Non Convex-Concave Saddle Point Optimization

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Non Convex-Concave Saddle Point Optimization

Master Thesis

Leonard Adolphs

April 09, 2018

Advisors: Prof. Dr. Thomas Hofmann, M.Sc. Hadi Daneshmand
Department of Computer Science, ETH Zürich

Abstract

This thesis investigates the theoretically properties of commonly used optimization methods on the saddle point problem of the form

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^n} \max_{\boldsymbol{\varphi}\in\mathbb{R}^m} f(\boldsymbol{\theta}, \boldsymbol{\varphi}),$$

where f is neither convex in $\boldsymbol{\theta}$ nor concave in $\boldsymbol{\varphi}$. We show that gradient-based optimization schemes have undesired stable stationary points; hence, even if convergent, they are not guaranteed to yield a solution to the problem. To remedy this issue, we propose a novel optimizer that exploits extreme curvatures to escape from non-optimal stationary points. We theoretically and empirically prove the advantage of extreme curvature exploitation on the saddle point problem.
Moreover, we explore the idea of using curvature information even further and investigate the properties of second-order methods in saddle point optimization. In this vein, we theoretically analyze the issues that arise in this context and propose a novel approach that uses second-order information to find a structured saddle point.

Acknowledgements

First and foremost, I would like to express my gratitude to my supervisor Hadi Daneshmand for his advice and assistance. Without his vital support and constant help, this thesis would not have been possible.

Furthermore, I would like to thank Prof. Dr. Thomas Hofmann for providing me with the opportunity to do this thesis in his group and his guidance along the way.

Also, I like to thank Dr. Aurelien Lucchi for the interesting research discussions and his valuable comments.

Contents

Chapter 1

Introduction

1.1 Saddle Point Problem

Throughout the thesis, we consider the problem of finding a structured saddle point on a smooth objective, namely the *saddle point problem* (SPP). The optimization is done over the function $f : \mathbb{R}^{n+m} \to \mathbb{R}$, parameterized with $\boldsymbol{\theta} \in \mathbb{R}^n, \boldsymbol{\varphi} \in \mathbb{R}^m$, and the goal is to find a pair of arguments such that f is minimized with respect to the former and maximized with respect to the latter, i.e.,

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}). \tag{1.1}$$

Here, we assume that f is smooth in $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ but **not** necessarily convex in $\boldsymbol{\theta}$ or concave in $\boldsymbol{\varphi}$. As we will see in the upcoming chapter, this problem arises in many applications: e.g., probability density estimation [10], robust optimization [21], and game theory [15].

Solving the saddle point problem of Eq. (1.1) is equivalent to finding a point $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ such that

$$f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}) \leq f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) \leq f(\boldsymbol{\theta}, \boldsymbol{\varphi}^*). \tag{1.2}$$

holds for all $\boldsymbol{\theta} \in \mathbb{R}^n$ and $\boldsymbol{\varphi} \in \mathbb{R}^m$. For a non convex-concave function f, finding such a saddle point is computationally infeasible. For example, even if the parameter vector $\boldsymbol{\varphi}^*$ is given, finding $\boldsymbol{\theta}^*$ is a global non-convex minimization problem that is generally known as being NP-hard.

Local Optima of Saddle Point Optimization Instead of optimizing for a global saddle point, we consider a more modest goal: finding a locally optimal saddle point, i.e., a point $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ for which the condition of Eq. (1.2)

holds true in a local neighborhood

$$\mathcal{K}_\gamma^* = \{(\boldsymbol{\theta}, \boldsymbol{\varphi}) \big| \ \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\| \leq \gamma, \|\boldsymbol{\varphi} - \boldsymbol{\varphi}^*\| \leq \gamma\} \tag{1.3}$$

around $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$, with a sufficiently small $\gamma > 0$. We will call such points *locally optimal saddle points*[1].

Definition 1.1 *The point $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ is a locally optimal saddle point of the problem in Eq.* (1.1) *if*

$$f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}) \leq f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) \leq f(\boldsymbol{\theta}, \boldsymbol{\varphi}^*) \tag{1.4}$$

holds for $\forall (\boldsymbol{\theta}, \boldsymbol{\varphi}) \in \mathcal{K}_\gamma^*$.

