\|_{\mathcal{F}/\sim} = c_2 \big\|[f]\big\|_{F/\sim}.$$

Thus the operator T is bounded. In combination with the linearity of T, we conclude

$$\forall [f], [h] \in \mathcal{F}/\!\sim : \Big|T\big([f] + [h]\big) - T\big([f]\big)\Big| = \Big|T\big([h]\big)\Big| \leqslant c \big\|[h]\big\|_{F/\sim}$$

for some constant $c \geqslant 0$. Taking the limit $[h] \to 0$ yields the continuity of T at $[f]$. Thus T is linear and continuous, which implies, that $\partial_f \mu|_{f_0}$ is indeed linear and continuous. □

Thus the functional derivative $\partial_f \mu|_{f_0}$ can be viewed as an element of $\mathcal{F}^*$. According to theorem 3.8 we conclude the existence of a functional gradient $\nabla \mu|_{f_0}$, such that

$$\forall \phi \in \mathcal{F} : \big\langle \nabla \mu|_{f_0}, \phi \big\rangle_{\mathcal{D}_x}.$$

Theorem 3.8 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 $J : \mathcal{H} \to \mathcal{H}^* : x \mapsto \langle x, \cdot \rangle_{\mathcal{H}}$. However, 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. This is a severe problem for our approach of minimizing a functional via gradient descent, because the functional gradient is not uniquely determined. Namely any descent direction from the corresponding equivalence class according to $\sim$ could be chosen for the iterative updates, but we have no criterium to select one of them.

In terms of machine learning scenarios, this problem can be explained as follows. We assume that there exists an unknown probability distribution $\mathcal{D}_x$ and approximate it by the empirical distribution corresponding to our dataset. Then, we could minimize the functional cost C via the same iterative approach in gradient descent by choosing any element from the corresponding equivalence class of functional gradients. This would perfectly minimize C on the training dataset. However, we have no criterium to optimize our target function on unseen data points, which is the main objective of our machine learning task. Thus we have to find a solution, to generalize the functional derivative to values outside the dataset.

Kernel gradient descent is a method, that addresses this problem by minimizing the cost C in a special subspace of $\mathcal{F}$, namely a reproducing kernel Hilbert space, in which a unique descent direction can be constructed. This idea will be explained in detail in section 3.4, but first we need some background knowledge on kernels and reproducing kernel Hilbert spaces.

3.3 Reproducing Kernel Hilbert Space

Reproducing kernel Hilbert spaces are widely used in machine learning scenarios. In the scalar case, their main application is to reduce the computation effort needed or to simplify infinite-dimensional optimization problems to finite-dimensional ones via the so kernel trick. In here, we will study a generalization of this concept to vector-valued reproducing kernel Hilbert spaces. These are based on the definition of multi-dimensional kernels from [7].

Definition 3.10. *A multi-dimensional kernel over $\mathbb{R}^{n_0}$ is a measurable map*

$$K : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}^{n_l \times n_l} : (x, x') \mapsto K(x, x'),$$

such that $K(x, x') = K(x', x) = K(x, x')^\mathsf{T}$ for any $x, x' \in \mathbb{R}^{n_0}$.

Equivalently one could define a kernel K as a symmetric tensor in $\mathcal{F} \otimes \mathcal{F}$. Further on, we will always work with multi-dimensional kernels $K : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}^{n_l \times n_l}$ and for reasons of simplicity just call them kernels. We are especially interested in positive semidefinite kernels, as they naturally induce vector-valued reproducing kernel Hilbert spaces.

Definition 3.11. *We say that a kernel K is positive semidefinite, if it suffices*

$$\forall m \in \mathbb{N} : \forall x_1, \dots, x_m \in \mathbb{R}^{n_0} : \forall y_1, \dots, y_m \in \mathbb{R}^{n_l} : \sum_{i=1}^{m} \sum_{j=1}^{m} y_i^\mathsf{T} K(x_i, x_j) y_j \geqslant 0.$$

Based on this definition, we can formulate the following theorem according to [10].

Theorem 3.12. *Let K be a positive semidefinite kernel and define*

$$\mathcal{H}_0 \coloneqq span\Big\{ K_x y : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l} : z \mapsto K(z, x) y \mid x \in \mathbb{R}^{n_0}, y \in \mathbb{R}^{n_l} \Big\},$$

then there exist an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_K}$, such that the closure $\mathcal{H}_K \coloneqq \overline{H_0}$ is a Hilbert space. We call $\big(\mathcal{H}_K, \langle \cdot, \cdot \rangle_{\mathcal{H}_K}\big)$ the vector-valued reproducing kernel Hilbert space associated with K.

Proof. The idea of the proof is to construct an inner product on $\mathcal{H}_0$, and afterwards complete the space by adding all limits of Cauchy sequences in $\mathcal{H}_0$. We will denote $f, g, h \in \mathcal{H}_K$ as

$$f = \sum_i K_{x_i} y_i \quad \wedge \quad g = \sum_j K_{\hat{x}_j} \hat{y}_j \quad \wedge \quad h = \sum_k K_{\bar{x}_k} \bar{y}_k.$$

Based on this notation we can define the inner product

$$\langle \cdot, \cdot \rangle_{\mathcal{H}_0} : \mathcal{H}_0 \times \mathcal{H}_0 \to \mathbb{R} : (f, g) \mapsto \sum_{i,j} y_i^\mathsf{T} K(x_i, \hat{x}_j) \hat{y}_j.$$

To prove that $\langle \cdot, \cdot \rangle_{\mathcal{H}_0}$ is indeed an inner product, we start with linearity and symmetry.

(i) $\langle f + g, h \rangle_{\mathcal{H}_0} = \langle f, h \rangle_{\mathcal{H}_0} + \langle g, h \rangle_{\mathcal{H}_0}$:

$$\langle f + g, h \rangle_{\mathcal{H}_0} = \sum_{i,k} y_i^\mathsf{T} K(x_i, \bar{x}_k) \bar{y}_k + \sum_{j,k} \hat{y}_j^\mathsf{T} K(\hat{x}_j, \bar{x}_k) \bar{y}_k = \langle f, h \rangle_{\mathcal{H}_0} + \langle g, h \rangle_{\mathcal{H}_0},$$

(ii) $\langle \lambda f, g \rangle_{\mathcal{H}_0} = \lambda \langle f, g \rangle_{\mathcal{H}_0}$:

$$\langle \lambda f, g \rangle_{\mathcal{H}_0} = \sum_{i,j} (\lambda y_i)^\mathsf{T} K(x_i, \hat{x}_j) \hat{y}_j = \lambda \cdot \sum_{i,j} y_i^\mathsf{T} K(x_i, \hat{x}_j) \hat{y}_j = \lambda \langle f, g \rangle_{\mathcal{H}_0},$$

(iii) $\langle f, g \rangle_{\mathcal{H}_0} = \langle g, f \rangle_{\mathcal{H}_0}$:

$$\langle f, g \rangle_{\mathcal{H}_0} = \sum_{i,j} y_i^\mathsf{T} K(x_i, \hat{x}_j) \hat{y}_j = \sum_{i,j} y_i^\mathsf{T} K(\hat{x}_j, x_i)^\mathsf{T} \hat{y}_j = \sum_{i,j} \hat{y}_j^\mathsf{T} K(\hat{x}_j, x_i) y_i = \langle g, f \rangle_{\mathcal{H}_0}.$$

Next we need to show, that $\langle \cdot, \cdot \rangle_{\mathcal{H}_0}$ is strictly positive definite. To do this, it should be noted, that for any $x \in \mathbb{R}^{n_0}, y \in \mathbb{R}^{n_l}$ and $f \in \mathcal{H}_0$ the so called reproducing property holds. That is

$$\big\langle f(x), y \big\rangle = \sum_i \big(K(x, x_i) y_i \big)^\mathsf{T} y = \sum_{i,j} y_i^\mathsf{T} K(x_i, x) y = \big\langle f, K_x y \big\rangle_{\mathcal{H}_0}.$$

Moreover, since the kernel K is assumed to be positive semidefinite, we have

$$\forall f \in \mathcal{H}_0 : \langle f, f \rangle_{\mathcal{H}_0} = \sum_{i,j} y_i^\mathsf{T} K(x_i, x_j) y_j \geqslant 0.$$

To prove strict definiteness, we can apply the Cauchy-Schwarz inequality and derive

$$\big\langle f(x), y \big\rangle^2 = \big\langle f, K_x y \big\rangle_{\mathcal{H}_0}^2 \leqslant \big\langle f, f \big\rangle_{\mathcal{H}_0} \cdot \big\langle K_x y, K_x y \big\rangle_{\mathcal{H}_0}.$$

Again, using the Cauchy-Schwarz inequality, we observe that

$$\big\langle K_x y, K_x y \big\rangle_{\mathcal{H}_0} = y^T K(x, x) y = \big\langle y, K(x, x) y \big\rangle \leqslant \|y\|_2 \cdot \|K(x, x) y\|_2 \leqslant \|y\|_2^2 \cdot \|K(x, x)\|_2,$$

where $\|K(x, x)\|_2$ denotes the spectral norm of $K(x, x)$. In total, we can conclude

$$\big\langle f(x), y \big\rangle^2 \leqslant \big\langle f, f \big\rangle_{\mathcal{H}_0} \cdot \|y\|_2^2 \cdot \|K(x, x)\|_2.$$

Since x, y and f were chosen arbitrary, we conclude that

$$\forall f \in \mathcal{H}_0 \ : \ \langle f, f \rangle_{\mathcal{H}_0} = 0 \ \Leftrightarrow \ f \equiv 0.$$

In other words, $\langle \cdot, \cdot \rangle_{\mathcal{H}_0}$ is strictly positive definite and therefore an inner product.