Let $\mathcal{S}_f \subset \mathbb{R}^n \times \mathbb{R}^m$ be the set of all locally optimal saddle points; then the problem reduces to finding a point $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) \in \mathcal{S}_f$. Here, we do not consider any partial ordering on members of $\mathcal{S}_f$. We assume that all elements in $\mathcal{S}_f$ are equally good solutions to our saddle point problem.

1.2 Structure of the Thesis and Contributions

The thesis starts by introducing the reader to the saddle point problem and by stating the issues that arise for non convex-concave functions. By showing, in chapter 2, that multiple practical learning scenarios give rise to such a problem, we emphasize the relevance of our following analysis. In chapter 3, we lay out the mathematical foundation for the thesis by introducing the notation, basic definitions, and assumptions. We define the most important gradient-based optimization methods for the saddle point problem in chapter 4 and try to build an intuition for their properties by investigating their behavior on a simple example. Chapter 5 analyzes the dynamics of gradient-based optimization in terms of stability. Through this analysis, we discover a severe issue of gradient-based methods that is unique to saddle point optimization, namely that it introduces stable points that are no solution to the problem we are trying to solve. This is in clear contrast to GD for minimization tasks. Our observation leads to the design of a novel optimizer in chapter 6. With the use of curvature exploitation, we can remedy the identified issue. Moreover, we establish theoretical guarantees for the new method and provide an efficient implementation. We empirically support our arguments with multiple experiments on artificial and real-world data. In chapter 7, we extend our idea of using curvature information in saddle point optimization: instead of considering only the *extreme* curvature, we explore the advantages of using a *full* second-order method. We theoretically

[1] In the context of game theory or Generative Adversarial Networks [10] these points are called local Nash-equilibria. We will formalize this expression and the relationship to locally optimal saddle points in section 3.1.

identify the problems that arise in this context and propose a novel second-order optimizer, that empirically outperforms gradient-based optimization on the saddle point problem.

The contribution of this thesis is fivefold:

1. We show that the Stable-Center manifold theorem [14] also applies to the gradient-based method for the saddle point problem. This implies that a convergent series will almost surely yield a solution that is in the stable set of the method.
2. We provide evidence that gradient-based optimization on the saddle point problem introduces stable points that are not optimal and therefore not a solution to Eq. 1.1.
3. We propose a novel optimizer, called CESP, that uses extreme curvature exploitation [7] to remedy the identified issues. We theoretically prove that – if convergent – this method will yield a solution to the saddle point problem.
4. We prove that optimization methods with a linear-transformed gradient step, such as Adagrad, can't solve the issue of GD as they also introduce non-optimal stable points. For this class of optimizers, we propose a modified variant of CESP.
5. We use the theoretical framework of the generalized trust region method [8] to design a second-order optimizer for saddle point problems.

Chapter 2

Examples of Saddle Point Problems

The problem of finding a structured saddle point arises in varying forms in many different applications. We start this chapter by introducing examples that give rise to a classical saddle point problem with a convex-concave structure. This case has been studied extensively and provides rich theoretical guarantees for conventional optimization methods. In the next step, we motivate the need for *local* saddle point optimization by providing prominent examples of applications where the objective does not have a convex-concave structure.

2.1 Convex-Concave Saddle Point Problems

In many practical learning scenarios the saddle point function f – in Eq. (1.1) – is convex in $\boldsymbol{\theta}$ and concave in $\boldsymbol{\varphi}$. Optimization on such problems has been studied in great depth and, therefore, we only mention one example here for completeness. A detailed list of different applications of convex-concave saddle point problems, ranging from fluid dynamics to economics, is presented in [4].

Constrained Optimization To solve a constrained minimization problem, we need to construct the Lagrangian and minimize it with respect to the model parameters, while maximizing with respect to its Lagrangian multipliers. For example, let's consider the following quadratic problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \mathbf{x}^\top \mathbf{A}\mathbf{x} - \mathbf{b}^\top \mathbf{x} \tag{2.1}$$

where $\mathbf{A} \in \mathbb{R}^{n\times n}$ and $\mathbf{b} \in \mathbb{R}^n$. The minimization is with subject to the constraint

$$\mathbf{B}\mathbf{x} = \mathbf{g} \tag{2.2}$$

where $\mathbf{B} \in \mathbb{R}^{m \times n}$ ($m < n$). By introducing the Lagrangian multiplier $\mathbf{y}$, we can re-formulate it as an unconstrained problem.

$$\min_{\mathbf{x} \in \mathbb{R}^n} \max_{\mathbf{y} \in \mathbb{R}^m} \mathbf{x}^\top \mathbf{A}\mathbf{x} - \mathbf{b}^\top \mathbf{x} + \mathbf{y}^\top (\mathbf{B}\mathbf{x} - \mathbf{g}) \tag{2.3}$$

Hence, transforming a constrained problem of the form (2.1) into the equivalent unconstrained problem in Eq. (2.3), gives rise to a saddle point problem of the general form (1.1). In many applications of computer science and engineering, the Lagrangian multiplier $\mathbf{y}$ has a physical interpretation and its computation is also of interest.

2.2 Generative Adversarial Network

In recent years, we have seen a great rise of interest in the (non convex-concave) saddle point problem, mostly due to the invention of *Generative Adversarial Networks* (GAN) by Goodfellow et al. [10]. The introduction of GANs in 2014 was a significant improvement in the field of generative models. The principal problem in previous generative models was the difficult approximation of intractable probabilistic computations. The goal of a generative model is to capture the underlying data distribution. Observing that most interesting distributions, as for examples images, are high dimensional and potentially very complicated, we intuitively realize that modeling the distribution explicitly may be intractable. GANs overcome this problem by modeling the data distribution in an implicit form. Instead of computing the distribution itself, it models a device called *generator* which can draw samples from the distribution. The breakthrough idea of GANs is embedded in the training procedure of this generator. Inspired by game theory, the generator is faced with an opponent, the *discriminator*, during the training process. The two networks compete in a minimax game where the generator tries to minimize the same objective the discriminator seeks to maximize. Iteratively optimizing this objective with respect to both networks leads to more realistic generated samples over time. Intuitively, this procedure can be described with the simple game where the generator wants to fool the discriminator into believing the generated samples are real. Initially, the generator achieves this goal with very low quality samples. But over time, as the discriminator gets stronger, the generator needs to generate more realistic samples. Eventually, the generator is able to fool any discriminator once it can perfectly imitate the data distribution.

2.2.1 Framework

During the learning process the generator is adjusted such that the generative distribution p_g gets closer to the data distribution p_d. The distribution p_g is implicitly represented by a neural network G from which we can draw

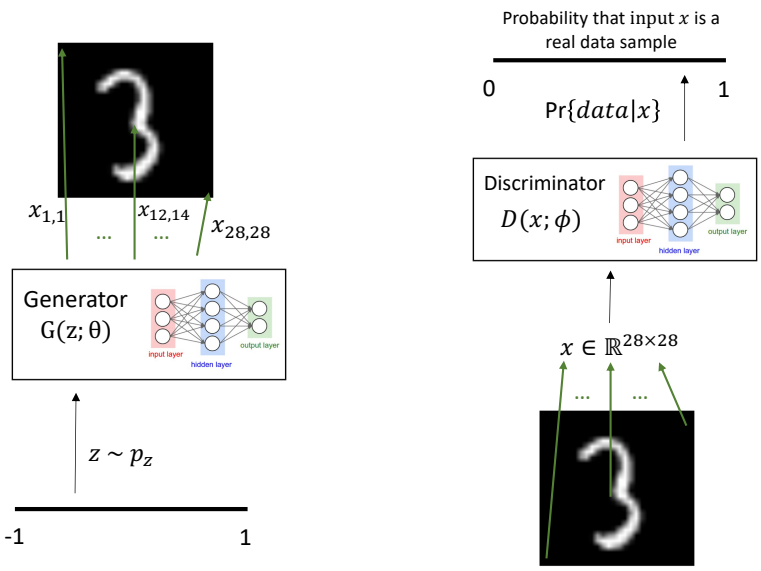


Figure 2.1: High-level sketch of the GAN structure and the interaction of the generator and discriminator networks with an MNIST image as input.