It remains to prove, that the completion of $\mathcal{H}_0$ is a Hilbert space. Let $(f_n)_{n=1}^\infty$ be a Cauchy sequence in $\mathcal{H}_0$. For any $x \in \mathbb{R}^{n_0}$ we can apply the same argumentation as before, so that

$$\big\langle f_n(x) - f_m(x), y \big\rangle^2 \leqslant \|f_n - f_m\|_{\mathcal{H}_0} \cdot \|y\|_2^2 \cdot \|K(x, x)\|_2 \leqslant c \cdot \|f_n - f_m\|_{\mathcal{H}_0}$$

for a constant $c > 0$. Since $(f_n)_{n=1}^\infty$ was assumed to be Cauchy with respect to $\| \cdot \|_{\mathcal{H}_0}$, we conclude that $(f_n(x))_{n=1}^\infty$ is a Cauchy sequence in $\mathbb{R}^{n_l}$. Hence the limit

$$f_{lim} : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l} : x \mapsto \lim_{n \to \infty} f_n(x)$$

is well-defined. Now let $\mathcal{H}_K$ be the completion of $\mathcal{H}_0$ by adding all these limits of Cauchy sequences. Thus for any $f, g \in \mathcal{H}_K$ there exist sequences $(f_n)_{n=1}^\infty$ and $(g_n)_{n=1}^\infty$ in $\mathcal{H}_0$ with

$$f = \lim_{n \to \infty} f_n \quad \wedge \quad g = \lim_{n \to \infty} g_n.$$

Based on the continuity of the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_0}$, we are able to define the inner product

$$\langle \cdot, \cdot \rangle_{\mathcal{H}_K} : \mathcal{H}_K \times \mathcal{H}_K \to \mathbb{R} : (f, g) \mapsto \lim_{n \to \infty} \langle f_n, g_n \rangle_{\mathcal{H}_0}.$$

To ensure that $\langle \cdot, \cdot \rangle_{\mathcal{H}_K}$ is well-defined, we have to prove the existence of the limit. For this we let $n, m \in \mathbb{N}$ be arbitrary and apply the Cauchy-Schwarz inequality to derive

$$\begin{aligned}
\left| \langle f_n, g_n \rangle_{\mathcal{H}_0} - \langle f_m, g_m \rangle_{\mathcal{H}_0} \right| &= \left| \langle f_n, g_n \rangle_{\mathcal{H}_0} - \langle f_n, g_m \rangle_{\mathcal{H}_0} + \langle f_n, g_m \rangle_{\mathcal{H}_0} - \langle f_m, g_m \rangle_{\mathcal{H}_0} \right| \\
&= \left| \langle f_n, g_n - g_m \rangle_{\mathcal{H}_0} + \langle f_n - f_m, g_m \rangle_{\mathcal{H}_0} \right| \\
&\leqslant \left| \langle f_n, g_n - g_m \rangle_{\mathcal{H}_0} \right| + \left| \langle f_n - f_m, g_m \rangle_{\mathcal{H}_0} \right| \\
&\leqslant \|f_n\|_{\mathcal{H}_0} \|g_n - g_m\|_{\mathcal{H}_0} + \|f_n - f_m\|_{\mathcal{H}_0} \|g_m\|_{\mathcal{H}_0}.
\end{aligned}$$

Furthermore, we observe that

$$\left| \|f_n\|_{\mathcal{H}_0} - \|f_m\|_{\mathcal{H}_0} \right| \leqslant \|f_n - f_m\|_{\mathcal{H}_0}.$$

Since $(f_n)_{n=1}^{\infty}$ is a Cauchy sequence in $\mathcal{H}_0$, there exists a constant $c_f \geqslant 0$, such that

$$\lim_{n \to \infty} \|f_n\|_{\mathcal{H}_0} = c_f.$$

Using the same argumentation for $(g_n)_{n=1}^{\infty}$, we conclude the existence of $c_g \geqslant 0$, such that

$$\lim_{n \to \infty} \|g_n\|_{\mathcal{H}_0} = c_g.$$

Since $(f_n)_{n=1}^{\infty}$ and $(g_n)_{n=1}^{\infty}$ are Cauchy sequences in $\mathcal{H}_0$ we conclude, that $(\langle f_n, g_n \rangle_{\mathcal{H}_0})_{n=1}^{\infty}$ is a Cauchy sequence in $\mathbb{R}$. This implies well-definiteness of $\langle \cdot, \cdot \rangle_{\mathcal{H}_K}$.

It remains to prove that $\langle \cdot, \cdot \rangle_{\mathcal{H}_K}$ is indeed an inner product. We verify

(i) $\langle f + g, h \rangle_{\mathcal{H}_K} = \langle f, h \rangle_{\mathcal{H}_K} + \langle g, h \rangle_{\mathcal{H}_K}$:

$$\langle f + g, h \rangle_{\mathcal{H}_K} = \lim_{n \to \infty} \langle f_n, h_n \rangle_{\mathcal{H}_0} + \lim_{n \to \infty} \langle g_n, h_n \rangle_{\mathcal{H}_0} = \langle f, h \rangle_{\mathcal{H}_K} + \langle g, h \rangle_{\mathcal{H}_K},$$

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

$$\langle \lambda f, g \rangle_{\mathcal{H}_K} = \lim_{n \to \infty} \langle \lambda f_n, g_n \rangle_{\mathcal{H}_0} = \lambda \cdot \lim_{n \to \infty} \langle f_n, g_n \rangle_{\mathcal{H}_0} = \lambda \langle f, g \rangle_{\mathcal{H}_K},$$

(iii) $\langle f, g \rangle_{\mathcal{H}_K} = \langle g, f \rangle_{\mathcal{H}_K}$:

$$\langle f, g \rangle_{\mathcal{H}_K} = \lim_{n \to \infty} \langle f_n, g_n \rangle_{\mathcal{H}_0} = \lim_{n \to \infty} \langle g_n, f_n \rangle_{\mathcal{H}_0} = \langle g, f \rangle_{\mathcal{H}_K}.$$

Again, by Cauchy-Schwarz we have for any $x \in \mathbb{R}^{n_0}$, $y \in \mathbb{R}^{n_l}$ and $f \in \mathcal{H}_K$, that

$$\left\langle f(x), y \right\rangle^2 = \lim_{n \to \infty} \left\langle f_n(x), y \right\rangle^2 \leqslant \lim_{n \to \infty} \left\langle f_n, f_n \right\rangle_{\mathcal{H}_0} \cdot \|y\|_2^2 \cdot \|K(x, x)\|_2 = c \cdot \left\langle f, f \right\rangle_{\mathcal{H}_K}$$

for a constant $c \geqslant 0$. Hence $\langle \cdot, \cdot \rangle_{\mathcal{H}_K}$ is positive definite and therefore an inner product. □

The naming arises from the fact, that the following reproducing property holds.

Corollary 3.13. *For any $f \in \mathcal{H}_K$ the reproducing property holds, that is*

$$\forall x \in \mathbb{R}^{n_0}, y \in \mathbb{R}^{n_l} : \big\langle f(x), y \big\rangle = \big\langle f, K_x y \big\rangle_{\mathcal{H}_K}.$$

Proof. Let $x \in \mathbb{R}^{n_0}, y \in \mathbb{R}^{n_l}$ and $f \in \mathcal{H}_K$ be arbitrary. We investigate the following two cases.

Case $f \in \mathcal{H}_0$: We have already seen in the proof, of theorem 3.12, that

$$\big\langle f(x), y \big\rangle = \sum_i \big(K(x, x_i) y_i\big)^\mathsf{T} y = \sum_{i,j} y_i^\mathsf{T} K(x_i, x) y = \big\langle f, K_x y \big\rangle_{\mathcal{H}_K}.$$

Case $f \notin \mathcal{H}_0$: By construction of $\mathcal{H}_K$ there exists a sequence $(f_n)_{n=1}^\infty$ in $\mathcal{H}_0$, such that

$$\big\langle f(x), y \big\rangle = \lim_{n\to\infty} \big\langle f_n(x), y \big\rangle = \lim_{n\to\infty} \big\langle f_n, K_x y \big\rangle_{\mathcal{H}_K} = \big\langle f, K_x y \big\rangle_{\mathcal{H}_K}. \qquad \square$$

This completes the necessary background knowledge on vector-valued reproducing kernel Hilbert spaces. Finally we can introduce the kernel gradient descent method.

3.4 Kernel Gradient Descent

Kernel gradient descent is a generalization of gradient descent to functionals. The idea is to minimize a differentiable cost C in a special subspace of $\mathcal{F}$, namely a vector-valued reproducing kernel Hilbert space, in which the descent direction is unique. This prevents the problem of choosing a suitable functional gradient. All of the theory in here is based on [7].

Definition 3.14. *Let $K : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}^{n_l \times n_l}$ be a kernel. For $j = 1, \ldots, n_l$ one defines the partial application of K as the function*

$$K_j : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}^{n_l} : (x, x') \mapsto K(x, x')_{\cdot,j},$$

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

Fixing one argument of K_j results in a function in $\mathcal{F}$. Namely for any $x \in \mathbb{R}^{n_0}$, we have

$$K_j(x, \cdot) : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l} : x' \mapsto K_j(x, x'),$$

which is an element of the function space $\mathcal{F}$. This enables the following definition.

Definition 3.15. *Given a kernel $K : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}^{n_l \times n_l}$, 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

$$f_\mu : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l} : x \mapsto \begin{bmatrix} \mu\big[K_1(x, \cdot)\big] \\ \vdots \\ \mu\big[K_{n_l}(x, \cdot)\big] \end{bmatrix}.$$

The mapping Φ_K will be used to restrict our minimization problem to a vector-valued reproducing kernel Hilbert space. To explain this in detail we have to mention the following characterization of Φ_K, which is based on the representation theorem 3.8 and the restriction, that the marginal distribution $\mathcal{D}_x$ is given by the empirical distribution on a finite dataset.

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

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

where $x_1, \dots, x_m \in \mathbb{R}^{n_0}$ denotes the finite support of the empiricial distribution $\mathcal{D}_x$.