instances. The network is parameterized with $\boldsymbol{\theta}$ and takes as input a latent variable vector $\mathbf{z}$ that we sample from a noise prior p_z. The goal of the training is to learn a parameterization $\boldsymbol{\theta}$ of the network such that its implicit sample distribution mimics the data distribution, i.e., $G(\mathbf{z};\boldsymbol{\theta}) \sim p_d$.
In order to achieve this goal, GANs use a discriminator network $D : \mathcal{X} \to [0,1]$ – parameterized by $\boldsymbol{\varphi}$ – that distinguishes real samples from generated samples by assigning each a probability (0: generated; 1: real). The discriminator is trained in parallel with the generator with real samples $\mathbf{x} \in \mathcal{X}$ and generated samples $\hat{\mathbf{x}} \in G(\mathbf{z};\boldsymbol{\theta})$. Through the simultaneous updates, both networks advance in their task concurrently, inducing the need for their opponent to further improve. A high-level view of the network structure with MNIST images as data source is shown in figure 2.1.

Objective The two networks are optimized over the same objective $f(\boldsymbol{\theta},\boldsymbol{\varphi})$. While the generator tries to minimize the expression, the discriminator tries to maximize it. Mathematically, this is described with the following saddle point problem:

$$\begin{aligned} &\min_{\boldsymbol{\theta}\in\mathbb{R}^n} \max_{\boldsymbol{\varphi}\in\mathbb{R}^m} f(\boldsymbol{\theta},\boldsymbol{\varphi}) \\ =&\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\varphi}} \mathbb{E}_{\mathbf{x}\sim p_d} \log D(\mathbf{x};\boldsymbol{\varphi}) + \mathbb{E}_{\mathbf{z}\sim p_z} \log(1 - D(G(\mathbf{z};\boldsymbol{\theta});\boldsymbol{\varphi})) \end{aligned} \tag{2.4}$$

The original GAN training, proposed by Goodfellow et al. [10], uses an alternating stochastic gradient descent/ ascent approach as shown in Algorithm 1.

Algorithm 1 GAN Training

1: **for** number of training iterations **do**
2: Sample $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior p_z.
3: Sample $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ from data distribution p_d.
4: Update discriminator by ascending its stochastic gradient:

$$\frac{1}{m}\sum_{i=1}^{m} \nabla_{\boldsymbol{\varphi}} [\log D(\mathbf{x}^{(i)}; \boldsymbol{\varphi}) + \log(1 - D(G(\mathbf{z}^{(i)}; \boldsymbol{\theta}); \boldsymbol{\varphi}))]$$

5: Update generator by descending its stochastic gradient:

$$\frac{1}{m}\sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} \log(1 - D(G(\mathbf{z}^{(i)}; \boldsymbol{\theta}); \boldsymbol{\varphi}))$$

6: **end for**

Mathematical evidence on why GANs work In this paragraph, we want to formalize our intuitive understanding of the GAN training. The following argument relies on the assumption that for any fixed generator we have access to an optimal discriminator, which is given by:

$$D_G^*(\mathbf{x}; \boldsymbol{\theta}) = \frac{p_d(\mathbf{x})}{p_d(\mathbf{x}) + p_g(\mathbf{x}; \boldsymbol{\theta})}, \tag{2.5}$$

where p_g is the modeled distribution that is implicitly defined through the generator network G [10]. Using the expression for the optimal discriminator, the training objective in Eq. (2.4) can be re-written as:

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) = \min_{\boldsymbol{\theta}} c(\boldsymbol{\theta}) \tag{2.6}$$

$$= \min_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x}\sim p_d} \log\left(\frac{p_d(\mathbf{x})}{p_d(\mathbf{x}) + p_g(\mathbf{x}; \boldsymbol{\theta})}\right) + \mathbb{E}_{\mathbf{x}\sim p_g} \log\left(\frac{p_g(\mathbf{x}; \boldsymbol{\theta})}{p_d(\mathbf{x}) + p_g(\mathbf{x}; \boldsymbol{\theta})}\right) \tag{2.7}$$

$$= \min_{\boldsymbol{\theta}} -\log(4) + KL\left(p_d \| \frac{p_d + p_g}{2}\right) + KL\left(p_g \| \frac{p_d + p_g}{2}\right) \tag{2.8}$$

$$= \min_{\boldsymbol{\theta}} -\log(4) + JSD(p_d \| p_g) \tag{2.9}$$

Note that the Jensen-Shannon Divergence (JSD) has a single minimum at $p_d = p_g$, which is the global minimum of $c(\boldsymbol{\theta})$. Hence, for an optimal discriminator the minimization of the objective results in a perfectly replicated data generating distribution.

Problems of GANs Generative Adversarial Networks have massively improved state-of-the-art performance in learning complicated probability distributions. Like no other model, they are able to generate impressive results, even on very high-dimensional image data [24, 5, 26]. Despite all their success, GANs are also known to be notoriously hard to train [18]. Many theoretical and practical adjustments to the training procedure have been proposed in order to stabilize it [25, 1, 18]. While most of the modifications are out of the scope of this thesis, we will cover the earliest and probably most popular fix to GAN training: the use of individual loss functions.

From the Saddle Point Problem to the Zero-Sum Game Already in the initial GAN paper [10], Goodfellow pointed out that the saddle point objective of Eq. (2.4) does not work well in practice. He argues that early in learning the discriminator network is able to reject generated samples really well, leading to a saturation of the generator's objective $\log(1 - D(G(\mathbf{z};\boldsymbol{\theta});\boldsymbol{\varphi}))$. To remedy this issue, he proposes to minimize the generator's parameters over the *non-saturating* loss $-\log D(G(\mathbf{z};\boldsymbol{\theta});\boldsymbol{\varphi})$. This results in the following optimization objective:

$$\min_{\boldsymbol{\theta}} f_1(\boldsymbol{\theta},\boldsymbol{\varphi}), \quad \max_{\boldsymbol{\varphi}} f_2(\boldsymbol{\theta},\boldsymbol{\varphi}) \tag{2.10}$$

$$f_1(\boldsymbol{\theta},\boldsymbol{\varphi}) = -\mathbb{E}_{\mathbf{z}\sim p_z} \log D(G(\mathbf{z};\boldsymbol{\theta});\boldsymbol{\varphi}) \tag{2.11}$$

$$f_2(\boldsymbol{\theta},\boldsymbol{\varphi}) = \mathbb{E}_{\mathbf{x}\sim p_d} \log D(\mathbf{x};\boldsymbol{\varphi}) + \mathbb{E}_{\mathbf{z}\sim p_z} \log(1 - D(G(\mathbf{z};\boldsymbol{\theta});\boldsymbol{\varphi})). \tag{2.12}$$

Note that this change preserves the fixed points of the training dynamics, but adjusts the generator's objective to be convex in $D(G(\mathbf{z};\boldsymbol{\theta});\boldsymbol{\varphi})$, instead of concave. Even though this modification leads to serious improvement in practice, it can theoretically no longer be described as a saddle point problem. Instead, the new objective gives rise to a *Zero-Sum* game [15]. However, since the fixed points of the dynamics are preserved, we can further generalize our optimization goal in Eq. (1.2) to handle the new objective. In particular, we do this generalization by defining a *local Nash-equilibrium*, which is a well known notion in game theory [15], and the GAN community [19, 18, 17].

Definition 2.1 (Local Nash-Equilibrium) *The point $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ is a local Nash-equilibrium to the adapted saddle point problem of the form*

$$\begin{cases} \min_{\boldsymbol{\theta}} f_1(\boldsymbol{\theta},\boldsymbol{\varphi}) \\ \max_{\boldsymbol{\varphi}} f_2(\boldsymbol{\theta},\boldsymbol{\varphi}) \end{cases} \tag{2.13}$$

if for $\forall(\boldsymbol{\theta}, \boldsymbol{\varphi}) \in \mathcal{K}_\gamma^*$

$$f_1(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) \leq f_1(\boldsymbol{\theta}, \boldsymbol{\varphi}^*) \tag{2.14}$$
$$f_2(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) \geq f_2(\boldsymbol{\theta}^*, \boldsymbol{\varphi}) \tag{2.15}$$

holds.