Proof. Let $x \in \mathbb{R}^{n_0}$ be arbitrary. By the definition of Φ_K we have

$$\Phi_K\big(\mu\big)(x) = \Phi_K\Big(\langle d, \cdot \rangle_{\mathcal{D}_x}\Big)(x) = \begin{bmatrix} \mu\big[K_1(x, \cdot)\big] \\ \vdots \\ \mu\big[K_{n_l}(x, \cdot)\big] \end{bmatrix} = \begin{bmatrix} \big\langle d, K_1(x, \cdot) \big\rangle_{\mathcal{D}_x} \\ \vdots \\ \big\langle d, K_{n_l}(x, \cdot) \big\rangle_{\mathcal{D}_x} \end{bmatrix}.$$

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

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

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

$$\Phi_K\big(\mu\big)(x) = \begin{bmatrix} \big\langle d, K_1(x, \cdot) \big\rangle_{\mathcal{D}_x} \\ \vdots \\ \big\langle d, K_{n_l}(x, \cdot) \big\rangle_{\mathcal{D}_x} \end{bmatrix} = \mathop{\mathbb{E}}_{x' \sim \mathcal{D}_x} \begin{bmatrix} d(x')^\mathsf{T} K_1(x, x') \\ \vdots \\ d(x')^\mathsf{T} K_{n_l}(x, x') \end{bmatrix} = \mathop{\mathbb{E}}_{x' \sim \mathcal{D}_x} \Big[d(x')^\mathsf{T} K(x, x') \Big]^\mathsf{T}.$$

With $K(x, x')^\mathsf{T} = K(x, x')$ for every $x, x' \in \mathbb{R}^{n_0}$ by definition, we conclude

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

The last step consists of using the assumption, that $\mathcal{D}_x$ is the empirical distribution on a finite dataset $x_1, \dots, x_m \in \mathbb{R}^{n_0}$. This enables us to derive

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

which proves the statement of the lemma. □

Note that the definition of Φ_K does not require definiteness of the kernel K. However, in the case where K is positive semidefinite we have seen in theorem 3.12, that K induces a vector-valued reproducing kernel Hilbert space. Thus Φ_K can be understood as a mapping

$$\Phi_K : \mathcal{F}^* \to \mathcal{H}_K : \mu \mapsto \frac{1}{m} \sum_{i=1}^{m} K_{x_i} y_i,$$

where $y_i \coloneqq d(x_i)$. Based on the notion of Φ_K we can define the kernel gradient of a functional.

Definition 3.17. *Let K be a kernel and the functional $\mu : \mathcal{F} \to \mathbb{R}$ be differentiable at $f_0 \in \mathcal{F}$. The kernel gradient of μ at f_0 with respect to K is defined as*

$$\nabla_K \mu|_{f_0} : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l} : x \mapsto \Phi_K\Big(\partial_f \mu|_{f_0}\Big)(x).$$

The kernel gradient is well-defined based on the restriction, that $\mathcal{D}_x$ is given by the empirical distribution on a finite dataset. In lemma 3.9 we have shown, that $\partial_f \mu|_{f_0} \in \mathcal{F}^*$ for any $f_0 \in \mathcal{F}$. In the case where the kernel K is positive semidefinite, we can think of the kernel gradient as a descent direction in the vector-valued reproducing kernel Hilbert space associated with K.

In terms of minimizing a differentiable cost $C : \mathcal{F} \to \mathbb{R}$, the notation translates as follows. By theorem 3.8 there exists $d|_{f_0} \in \mathcal{F}$, such that the functional derivative of C is given by

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

The element $d|_{f_0} \in \mathcal{F}$ can be thought of as a functional gradient. Using lemma 3.16, the kernel gradient of C at f_0 with respect to any kernel K can be written as

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

In contrast to a functional gradient $d|_{f_0} \in \mathcal{F}$, the kernel gradient is in general not the direction of steepest descent. Instead it computes a linear combination of the values of a functional gradient $d|_{f_0}$ at the dataset, using the kernel K. This construction is independent of the functional gradient chosen, since they are equal on the data. Thanks to the kernel K, the kernel gradient generalizes to values outside the dataset, which prevents the problem of choosing a suitable descent direction from the set of functional gradients.

Based on the uniqueness of the kernel gradient, we can finally generalize the gradient descent algorithm to kernel gradient descent, which is formalized as follows.

Algorithm 5 Kernel Gradient Descent

1: Requires kernel K, differentiable cost $C : \mathcal{F} \to \mathbb{R}$, random $f_0 \in \mathcal{F}$ and $\epsilon \geqslant 0$.

2: Initialize $k \leftarrow 0$.

3: **while** $\big\|\nabla_K C|_{f_k}\big\|_{\mathcal{D}_x} > \epsilon$ **do**:

4: Choose a step size $\eta_k > 0$.

5: Set $f_{k+1} \leftarrow f_k - \eta_k \cdot \nabla_K C|_{f_k}$ and $k \leftarrow k + 1$.

6: **end while**

Again, this is based on the assumption, that $\mathcal{D}_x$ is given by the empirical distribution on a finite dataset. Otherwise we could not guarantee, that the sequence of kernel gradients produced by the algorithm is well-defined.

In the convergence analysis of gradient descent, with the choice of an appropriate step size, the empirical error E was strictly decreasing along the parameter sequence $(w_k)_{k=0}^{\infty}$ in $\mathbb{R}^n$.

This guaranteed the convergence to a critical point, as long as the objective function was bounded from below. We will analyze kernel gradient descent in a similar manner. However, in contrast to the analysis of gradient descent, we will investigate the change in C during kernel gradient descent along a continuous curve in the function space $\mathcal{F}$.

Definition 3.18. *Let K be a kernel. A continuous time dependent function*

$$f : [0, \infty) \to \mathcal{F} : t \mapsto f(t)$$

is said to follow the kernel gradient descent on C with respect to K if it satisfies

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

Roughly speaking, we can think of such a time dependent function as the continuous extension to the sequence $(f_k)_{k=0}^{\infty}$ in $\mathcal{F}$, produced by kernel gradient descent on C, when using an infinitely small step size. Hence $f(0) = f_0$ denotes the initialization of kernel gradient descent.

Furthermore, the convergence analysis is based on the observation, that kernels induce symmetric bilinear forms with respect to $\mathcal{D}_x$. This is formulated in the following lemma.

Lemma 3.19. *A kernel K induces the symmetric bilinear form*

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

where $x, x' \in \mathbb{R}^{n_0}$ are drawn independently according to $\mathcal{D}_x$.

Proof. Let $f_1, f_2, f, g \in \mathcal{F}$ and $\lambda \in \mathbb{R}$. We observe, that

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

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

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

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

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

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

Thus $\langle \cdot, \cdot \rangle_K$ is indeed symmetric and bilinear. □

The approach to prove convergence of kernel gradient descent is to show, that the cost C is strictly decreasing along the curve, defined by a continuous time dependent function f, that follows the kernel gradient descent on C with respect to a kernel K. In other words

$$\forall t \in [0, \infty) : \partial_t C|_{f(t)} \coloneqq \partial_t \big(C \circ f\big)(t) \leqslant 0,$$

such that the inequality is strict as long as kernel gradient descent has not converged. More precisely, convergence along the curve $f(t)$ refers to finding some $t \in [0, \infty)$, such that

$$\big\|\nabla_K C|_{f(t)}\big\|_{\mathcal{D}_x} = 0.$$

Based on this we denote the following result, to prove convergence of kernel gradient descent.

Lemma 3.20. *Let K be a kernel and f be a time dependent function, that follows the kernel gradient descent on C with respect to K. During kernel gradient descent, $C \circ f$ evolves as*

$$\partial_t C|_{f(t)} = -\big\langle d|_{f(t)}, d|_{f(t)} \big\rangle_K,$$

where $d|_{f(t)} \in \mathcal{F}$ denotes a 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. According to the chain rule for functionals in Appendix A of [8], we have

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

where we have used the assumption, that f follows the kernel gradient descent on C with respect to the kernel K. 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

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

Finally we conclude, that

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

This completes the proof. □

Thus, to guarantee that the cost C is strictly decreasing, we have to ensure, that

$$\forall t \in [0, \infty) : \left\langle d|_{f(t)}, d|_{f(t)} \right\rangle_K \geqslant 0,$$

such that the inequality is strict, except at points $t \in [0, \infty)$, where the kernel gradient vanishes, that is $\left\|\nabla_K C|_{f(t)}\right\|_{\mathcal{D}_x} = 0$. In contrast to $\langle \cdot, \cdot \rangle_{\mathcal{D}_x}$, the bilinear map $\langle \cdot, \cdot \rangle_K$ is in general not positive semidefinite. This can be seen from a simple counterexample. Consider

$$K : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}^{n_l} : x \mapsto -I_{n_l},$$

then for $f \equiv (1, \ldots, 1)^T \in \mathcal{F}$ we observe, that

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

However, the semidefiniteness of $\langle \cdot, \cdot \rangle_K$ can be realized with the choice of a suitable kernel.

Definition 3.21. *We say that a kernel K is positive definite with respect to $\mathcal{D}_x$, if*

$$\forall f \in \mathcal{F} : \|f\|_{\mathcal{D}_x} > 0 \implies \langle f, f \rangle_K > 0.$$

Given a positive definite kernel K, the bilinear form $\langle \cdot, \cdot \rangle_K$ is positive semidefinite. This follows directly from definition 3.21 and the fact, that for $f \in \mathcal{F}$ with $\|f\|_{\mathcal{D}_x} = 0$, we have

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

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

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

so $\langle \cdot, \cdot \rangle_K$ is positive semidefinite. Moreover, a positive definite kernel with respect to $\mathcal{D}_x$, ensures that C is strictly decreasing during kernel gradient descent. By definition 3.21, we just have to investigate points $t \in \mathbb{R}$, such that $\left\|d|_{f(t)}\right\|_{\mathcal{D}_x} = 0$. In this case, we have

$$\forall x \sim \mathcal{D}_x : d|_{f(t)}(x) = 0,$$

such that we can conclude $\nabla_K C|_{f(t)} = 0$ based on the construction on the data points, namely

$$\nabla_K C|_{f(t)} = \frac{1}{m} \sum_{i=1}^{m} K(\cdot, x_i) d|_{f(t)}(x_i) = 0.$$

Finally, if the cost is convex and bounded from below and the kernel K is positive definite with respect to $\mathcal{D}_x$, we conclude the convergence of $f(t)$ to a global minimum as $t \to \infty$.

As explained before, it is easy to ensure convexity of the cost C in the function space, such that we have nice convergence properties. This is great for theoretical analysis. In practice however, we are still in need of the explicit parameters $w \in \mathbb{R}^n$, that describe the objective function. Since in general any function can be parametrized in several ways, kernel gradient descent does not prevent us from training neural networks via some sort of gradient descent. Nevertheless, in the next section a relationship between the two minimization approaches will be established, such that we get a better understanding of gradient descent in the parameter space, based on the properties of kernel gradient descent in the function space.

3.5 Neural Tangent Kernel

The parameter training of neural networks via gradient descent is strongly connected to kernel gradient descent with respect to a special kernel, the neural tangent kernel. To motivate this connection, we start with a simple example, inspired by the approach of [11], in which the objective function depends linear on its parameters. The example will be generalized, leading to the definition of the neural tangent kernel. The theory in here is based on [7].

At first, we let $\mathcal{D}_{\mathcal{F}}$ be any probability distribution on the function space $\mathcal{F}$, whose support is given by a subset of the continuous functions on $\mathbb{R}^{n_0}$. Next, we choose an $n \in \mathbb{N}$ and draw random functions $f^{(k)} : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l}$ for $k = 1, \dots, n$ independently from the distribution $\mathcal{D}_{\mathcal{F}}$. Using these, we can define an objective function, which is parametrized via

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

Clearly, we observe for any $w, w' \in \mathbb{R}^n$, that

$$f_{w+w'} = \frac{1}{\sqrt{n}} \sum_{k=1}^{n} (w + w')_k f^{(k)} = \frac{1}{\sqrt{n}} \sum_{k=1}^{n} w_k f^{(k)} + \frac{1}{\sqrt{n}} \sum_{k=1}^{n} w'_k f^{(k)} = f_w + f_{w'}.$$

Moreover we have for any $w \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$, that

$$f_{\lambda w} = \frac{1}{\sqrt{n}} \sum_{k=1}^{n} \lambda w_k f^{(k)} = \lambda \cdot \frac{1}{\sqrt{n}} \sum_{k=1}^{n} w_k f^{(k)} = \lambda f_w,$$

such that F^{lin} is linear in the parameters $w \in \mathbb{R}^n$. Given labeled training data, the objective function can be learned via gradient descent on the empirical error

$$E \coloneqq C \circ F^{lin} : \mathbb{R}^n \to \mathbb{R},$$

where $C : \mathcal{F} \to \mathbb{R}$ denotes a cost functional. Our goal is to construct a kernel K, such that gradient descent on $C \circ F^{lin}$ is equivalent to kernel gradient descent on C with respect to K.

Similar to the approach of analyzing the convergence of kernel gradient descent along a continuous curve in the function space $\mathcal{F}$, we can analyze gradient descent along a continuous curve in the parameter space $\mathbb{R}^n$. For this we have to notate the following definition.

Definition 3.22. *A continuous time dependent function*

$$w : [0, \infty) \to \mathbb{R}^n : t \mapsto w(t)$$

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 $w(t)$.

Roughly speaking we can think of such a time dependent function as the continuous extension to the sequence $(w_k)_{k=0}^{\infty}$ in $\mathbb{R}^n$, produced by gradient descent on $C \circ F^{lin}$, when using an infinitely small step size. Hence $w(0) = w_0$ denotes the initialization of gradient descent.

To establish a relationship between gradient descent and kernel gradient descent, we have to construct a kernel K, such that given a continuous time dependent function $w : [0, \infty) \to \mathbb{R}^n$, that follows the gradient descent on $C \circ F^{lin}$, the continuous time dependent function

$$F^{lin} \circ w : [0, \infty) \to \mathcal{F} : t \mapsto f_{w(t)}$$

follows the kernel gradient descent on C with respect to the kernel K. In other words, the evolution of C during kernel gradient descent and the evolution of $C \circ F^{lin}$ during gradient descent is governed by the same parameter curve $w(t)$.

To simplify the further notation, we introduce the following definition.

Definition 3.23. *For $y, \hat{y} \in \mathbb{R}^{n_l}$ we define the outer product $y \otimes \hat{y}$ as the $n_l \times n_l$ matrix*

$$y \otimes \hat{y} \coloneqq y\hat{y}^{\mathsf{T}}.$$

For $k = 1, \dots, n$, the partial derivatives $\partial_{w_k} F^{lin}$ of the parametrization are given by

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

Using these, we can introduce the so called tangent kernel $\tilde{K}$, which is defined as follows.

Definition 3.24. *Given n random functions $f^{(k)} : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l}$ 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*

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

In other words, $\tilde{K}$ maps a tuple $(x, x') \in \mathbb{R}^{n_0} \times \mathbb{R}^{n_0}$ to the matrix $\tilde{K} \in \mathbb{R}^{n_l \times n_l}$ with entries

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

We will prove, that gradient descent on $C \circ F^{lin}$ is equivalent to kernel gradient descent on the cost C with respect to the tangent kernel $\tilde{K}$, which is based on the following result.

Lemma 3.25. *Let $w : [0, \infty) \to \mathbb{R}^n$ 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*

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

where $d|_{f_{w(t)}} \in \mathcal{F}$ denotes the representer of $\partial_f C|_{f_{w(t)}} = \left\langle d|_{f_{w(t)}}, \cdot \right\rangle_{\mathcal{D}_x} \in \mathcal{F}^*$ for $t \in [0, \infty)$.

Proof. According to the chain rule for functionals in Appendix A of [8], we have

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

for $k = 1, \dots, n$. Using the linearity of $\langle \cdot, \cdot \rangle_{\mathcal{D}_x}$ completes the proof. □

Based on this, we can prove that the dynamics of gradient descent on $C \circ F^{lin}$ and kernel gradient descent on C with respect to the tangent kernel $\tilde{K}$ are governed by the same parameter curve $w(t)$ in $\mathbb{R}^n$. The corresponding result is formulated as follows.

Lemma 3.26. *Let $w : [0, \infty) \to \mathbb{R}^n$ be a time dependent function, that follows the gradient descent on $C \circ F^{lin}$, then the time dependent function $F^{lin} \circ w : [0, \infty) \to \mathcal{F} : t \mapsto f_{w(t)}$ 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. On one hand, we can use the definition of F^{lin} and apply lemma 3.25 to derive

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

On the other hand, using the equation

$$\tilde{K}(x, x_i) = \frac{1}{n} \sum_{k=1}^{n} f^{(k)}(x) \otimes f^{(k)}(x_i) = \frac{1}{n} \sum_{k=1}^{n} f^{(k)}(x) f^{(k)}(x_i)^{\mathsf{T}},$$

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

$$\begin{aligned}
\nabla_{\tilde{K}} C|_{f_{w(t)}}(x) &= \frac{1}{m} \sum_{i=1}^{m} \tilde{K}(x, x_i) d|_{f_{w(t)}}(x_i) \\
&= \frac{1}{m} \sum_{i=1}^{m} \frac{1}{n} \sum_{k=1}^{n} f^{(k)}(x) f^{(k)}(x_i)^{\mathsf{T}} d|_{f_{w(t)}}(x_i) \\
&= \frac{1}{n} \sum_{k=1}^{n} f^{(k)}(x) \cdot \frac{1}{m} \sum_{i=1}^{m} f^{(k)}(x_i)^{\mathsf{T}} d|_{f_{w(t)}}(x_i) \\
&= \frac{1}{n} \sum_{k=1}^{n} f^{(k)}(x) \cdot \mathop{\mathbb{E}}_{x' \sim \mathcal{D}_x} \Big[f^{(k)}(x')^{\mathsf{T}} d|_{f_{w(t)}}(x') \Big] \\
&= \frac{1}{n} \sum_{k=1}^{n} f^{(k)}(x) \cdot \big\langle f^{(k)}, d|_{f_{w(t)}} \big\rangle_{\mathcal{D}_x},
\end{aligned}$$

Comparison of the two expressions results in

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

In other words, $F^{lin} \circ w$ 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 $i, j = 1, \dots, n_l$ the law of large numbers yields

$$\forall x, x' \in \mathbb{R}^{n_0} : \lim_{n\to\infty} \tilde{K}_{i,j}(x, x') = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} f_i^{(k)}(x) \cdot f_j^{(k)}(x') = \mathop{\mathbb{E}}_{f\sim\mathcal{D}_\mathcal{F}} \Big[f_i(x) \cdot f_j(x') \Big].$$

This motivates the definition of a constant limiting kernel K, such that for $x, x' \in \mathbb{R}^{n_0}$ the image $K(x, x')$ is given by the $n_l \times n_l$ matrix defined as

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

This definition is well-defined, since we assumed any $f \sim \mathcal{D}_\mathcal{F}$ to be continuous on $\mathbb{R}^{n_0}$. The random tangent kernel $\tilde{K}$ is hence an approximation of the constant limiting kernel K.

The training of neural networks behaves quite similar to the kernel gradient descent with respect to the tangent kernel. We consider the realization function

$$F : \mathbb{R}^n \to \mathcal{F}|_{\tilde{f}} : w \mapsto f_w \coloneqq \tilde{f}(\cdot, w),$$

mapping a parameter vector $w \in \mathbb{R}^n$ to its corresponding network function, with regard to the architecture $\tilde{f}$. Similarly to the definition of the tangent kernel, we can introduce the neural tangent kernel. Formally it is given as follows.

Definition 3.27. *Given parameters $w \in \mathbb{R}^n$, the neural tangent kernel is defined as*

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

In contrast to the previous example, the realization function F is not linear, due to the nested construction of neural networks, so the partial derivatives $\partial_{w_k} F$ are no longer independent of the parameters w. Thus, in contrast to the tangent kernel, which is constant during training due to its independency of the parameters w, the neural tangent kernel is random at initialization and varies during training. As a result, the relationship between gradient descent and kernel gradient descent with respect to the neural tangent kernel is more complex.

Let us clarify this problem. For the same reasons as in lemma 3.26, during training via gradient descent on $C \circ F$, the network function f_w evolves along the differential equation

$$\partial_t f_{w(t)} = -\nabla_{\Theta_w} C|_{f_{w(t)}}.$$

However, in contrast to the situation in lemma 3.26, the kernel Θ_w varies during training, such that there is no direct connection to kernel gradient descent.