Note that for $f_1 = f_2$ the definition of the local Nash-equilibrium reduces to definition 1.1 of the locally optimal saddle point.

2.3 Robust Optimization for Empirical Risk Minimization

The idea of robust optimization [3] is to adapt the optimization procedure such that it becomes more robust against uncertainty which is usually represented by variability in the data. For a loss function $l : \Theta \times \mathcal{X} \to \mathbb{R}$, we consider the typical problem of finding a parameter vector $\boldsymbol{\theta} \in \Theta$ to minimize the risk

$$R(\mathbf{X}; \boldsymbol{\theta}) = \mathbb{E}_P[l(\mathbf{X}; \boldsymbol{\theta})], \tag{2.16}$$

where P denotes the distribution of the data $\mathbf{X} \in \mathcal{X}$. Since the true distribution is unknown, the problem is usually reduced to *empirical* risk minimization. Instead of taking the expected value over the true distribution, the empirical distribution $\hat{P}_n = \left\{\frac{1}{n}\right\}_{i=1}^n$ over the training data $\{\mathbf{X}_1, \ldots, \mathbf{X}_n\} \in \mathcal{X}$ is used.

$$R_{\text{erm}}(\mathbf{X}; \boldsymbol{\theta}) = \mathbb{E}_{\hat{P}_n}[l(\mathbf{X}; \boldsymbol{\theta})] = \frac{1}{n} \sum_{i=1}^n l(\mathbf{X}_i; \boldsymbol{\theta}) \tag{2.17}$$

Robust optimization aims to minimize this quantity while simultaneously being robust against variation in the data, i.e., discrepancy between $\hat{P}_n$ and P. The approach it hereby takes is to consider the *worst* possible distribution $\hat{P}_n$, within the limit of some distance constraint, at training time. It, therefore, becomes more resilient against problems caused by a training set that does not represent the true distribution appropriately. Putting this idea into an objective yields

$$\min_{\boldsymbol{\theta}} \sup_{P \in \mathcal{P}} \left[R_{\text{r}}(\mathbf{X}; \boldsymbol{\theta}, \mathcal{P}) = \left\{ \mathbb{E}_P[l(\mathbf{X}; \boldsymbol{\theta})] : D(P \| \hat{P}_n) \leq \frac{\rho}{n} \right\} \right] \tag{2.18}$$

where $D(P \| \hat{P}_n)$ is a divergence measure between the distribution P and the empirical distribution of the training data $\hat{P}_n$. Hence, we arrive at an objective that is minimized over the model parameter $\boldsymbol{\theta}$ and maximized with respect to the parameters of the distribution P. This is the standard form of

a saddle point problem as defined in 1.1. Contrary to the GAN objective, it has the additional divergence constraint that needs to be fulfilled.

Particularly interesting is the connection between robust optimization and the bias-variance tradeoff in statistical learning. Recent work by Namkoong et al. [21] showed that robust optimization is a good surrogate for the variance-regularized empirical risk

$$R_{\text{erm}}(\mathbf{X};\boldsymbol{\theta}) + c\sqrt{\frac{1}{n}\text{Var}_{\hat{P}_n}(l(\mathbf{X};\boldsymbol{\theta}))}. \tag{2.19}$$

Being a good approximation to this particular quantity is very desirable because it has been shown by [2] that, under appropriate assumptions, the true error is upper bounded with high probability by

$$R(\mathbf{X};\boldsymbol{\theta}) \leq R_{\text{erm}}(\mathbf{X};\boldsymbol{\theta}) + c_1\sqrt{\frac{1}{n}\text{Var}(l(\mathbf{X};\boldsymbol{\theta}))} + \frac{c_2}{n}. \tag{2.20}$$

Therefore, robust optimization is a good surrogate for an upper bound on the true error and holds great promises for reliable risk minimization.

Chapter 3

Preliminaries

3.1 Notation and Definitions

Throughout the thesis, vectors are denoted by lower case bold symbols, scalars by roman symbols and matrices by bold upper case letters. For a vector $\mathbf{v}$, we use $\|\mathbf{v}\|$ to denote the ℓ_2-norm, whereas for a matrix $\mathbf{M}$, we use $\|\mathbf{M}\|$ to denote the spectral norm. We regularly use the short notation $\mathbf{z} := (\boldsymbol{\theta}, \boldsymbol{\varphi})$ to refer to the parameter vector and $\mathbf{H} := \nabla^2 f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ for the Hessian. We use $\mathbf{A} \succeq \mathbf{B}$ for two symmetric matrices $\mathbf{A}$ and $\mathbf{B}$ to express that $\mathbf{A} - \mathbf{B}$ is positive semi-definite.

3.1.1 Stability

Major parts of our theoretical analysis of the saddle point problem rely on the notion of *stability*. It is a well-known concept and tool from non-linear systems theory. In the following, we present a definition of the most important types of stability of dynamic systems [12].

Consider a system with parameters $\mathbf{x} \in \mathrm{R}^n$, whose gradient with respect to the time is given by the function $g(\mathbf{x})$, i.e.,

$$\dot{\mathbf{x}} = g(\mathbf{x}) \tag{3.1}$$

Assume, without loss of generality, that the origin is a critical point (equilibrium) of the system. Hence, it holds that $g(\mathbf{0}) = \mathbf{0}$.

Definition 3.1 ***Stability*** *(Definition 4.1 from [12]): The equilibrium point $\mathbf{x} = \mathbf{0}$ of 3.1 is*

- *stable, if for each $\varepsilon > 0$, there is $\delta = \delta(\varepsilon) > 0$ such that*

$$\|\mathbf{x}_0\| < \delta \Rightarrow \|\mathbf{x}_t\| < \varepsilon, \quad \forall t \geq 0 \tag{3.2}$$

- *unstable if it is not stable.*
- *asymptotically stable if it is stable and δ can be chosen such that*

$$\|\mathbf{x}_0\| < \delta \Rightarrow \lim_{t\to\infty} \|\mathbf{x}_t\| = 0 \tag{3.3}$$

- *exponentially stable if it is asymptotically stable and $\delta, k, \lambda > 0$ can be chosen such that*

$$\|\mathbf{x}_0\| < \delta \Rightarrow \|\mathbf{x}_t\| \leq k\|\mathbf{x}_0\|e^{-\lambda t} \tag{3.4}$$

The system is called stable if we can be sure that it stays within a ball of radius ε when it is initialized within a ball of radius $\delta(\varepsilon)$. This notion alone does not guarantee convergence to the equilibrium point. It might as well orbit around the point as long as it stays within the ε-ball. The stronger concept of asymptotic stability, on the other hand, doesn't allow infinite orbiting around the equilibrium, but requires convergence in the limit of $t \to \infty$.

3.1.2 Critical Points

Definition 3.2 *A critical point $(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})$ of the function $f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ is a point where the gradient of the function is zero, i.e.*

$$\nabla f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) = \mathbf{0}. \tag{3.5}$$

Critical points can be further analyzed by the curvature of the function, i.e., the Hessian of f. The structure of the function f, with its two different input vectors, gives rise to the following block structure of the Hessian.

$$\mathbf{H} = \begin{bmatrix} \nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) & \nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \\ \nabla_{\boldsymbol{\varphi}\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) & \nabla^2_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \end{bmatrix} \tag{3.6}$$

Note that the Hessian is a symmetric matrix and, thus, all its eigenvalues are real numbers.
There are four important types of critical points that can be categorized through the eigenvalues of the Hessian and its individual blocks:

- If all eigenvalues of $\mathbf{H}$ are strictly positive, then the critical point is a local minimum.
- If all eigenvalues of $\mathbf{H}$ are strictly negative, then the critical point is a local maximum.
- If $\mathbf{H}$ has both positive and negative eigenvalues, then the critical point is a saddle point.

- If all eigenvalues of $\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ are strictly positive and all eigenvalues of $\nabla^2_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ are strictly negative, then the critical point is a locally optimal saddle point (local Nash-equilibrium). This claim is formalized in lemma 3.3. Note that the set of locally optimal saddle points is a subset of the general saddle points defined before.

Note that for a function $f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ that is convex in $\boldsymbol{\theta}$ and concave in $\boldsymbol{\varphi}$, all its saddle points are locally optimal and according to the *Minimax Theorem (Sion-Kakutani)* it holds that.