In the previous example, the random kernel $\tilde{K}$ converged to the fixed kernel K, as the number of parameters n tended to infinity. This motivates to study the training of neural networks in the infinite width limit. More precisely the setting, where the width of each hidden layer increases to infinity sequentially over the total number of hidden layers. This is denoted as

$$n_1, \dots, n_{l-1} \to \infty \;\equiv \lim_{n_{l-1}\to\infty} \cdots \lim_{n_1\to\infty}.$$

In the following we assume, that the network parameters $w \in \mathbb{R}^n$ are initialized as i.i.d. Gaussian random variables. In other words, for $i = 1, \dots, n$ it holds, that

$$w_i \sim \mathcal{N}(0, 1).$$

Given this initialization, one of the main results in [7] is, that in the infinite width limit, the neural tangent kernel converges in probability to an explicit deterministic limiting kernel.

Theorem 3.28. *Let $f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l}$ be a neural network of depth l at initialization, with a Lipschitz continuous non-linear activation function $\sigma : \mathbb{R} \to \mathbb{R}$ at each layer. Then, in the limit as the layer widths $n_1, \dots, n_{l-1} \to \infty$ sequentially, the initial neural tangent kernel Θ_{w_0} converges in probability to a deterministic limiting kernel, that is*

$$\Theta_{w_0} \to \Theta_\infty \otimes I_{n_l}.$$

The scalar kernel $\Theta_\infty : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}$ is defined recursively as $\Theta_\infty \coloneqq \Theta_\infty^{(l)}$ by

$$\begin{aligned} \Theta_\infty^{(1)}(x, x') &= \Sigma^{(1)}(x, x') \\ \Theta_\infty^{(i+1)}(x, x') &= \Theta_\infty^{(i)}(x, x')\dot{\Sigma}^{(i+1)}(x, x') + \Sigma^{(i+1)}(x, x'), \end{aligned}$$

where

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

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

A proof via induction over the number of layers is given in [7]. Note that Θ_∞ depends only on the choice of σ, the number of layers and the variance of parameters at initialization. As a consequence, the limiting kernel Θ_∞ is deterministic, despite the random initialization of w.

The theorem states, that for any initialization $w_0 \in \mathbb{R}^n$ and any $x, x' \in \mathbb{R}^{n_0}$ it holds, that

$$\forall \epsilon > 0 : \lim_{n_{l-1} \to \infty} \cdots \lim_{n_1 \to \infty} \mathbb{P}\Big(\big| (\Theta_{w_0})_{i,j}(x, x') - \Theta_\infty(x, x') \big| > \epsilon \Big) = 0, \quad i, j = 1, \dots, n_l.$$

The second key result in [7] is that the neural tangent kernel stays asymptotically constant during training. To formalize this, we have to notate a slightly more general definition of training. More precisely, in the infinite width limit the parameter space is infinite-dimensional, such that it is in practical nonsense to speak of parameter training. Thus, instead of denoting objects in dependence of the parameters $w(t)$, we use a notation in dependence of time t. In contrast to lemma 3.25, where a descent direction was denoted as $d|_{f_{w(t)}} \in \mathcal{F}$, such that

$$\partial_t w_k(t) = -\big\langle d|_{f_{w(t)}}, \partial_{w_k} f_{w(t)} \big\rangle_{\mathcal{D}_x},$$

we denote a descent direction as $d|_{f_t} \in \mathcal{F}$, such that

$$\partial_t w_k(t) = -\big\langle d|_{f_t}, \partial_{w_k} f_t \big\rangle_{\mathcal{D}_x}.$$

Furthermore, we denote the neural tangent kernel at time $t \in [0, \infty)$ as Θ_t, instead of $\Theta_{w(t)}$. Based on this, we can formalize the asymptotic behavior of the neural tangent kernel.

Theorem 3.29. *Let $f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_l}$ be a neural network of depth l, with a twice differentiable non-linear activation function $\sigma : \mathbb{R} \to \mathbb{R}$ at each layer, whose second derivative is bounded. Then, for any $T > 0$, such that in the limit as the layer widths $n_1, \dots, n_{l-1} \to \infty$ sequentially, the integral $\int_0^T \|d_t\|_{\mathcal{D}_x} dt$ stays stochastically bounded, we have uniformly for $t \in [0, T]$, that*

$$\Theta_t \to \Theta_\infty \otimes I_{n_l}.$$

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

$$\partial_t f_t = \Phi_{\Theta_\infty \otimes I_{n_l}}\Big(\big\langle d|_{f_t}, \cdot \big\rangle_{\mathcal{D}_x}\Big).$$

A proof via induction over the number of layers is given in [7].

The theorem states, that for any time $t \in [0, T]$ and any $x, x' \in \mathbb{R}^{n_0}$ it holds, that

$$\lim_{n_{l-1}\to\infty} \cdots \lim_{n_1\to\infty} \Big|\big(\Theta_{w(t)}\big)_{i,j}(x, x') - \big(\Theta_{w(0)}\big)_{i,j}(x, x')\Big| = 0, \quad i, j = 1, \dots, n_l.$$

Thus, in the infinite width limit, the neural tangent kernel stays asymptotically constant during training. As discussed in section 3.4, for a lower bounded and convex cost C, the convergence of kernel gradient descent with respect to the neural tangent kernel to a global minimum of C is hence guaranteed, if $\Theta_\infty \otimes I_{n_l}$ is positive definite with respect to $\mathcal{D}_x$.

This concludes the necessary theory on the neural tangent kernel. Next we will use its properties to motivate our approach of training deep neural networks in tiny subspaces.

4 Training in Low-Dimensional Subspaces

This section is dedicated to the theoretical considerations on how to efficiently reduce the computation effort needed, to train deep neural networks. If not otherwise specified, the following results stem from [1]. The authors approach is based on the hypothesis, that deep neural networks can be trained in low-dimensional subspaces. They argue, that the parameter optimization trajectory could be embedded in a tiny subspace. We will start by motivating this hypothesis based on the previous seen properties of the neural tangent kernel. Afterwards we will introduce an efficient method for the subspace extraction, that enables us to train the network in a tiny subspace. Finally we can construct a second order optimization method training the parameters in the lower-dimensional space.

4.1 Approach

In the following we will denote the parameter training sequence by $(w_k)_{k=0,\dots,K}$ after $K \in \mathbb{N}$ iterations. This sequence naturally arises during any of the proposed optimization methods in section 2.2. The low-dimensional landscape hypothesis from [1] assumes that, there exists a low-dimensional affine set which approximately contains the optimization trajectory. This can be simply illustrated by the following example.

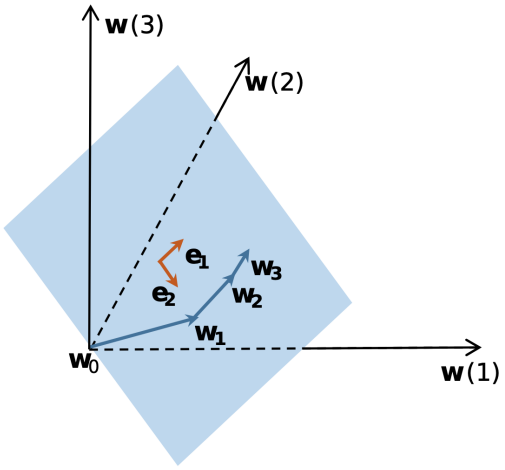

Figure 4: Low-dimensional parameter trajectory

Figure 4 visualizes a three-dimensional space containing the three optimizable parameters $w(1), w(2), w(3)$. However, the training trajectory $(w_k)_{k=0,\dots,3}$ could be embedded in a two-dimensional hyperplane, spanned by the orthogonal vectors e_1 and e_2. Note that none of the optimizable variables could be left out to reduce the dimension, but rather one has to construct new independent variables to obtain the lower-dimensional subspace that approximately contains the optimization trajectory.

In the following we will investigate this hypothesis from a theoretical background and conduct several experiments indicating that deep neural networks can be well trained in tiny subspaces of approximately dimension 50. This enables us to efficiently use second order information in the training process and improves robustness against label noise.

4.2 Theory

To effectively reduce the dimensionality we need some background knowledge on the singular value decomposition (SVD) and low-rank approximations.

4.2.1 Singular Value Decomposition

The following result is denoted and proven as theorem 2.5.2 in [14].

Theorem 4.1 (Singular Value Decomposition)**.** *For every matrix $A \in \mathbb{R}^{m\times n}$ there exist orthogonal matrices $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ and a diagonal matrix*

$$\Sigma \coloneqq diag(\sigma_1, \dots, \sigma_p) \in \mathbb{R}^{m\times n}$$

with $p \coloneqq min\{m, n\}$ and singular values $\sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_p \geqslant 0$, such that

$$A = U\Sigma V^\mathsf{T}.$$

The SVD decomposes any real valued matrix in a product of three special matrices. This representation yields the singular values which, like eigenvalues, characterize important properties of a matrix. For the further course we need the following definition.

Definition 4.2. *The columns of the matrix $U = [u_1, \dots, u_m]$ are referred to as left singular vectors and the columns of the matrix $V = [v_1, \dots, v_n]$ are referred to as right singular vectors.*

Theorem 4.1 shows that the sequence of singular values is unique, since they are sorted in descending order. However, the matrices U and V do not have to be unique. For example, the corresponding singular vectors can be interchanged if there are two identical singular values. Therefore, a matrix can have several singular value decompositions.

As a simple implication of theorem 4.1 we denote the following corollary.

Corollary 4.3. *Let $A \in \mathbb{R}^{m\times n}$ be a matrix with singular value decomposition $A = U\Sigma V^\mathsf{T}$, such that $p \coloneqq min\{m, n\}$ and $\sigma_1 \geqslant \cdots \geqslant \sigma_r > \sigma_{r+1} = \cdots = \sigma_p = 0$ for $r \leqslant p$.*

1. *Denoting the columns of U and V with u_i and v_i, we have*

$$Av_i = \sigma_i u_i, \quad A^\mathsf{T} u_i = \sigma_i v_i, \quad i = 1, \dots, p.$$

2. *The squares of the singular values $\sigma_1^2, \dots, \sigma_r^2$ are the eigenvalues of $A^\mathsf{T}A$ to the corresponding eigenvectors $v_1, \dots, v_r$.*

Proof.

1. Using $A = U\Sigma V^\mathsf{T}$ and the orthogonality of V we derive

$$Av_i = U\Sigma V^\mathsf{T} v_i = U\Sigma e_i = U\sigma_i e_i = \sigma_i u_i$$

for $i = 1, \dots, p$. Analogously we conclude the second statement

$$A^\mathsf{T} u_i = V\Sigma U^\mathsf{T} u_i = V\Sigma e_i = V\sigma_i e_i = \sigma_i v_i.$$

2. Since $A^\mathsf{T}A$ is symmetric, its spectral decomposition is given by

$$A^\mathsf{T}A = V\Sigma^\mathsf{T}U^\mathsf{T}U\Sigma V^\mathsf{T} = V\Sigma^\mathsf{T}\Sigma V^\mathsf{T} = V\Sigma^\mathsf{T}\Sigma V^{-1}.$$

Using $\Sigma^\mathsf{T}\Sigma = \operatorname{diag}(\sigma_1^2, \dots, \sigma_r^2, 0, \dots, 0) \in \mathbb{R}^{n\times n}$ we conclude the corollary. □

The SVD is an excellent tool to find low-rank approximations of a matrix $A \in \mathbb{R}^{m\times n}$, that is to find a matrix $B \in \mathbb{R}^{m\times n}$ with $\operatorname{rank}(B) < \operatorname{rank}(A)$ such that $\|A - B\|$ is small for some given matrix norm. We can generate such a low-rank approximation by simply reducing the number of singular values in the singular value decomposition.

Definition 4.4. *Let $A \in \mathbb{R}^{m\times n}$ be a matrix with singular value decomposition $A = U\Sigma V^\mathsf{T}$. For $k < r = \operatorname{rank}(A)$ we define an approximation of A as*

$$A^k \coloneqq \sum_{i=1}^{k} u_i\sigma_i v_i^\mathsf{T} = \tilde{U}\tilde{\Sigma}\tilde{V}^\mathsf{T},$$

where $\tilde{U} \coloneqq [u_1, \dots, u_k]$, $\tilde{V} \coloneqq [v_1, \dots, v_k]$ and $\tilde{\Sigma} \coloneqq \operatorname{diag}(\sigma_1, \dots, \sigma_k)$.

In terms of the spectral norm $\|\cdot\|_2$, the matrix A^k is the best possible rank-k approximation. The underlying theorem is the following, which is denoted and proven as theorem 2.5.3 in [14].

Theorem 4.5 (Eckart-Young-Mirsky)**.** *Let $A \in \mathbb{R}^{m\times n}$ be a matrix with singular value decomposition $A = U\Sigma V^\mathsf{T}$. For every $k < r = \operatorname{rank}(A)$ it holds, that*

$$\min_{\operatorname{rank}(B)=k} \|A - B\|_2 = \|A - A^k\|_2 = \sigma_{k+1}.$$

This concludes the necessary background knowledge on singular value decompositions and low-rank approximations, which will be useful for the subspace extraction.

4.2.2 Dynamic Linear Dimensionality Reduction

Next, we investigate a single output neural network architecture

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

As seen in section 2.2 the neural network architecture f induces a function space

$$\mathcal{F} \coloneqq \Big\{ f_w : \mathbb{R}^d \to \mathbb{R} : x \mapsto f(x, w) \mid w \in \mathbb{R}^n \Big\}.$$

In order to train the neural network we let

$$\mathcal{X} = \big\{(x_i, y_i) \mid i = 1, \dots, m\big\} \subset \mathbb{R}^d \times \mathbb{R}$$

denote a training set of size $m \in \mathbb{N}$. Recall that training of f refers to finding the optimal parameter vector w, which is accomplished by minimizing the empirical error function

$$E : \mathbb{R}^n \to \mathbb{R}_+ : w \mapsto \sum_{i=1}^{m} \ell\big(f(x_i, w), y_i\big).$$

Note that in comparison to the formulation given in section 2.2, we removed the constant factor $1/m$ in front of the sum, which has no influence on the minimum. Furthermore we assumed $\lambda = 0$, that is we do not penalize the complexity of w.

Under the assumption, that all activation functions in f are differentiable, thus f is differentiable itself, the choice of a differentiable loss function ℓ enables us to minimize E via gradient descent. In order to analyze this optimization process we define the cost functionals

$$\mathcal{L}_i : \mathcal{F} \to \mathbb{R} : f_w \mapsto \ell(f_w(x_i), y_i)$$

for $i = 1, \dots, m$, such that the empirical error E can be written as

$$E(w) = \sum_{i=1}^{m} \ell\big(f(x_i, w), y_i\big) = \sum_{i=1}^{m} \ell\big(f_w(x_i), y_i\big) = \sum_{i=1}^{m} \mathcal{L}_i(f_w).$$

To investigate the parameter evolution during gradient descent on E we will use the concept of gradient flows.

Definition 4.6. *The gradient flow according to E is a time dependent function*

$$w : [0, \infty) \to \mathbb{R}^n : t \mapsto w_t,$$

that suffices the ordinary differential equation

$$\partial_t w_t = -\nabla E(w_t).$$

Since gradient descent performs parameter updates with regard to the update rule

$$w_{k+1} \leftarrow w_k - \eta \cdot \nabla E(w_k),$$

we can let $\eta \to 0$, to derive the equation

$$\lim_{\eta \to 0} \frac{w_{k+1} - w_k}{\eta} = -\nabla E(w_k).$$

Thus, the gradient flow w covers the parameter dynamics during gradient descent on E.

Lemma 4.7. *In the case of a single output neural network architecture, the gradient flow according to E suffices the ordinary differential equation*

$$\partial_t w_t = -\nabla_w f(\mathcal{X}, w_t)^\mathsf{T} \cdot \nabla_{f(\mathcal{X},w)} \mathcal{L}|_{f_{w_t}},$$

where

$$\nabla_w f(\mathcal{X}, w_t)^\mathsf{T} \coloneqq \Big[\nabla_w f(x_1, w_t), \dots, \nabla_w f(x_m, w_t)\Big] \in \mathbb{R}^{n \times m}$$

denotes the gradients of f at w_t on the training data set $\mathcal{X}$ and

$$\nabla_{f(\mathcal{X},w)} \mathcal{L}|_{f_{w_t}} \coloneqq \Big[\nabla \mathcal{L}_1|_{f_{w_t}}(x_1), \dots, \nabla \mathcal{L}_m|_{f_{w_t}}(x_m)\Big]^\mathsf{T} \in \mathbb{R}^m$$

denotes the functional gradients of the costs $\mathcal{L}_i$ at f_{w_t} evaluated on x_i for $i = 1, \dots, m$.

Proof. By definition 4.6 we need to show, that

$$\partial_t w_t = -\nabla E(w_t) = -\nabla_w f(\mathcal{X}, w_t)^\mathsf{T} \cdot \nabla_{f(\mathcal{X},w)} \mathcal{L}|_{f_{w_t}}.$$

For arbitrary $w \in \mathbb{R}^n$ we can apply the chain rule to derive the gradient

$$\nabla E(w) = \sum_{i=1}^{m} \nabla_w \ell\big(f(x_i, w), y_i\big) = \sum_{i=1}^{m} \nabla_w f(x_i, w) \cdot \nabla \mathcal{L}_i|_{f_w}(x_i).$$

Using the definitions from the lemma, this is equivalent to

$$\nabla E(w) = \sum_{i=1}^{m} \nabla_w f(x_i, w) \cdot \nabla \mathcal{L}_i|_{f_w}(x_i) = \nabla_w f(\mathcal{X}, w_t)^\mathsf{T} \cdot \nabla_{f(\mathcal{X},w)} \mathcal{L}|_{f_{w_t}}.$$

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

In the infinite width limit, we can apply theorem **??** to approximate the neural network architecture f as a linear model

$$f^{lin}(x, w_t) \approx f(x, w_0) + \nabla_w f(\mathcal{X}, w_0)(w_t - w_0).$$

This is nothing else than a first order Taylor expansion. However, the key observation is, that in the infinite width limit, the error term converges to zero. Based on this, the differential equation in lemma 4.7 can be rewritten as

$$\partial_t w_t = -\nabla_w f(\mathcal{X}, w_0)^\mathsf{T} \cdot \nabla_{f(\mathcal{X},w)} \mathcal{L}|_{f_{w_t}},$$

where the first factor is a constant matrix and the second factor changes over time t.

At this stage we can explain the main idea of how to reduce the dimensionality of the parameter space. If $\nabla_w f(\mathcal{X}, w_0)$ can be approximated by a low-rank matrix, the training trajectory $(w_k)_{k=0,\dots,K}$ would only depend on this low-rank constant matrix and the functional gradient of C, which will enable us to effectively reduce the dimensionality. To clarify this, we apply singular value decomposition on $\nabla_w f(\mathcal{X}, w_0)$ and derive

$$\nabla_w f(\mathcal{X}, w_0) = U_0 \Sigma_0 V_0^\mathsf{T},$$

where $U_0 \in \mathbb{R}^{m \times m}$ and $V_0 \in \mathbb{R}^{n \times n}$ are orthogonal and $\Sigma_0 = \operatorname{diag}(\sigma_1, \dots, \sigma_m)$ is positive semidefinite. Using the notation

$$\tilde{f} : \mathbb{R}^n \to \mathbb{R}^m : w \mapsto \Big[f(x_1, w), \dots, f(x_m, w)\Big]^\mathsf{T},$$

we can apply definition 3.27 to derive the neural tangent kernel

$$\Theta_{w_0} = \sum_{k=1}^{n} \partial_{w_k} \tilde{f}(w_0) \otimes \partial_{w_k} \tilde{f}(w_0) = \nabla_w f(\mathcal{X}, w_0) \cdot \nabla_w f(\mathcal{X}, w_0)^\mathsf{T} = U_0 \Sigma_0 \Sigma_0^\mathsf{T} U_0^\mathsf{T}.$$

This is nothing else than the spectral decomposition of Θ_0. By theorem **??** $\cdots$. Thus by theorem 4.5 we can approximate Σ_0 by a low-rank matrix $\tilde{\Sigma}_0 = \operatorname{diag}(\sigma_1, \dots, \sigma_d)$, such that

$$\Sigma_0 \approx \tilde{U}_0 \tilde{\Sigma}_0 \tilde{V}_0^\mathsf{T},$$

where $\tilde{U}_0 \in \mathbb{R}^{m \times d}$ contains the first d columns of an $m-$dimensional identity matrix and $\tilde{V}_0 \in \mathbb{R}^{n \times d}$ contains the first d columns of an $n-$dimensional identity matrix. Using this approximation we can derive

$$\nabla_w f(\mathcal{X}, w_0) = U_0 \Sigma_0 V_0^\mathsf{T} \approx U_0 \tilde{U}_0 \tilde{\Sigma}_0 \tilde{V}_0^\mathsf{T} V_0^\mathsf{T}.$$

Thus, the dynamics of gradient flow are approximately governed by

$$\partial_t w_t \approx -\nabla_w f(\mathcal{X}, w_0)^\mathsf{T} \cdot \nabla_{f(\mathcal{X},w)} \mathcal{L}|_{f_{w_t}} \approx -V_0 \tilde{V}_0 \Big(\tilde{\Sigma}_0 \tilde{U}_0^\mathsf{T} U_0^\mathsf{T} \cdot \nabla_{f(\mathcal{X},w)} \mathcal{L}|_{f_{w_t}} \Big)$$

The clasped part yields the projected gradient in the $d-$dimensional subspace and the variables $V_0 \tilde{V}_0$ establish a projection from the lower-dimensional space back to the $n-$dimensional parameter space.

This concludes the theoretical motivation of the subspace extraction. Based on this approach we could effectively train neural networks in the lower-dimensional subspace. However, the previous discussion holds only in the infinite width limit for training in the lazy regime. The rest of this thesis is dedicated to numerical experiments indicating that deep neural networks indeed can be trained in low-dimensional subspaces, although the theoretical conditions do not hold in practice.

Problem: Chain rule for functionals.
Problem: Reference and existence of gradient flow

4.3 Methodology

The key issue of dimensionality reduction consist of finding a suitable dimension $d \in \mathbb{N}$ and a $d-$dimensional subspace that approximately covers the optimization trajectory. We will address this problem via principal component analysis (PCA), which is based on the concept of orthogonal projections.

4.3.1 Orthogonal Projections

The results of this subsection are denoted in section 2.6.1 in [14].

Lemma 4.8. *Let $S \subseteq \mathbb{R}^n$ be a subspace. There exists a unique matrix $P \in \mathbb{R}^{n\times n}$, such that*

$$Im(P) = S \ \wedge \ P^2 = P = P^\mathsf{T}.$$

The linear map $\tilde{P} : \mathbb{R}^n \to \mathbb{R}^n : x \mapsto Px$ is called the orthogonal projection onto S.

Proof. To prove the existence of an orthogonal projection, we let $d \coloneqq \dim(S)$ denote the dimension of S and $V := [v_1, \dots, v_d] \in \mathbb{R}^{n\times d}$ denote an orthonormal basis of S. We define

$$P \coloneqq VV^\mathsf{T} \in \mathbb{R}^{n\times n}$$

and observe by the orthonormality of V, that $V^\mathsf{T}V = I_d$. Thus P is idempotent, this is

$$P^2 = VV^\mathsf{T}VV^\mathsf{T} = VI_dV^\mathsf{T} = VV^\mathsf{T} = P.$$

Furthermore P is symmetric, since

$$P^\mathsf{T} = (VV^\mathsf{T})^\mathsf{T} = VV^\mathsf{T} = P.$$

By the orthonormality of V, for arbitrary $s \in S$ there exists $a = (a_1, \dots, a_d)^\mathsf{T} \in \mathbb{R}^d$ with

$$s = a_1v_1 + \cdots + a_dv_d = Va.$$

Based on this we derive $S \subseteq \mathrm{Im}(P)$ since

$$Ps = VV^\mathsf{T}s = VV^\mathsf{T}Va = VI_da = Va = s.$$

Additionally $\mathrm{Im}(P) \subseteq S$ due to the fact, that for $y \in \mathbb{R}^n$, we have $\tilde{a} \coloneqq V^\mathsf{T}y \in \mathbb{R}^d$, so

$$Py = VV^\mathsf{T}y = V\tilde{a} \in S.$$

Thus $\mathrm{Im}(P) = S$. This proves the existence of an orthogonal projection P onto S. To prove uniqueness of P, we should first note that based on $P^2 = P = P^\mathsf{T}$ we have

$$\forall x, y \in \mathbb{R}^n : (Px)^\mathsf{T}(y - Py) = x^\mathsf{T}P^\mathsf{T}y - x^\mathsf{T}P^\mathsf{T}Py = x^\mathsf{T}Py - x^\mathsf{T}Py = 0.$$

Using $Px \in S$ for every $x \in \mathbb{R}^n$, we can conclude that

$$\forall y \in \mathbb{R}^n : y - Py \in S^\perp := \big\{a \in \mathbb{R}^n \mid \forall b \in S : a^\mathsf{T}b = 0\big\}.$$

Next we let P_1 and P_2 denote orthogonal projections onto S, then for any $z \in \mathbb{R}^n$ we derive

$$\begin{aligned}\|(P_1 - P_2)z\|_2^2 &= \|P_1 z - P_2 z\|_2^2 \\ &= z^\mathsf{T} P_1^\mathsf{T} P_1 z - z^\mathsf{T} P_2^\mathsf{T} P_1 z - z^\mathsf{T} P_1^\mathsf{T} P_2 z + z^\mathsf{T} P_2^\mathsf{T} P_2 z \\ &= z^\mathsf{T} P_1 z - z^\mathsf{T} P_2^\mathsf{T} P_1 z - z^\mathsf{T} P_1^\mathsf{T} P_2 z + z^\mathsf{T} P_2 z,\end{aligned}$$

where we have used the property $P_i^2 = P_i = P_i^\mathsf{T}$ for $i = 1, 2$. Since each term on the right-hand side is a real number, we can transpose each term to derive

$$\begin{aligned}\|(P_1 - P_2)z\|_2^2 &= z^\mathsf{T} P_1 z - z^\mathsf{T} P_2^\mathsf{T} P_1 z - z^\mathsf{T} P_1^\mathsf{T} P_2 z + z^\mathsf{T} P_2 z \\ &= (P_1 z)^\mathsf{T} z - (P_1 z)^\mathsf{T} P_2 z - (P_2 z)^\mathsf{T} P_1 z + (P_2 z)^\mathsf{T} z \\ &= (P_1 z)^\mathsf{T}(z - P_2 z) + (P_2 z)^\mathsf{T}(z - P_1 z).\end{aligned}$$

Using $\mathrm{Im}(P_1) = S = \mathrm{Im}(P_2)$, we observe that

$$P_1 z \in S \ \wedge \ z - P_2 z \in S^\perp \ \wedge \ P_2 z \in S \ \wedge \ z - P_1 z \in S^\perp.$$

Thus the right-hand side of the equality is zero and by the positive definiteness of the norm we conclude $P_1 - P_2 = 0$. This implies $P_1 = P_2$ showing the uniqueness of P. □

In our setting, the d-dimensional subspace $S \subseteq \mathbb{R}^n$ is unknown, which is why the following definition will be needed.

Definition 4.9. *For $d \in \mathbb{N}$ we define the set of rank-d orthogonal projection matrices*

$$\mathcal{P}_d \coloneqq \left\{ P \in \mathbb{R}^{n \times n} \mid \mathit{rank}(P) = d \ \wedge \ P^2 = P = P^\mathsf{T} \right\}.$$

Having this background knowledge on orthogonal projections in mind, we can proceed with the theory on principal component analysis.

4.3.2 Principal Component Analysis

The results of this subsection can be found in chapter 15.1 of [15]. In the following we fix $d \leqslant n$ and let $W = [w_1, \ldots, w_t] \in \mathbb{R}^{n \times t}$ be a mean-centered data matrix, that is $\sum_{i=1}^t w_i = 0$.

Principal component analysis consists of finding the orthogonal projection matrix $P^* \in \mathcal{P}_d$, that minimizes the reconstruction error

$$\mathcal{P}_d \to \mathbb{R} : P \mapsto \sum_{i=1}^t \|P w_i - w_i\|_2^2.$$

Equivalently PCA can be formulated as finding the solution $P^* \in \mathcal{P}_d$, such that

$$P^* \coloneqq \operatorname*{arg\,min}_{P \in \mathcal{P}_d} \|PW - W\|_F^2,$$

where $\|\cdot\|_F$ denotes the Frobenius norm. Casually speaking, principal component analysis aims to project the original data to a lower-dimensional space while preserving as much information as possible. The following theorem provides a method on solving the PCA problem via singular value decomposition.

Theorem 4.10. *The PCA solution $P^* \in \mathcal{P}_d$ can be decomposed as*

$$P^* = U_d U_d^\mathsf{T},$$

where $U_d \in \mathbb{R}^{n\times d}$ is the matrix formed by the first d left singular vectors of W.

Proof. Let $P \in \mathcal{P}_d$ be arbitrary. By the linearity of the trace, we observe that

$$\begin{aligned}\|PW - W\|_F^2 &= \mathrm{Tr}\big[(PW-W)^\mathsf{T}(PW-W)\big] \\ &= \mathrm{Tr}\big[W^\mathsf{T}P^2W - 2W^\mathsf{T}PW + W^\mathsf{T}W\big] \\ &= \mathrm{Tr}\big[W^\mathsf{T}W\big] - \mathrm{Tr}\big[W^\mathsf{T}PW\big],\end{aligned}$$

since $P^\mathsf{T} = P = P^2$ by definition. Using that $\mathrm{Tr}\big[W^\mathsf{T}W\big]$ is independent of P, we derive

$$\operatorname*{arg\,min}_{P\in\mathcal{P}_d} = \|PW - W\|_F^2 = \operatorname*{arg\,max}_{P\in\mathcal{P}_d} \mathrm{Tr}\big[W^\mathsf{T}PW\big].$$

By the proof of lemma 4.8, there exists an orthonormal basis $Q = [q_1, \dots, q_d] \in \mathbb{R}^{n\times d}$, such that P can be decomposed as $P = QQ^\mathsf{T}$. Using the invariance of the trace under cyclic permutation and the orthonormality of Q, we have

$$\mathrm{Tr}\big[W^\mathsf{T}PW\big] = \mathrm{Tr}\big[W^\mathsf{T}QQ^\mathsf{T}W\big] = \mathrm{Tr}\big[Q^\mathsf{T}WW^\mathsf{T}Q\big] = \sum_{i=1}^{d} q_i^\mathsf{T}WW^\mathsf{T}q_i.$$

To maximize the rightmost expression we perform SVD on $W = U\Sigma V^\mathsf{T}$, such that

$$WW^\mathsf{T} = U\Sigma V^\mathsf{T}V\Sigma^\mathsf{T}U^\mathsf{T} = U\Sigma\Sigma^\mathsf{T}U^\mathsf{T}.$$

Using the notation $Z = U^\mathsf{T}QQ^\mathsf{T}U$ we derive by the invariance under cyclic permutation, that

$$\mathrm{Tr}\big[Q^\mathsf{T}WW^\mathsf{T}Q\big] = \mathrm{Tr}\big[Q^\mathsf{T}U\Sigma\Sigma^\mathsf{T}U^\mathsf{T}Q\big] = \mathrm{Tr}\big[U^\mathsf{T}QQ^\mathsf{T}U\Sigma\Sigma^\mathsf{T}\big] = \mathrm{Tr}\big[Z\Sigma\Sigma^\mathsf{T}\big].$$

As Z is orthogonal as a product of three orthogonal matrices we conclude (why orthogonal?)

$$\mathrm{Tr}\big[W^\mathsf{T}PW\big] = \mathrm{Tr}\big[Z\Sigma\Sigma^\mathsf{T}\big] = \sum_{i=1}^{d} z_{i,i}\sigma_i^2 \leqslant \sum_{i=1}^{d}\sigma_i^2.$$

The upper bound is attained in the case where $Q = U_d = [u_1, \dots, u_d]$ contains the first d columns of U, such that based on corollary 4.3 we can use $W^\mathsf{T}u_i = \sigma_i v_i$ to conclude

$$\mathrm{Tr}\big[W^\mathsf{T}PW\big] = \sum_{i=1}^{d} q_i^\mathsf{T}WW^\mathsf{T}q_i = \sum_{i=1}^{d} u_i^\mathsf{T}WW^\mathsf{T}u_i = \sum_{i=1}^{d} u_i^\mathsf{T}Wv_i\sigma_i = \sum_{i=1}^{d}\sigma_i^2.$$

This completes the proof. □

Using principal component analysis we can finally construct an effective method for dimensionality reduction.

4.3.3 Algorithm

To find the lower-dimensional subspace, that approximately covers the optimization trajectory we first need to sample $t \in \mathbb{N}$ parameter vectors along the optimization trajectory, which can be achieved by training the network with one of the proposed algorithms in section 2.2.

The idea of dimensionality reduction consists of applying PCA to the parameters $\{w_1, \dots, w_t\}$. For this we need to centralize these samples as $\bar{w} \coloneqq \sum_{i=1}^{t} w_i$ and define

$$W \coloneqq \big[w_1 - \bar{w}, \dots, w_t - \bar{w}\big] \in \mathbb{R}^{n \times t}.$$

This enables us to apply PCA on the mean-centered data matrix W. As seen in section 4.3.2, for $d \in \mathbb{N}$, the orthogonal projection on the $d-$dimensional subspace, that covers the parameter trajectory best, is induced by

$$P^* \coloneqq \operatorname*{arg\,min}_{P \in \mathcal{P}_d} \big\|PW - W\big\|_F^2.$$

By theorem 4.10 we can construct P^* from the left singular vectors of W. For this we apply corollary 4.3 to compute the right singular vectors v_i for $i = 1, \dots, d$ by the spectral decomposition of $W^\mathsf{T} W \in \mathbb{R}^{t \times t}$. These can then be used to derive the left singular vectors

$$u_i = \frac{1}{\sigma_i} W v_i, \quad i = 1, \dots, d.$$

Thus we have found an orthonormal basis $U_d = [u_1, \dots, u_d]$ of the d-dimensional subspace, that covers the optimization trajectory best. The orthogonal projection onto this subspace is given by

$$P : \mathbb{R}^n \to \mathbb{R}^n : w \mapsto U_d U_d^\mathsf{T} w.$$

In summary, the algorithm for dimensionality reduction can be notated as follows.

Algorithm 6 Dynamic Linear Dimensionality Reduction (DLDR)

1: Sample parameter trajectory $\{w_1, \dots, w_t\}$ along the training.

2: Compute the mean $\bar{w} \coloneqq \sum_{i=1}^{t} w_i$.

3: Centralize the samples as $W = [w_1 - \bar{w}, \dots, w_t - \bar{w}]$.

4: Perform spectral decomposition such that $W^\mathsf{T} W = V \Sigma^2 V^{-1}$.

5: Choose suitable subspace dimension $d \in \mathbb{N}$.

6: Obtain the d largest eigenvalues $\big[\sigma_1^2, \dots, \sigma_d^2\big]$ with eigenvectors $\big[v_1, \dots, v_d\big]$.

7: Determine the singular vectors $u_i = 1/\sigma_i W v_i$ for $i = 1, \dots, d$.

8: Return the orthonormal basis $\big[u_1, \dots, u_d\big]$.

For complexity ...

5 Numerical Experiments

6 Conclusion

In conclusion...

References

[1] Tao Li, Lei Tan, Qinghua Tao, Yipeng Liu, Xiaolin Huang. *Low Dimensional Trajectory Hypothesis is True: DNNs can be Trained in Tiny Subspaces.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.

[2] Augustin Cauchy. *Méthode générale pour la résolution des systemes d'équations simultanées.* Comptes rendus de l'Académie des sciences, volume 25, pages 536-538, 1847.

[3] Yurii Nesterov. *Introductory Lectures on Convex Optimization.* Applied Optimization, volume 87, 2004.

[4] Richard H. Byrd, Jorge Nocedal, Robert B. Schnabel. *Representations of Quasi-Newton Matrices and their Use in Limited Memory Methods.* Mathematical Programming, volume 60(1), pages 129-156, 1994

[5] Léon Bottou, Frank E. Curtis, Jorge Nocedal. *Optimization Methods for Large-Scale Machine Learning.* SIAM Review, volume 60(2), pages 223-311, 2018.

[6] Diederik P. Kingma, Jimmy Lei Ba. *Adam: A Method for Stochastic Optimization.* International Conference on Learning Representations (ICLR), 2015.

[7] 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.

[8] Robert G. Parr, Weitao Yang. *Density-Functional Theory of Atoms and Molecules.* International Series of Monographs on Chemistry Books, volume 16, 1989.

[9] Erwin Kreyszig. *Introductory Functional Analysis with Applications.* Wiley Classics Library edition, 1989.

[10] Charles A. Micchelli, Massimiliano Pontil. *On Learning Vector-Valued Functions.* Neural Computation, 17:177-204, 2005.

[11] Ali Rahimi, Ben Recht. *Random Features for Large-Scale Kernel Machines.* Advances in Neural Information Processing Systems, volume 20, pages 1177-1184, 2017.

— Check

[12] 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.

[13] Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, Jeffrey Pennington. *Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent.* Journal of Statistical Mechanics: Theory and Experiment. 2020(12):124002, 2020.

[14] Gene H. Golub, Charles F. Van Loan. *Matrix Computations.* Third edition, 1996.

[15] Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar. *Foundations of Machine Learning.* Second edition, 2018.

b : bias vector
w : weight vector
-
a : element of $\mathcal{A}$
b : batch size
c_i : constants
d : descent direction
$d_{\mathcal{F}}$ pseudo metric on $\mathcal{F}$
$d_{\mathcal{X}}$: metric on input space
$d_{\mathcal{Y}}$: metric on output space
e_i : basis elements
f : real valued function
g : real valued function
h : real valued function
$\tilde{f}$: neural network architecture
i : several indices
j : alternative index
k : iteration index
l : number of layers in neural network
ℓ : loss function
m : real number / number of training samples
m_k : vector in Adam
n_0 : input dimension of neural network
n_l : output dimension of neural network
n_x : input dimension of layer
n_y : output dimension of layer
n : number of parameters in neural network
s : real valued vector in BFGS
v_k : vector in Adam
w : parameter vector in neural network
x : real valued vector (input)
x' : real valued vector (input)
y : real valued vector (output)
z : real valued vector(input)
-
α : basis factors
β_1 : momentum factor
β_2 : momentum factor
ϵ : small positive scalar
η : step size
λ : real valued scalar
ν : neuron
μ : functional
ϕ : direction in functional derivative
ψ : real valued helper function
ρ : real value in BFGS

σ : activation function
-
C : cost functional
B : inverse hessian matrix approximation
E : empirical error
F : realization function
I_n : identity matrix
J : riesz mapping
K : number of epochs / kernel
L : lipschitz constant
M : real valued matrix
R : generalization error
T : linear operator
V : matrix in BFGS
W : weight matrix in layer
-
Φ_K : mapping to Hilbert space
Θ : neural tangent kernel
Σ : covariance matrix
-
$\mathcal{A}$: subset of $\mathcal{F}$
$\mathcal{F}$: function space
$\mathcal{F}^*$ dual space
$\mathcal{F}|_{\tilde{f}}$: set of neural networks with given architecture
$\mathcal{D}$: probability distribution
$\mathcal{D}_x$: marginal probability distribution
$\mathcal{I}$: set of indices
$\mathcal{H}$: hilbert space
$\mathcal{N}$: normal distribution
$\mathcal{X}$: input space
$\mathcal{Y}$: output space
-
$\mathbb{E}$: expected value
$\mathbb{N}$: natural numbers
$\mathbb{P}$: probability
$\mathbb{R}$: real numbers