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) = \max_{\boldsymbol{\varphi}} \min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}). \tag{3.7}$$

Since we are considering a more general saddle point problem, where $f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ is neither convex in $\boldsymbol{\theta}$ nor concave in $\boldsymbol{\varphi}$, these guarantees don't hold and a more complex analysis of the saddle points is necessary.

Characterization of locally optimal saddle points With the use of the non-degenerate assumption on the Hessian matrices (Assumption 3.5), we are able to establish sufficient conditions on $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ to be a locally optimal saddle point.

Lemma 3.3 *Suppose that f satisfies Assumption 3.5; then, $\mathbf{z}^* := (\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ is a locally optimal saddle point on $\mathcal{K}^*_\gamma$ if and only if the gradient with respect to the parameters is zero, i.e.*

$$\nabla f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) = \mathbf{0}, \tag{3.8}$$

and the second derivative at $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ is positive definite in $\boldsymbol{\theta}$ and negative definite in $\boldsymbol{\varphi}$, i.e. there exist $\mu_{\boldsymbol{\theta}}, \mu_{\boldsymbol{\varphi}} > 0$ such that

$$\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) \succ \mu_{\boldsymbol{\theta}} \mathbf{I}, \qquad \nabla^2_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) \prec -\mu_{\boldsymbol{\varphi}} \mathbf{I}. \tag{3.9}$$

Proof From definition 1.1 follows that a locally optimal saddle point $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) \in \mathcal{K}^*_\gamma$ is a point for which the following two conditions hold:

$$f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}) \leq f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \quad \text{and} \quad f(\boldsymbol{\theta}, \boldsymbol{\varphi}^*) \geq f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \quad \forall (x, y) \in \mathcal{K}^*_\gamma \tag{3.10}$$

Hence, $\boldsymbol{\theta}$ is a local minimizer of f and $\boldsymbol{\varphi}$ is a local maximizer. We therefore, without loss of generality, prove the statement of the lemma only for the minimizer $\boldsymbol{\theta}$, namely that

1. $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}) = \mathbf{0} \quad \forall \boldsymbol{\varphi}$ s.t. $\|\boldsymbol{\varphi} - \boldsymbol{\varphi}^*\| \leq \gamma$
2. $\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}) \succ \mu_{\boldsymbol{\theta}} \mathbf{I} \quad \forall \boldsymbol{\varphi}$ s.t. $\|\boldsymbol{\varphi} - \boldsymbol{\varphi}^*\| \leq \gamma, \mu_{\boldsymbol{\theta}} > 0$.

The proof for the maximizer $\boldsymbol{\varphi}$ directly follows from this.

1. If we assume that $\nabla_{\theta} f(\theta^*, \varphi) \neq \mathbf{0}$, then there exists a feasible direction $\mathbf{d} \in \mathbb{R}^n$ such that $\nabla_{\theta}^{\top} f(\theta^*, \varphi)\mathbf{d} < 0$ and we can find a step size $\alpha > 0$ for $\theta(\alpha) = \theta^* + \alpha\mathbf{d}$ s.t. $\alpha\|\mathbf{d}\| \leq \gamma$. Now with the use of Taylor's theorem, we arrive at the following inequality

$$f(\theta(\alpha), \varphi) = f(\theta^*, \varphi) + \underbrace{\alpha \nabla_{\theta}^{\top} f(\theta^*, \varphi)\mathbf{d}}_{<0} + \underbrace{\mathcal{O}(\alpha^2)}_{\to 0 \text{ for sufficiently small } \alpha} \tag{3.11}$$

$$< f(\theta^*, \varphi) \tag{3.12}$$

which contradicts that $f(\theta^*, \varphi)$ is a local minimizer. Hence, $\nabla_{\theta} f(\theta^*, \varphi) = \mathbf{0}$ is a necessary condition for a local minimizer.

2. To prove the second statement, we again make use of Taylor's theorem with $\alpha > 0, \mathbf{d} \in \mathbb{R}^n$ for $\theta(\alpha) = \theta^* + \alpha\mathbf{d}$ s.t. $\alpha\|\mathbf{d}\| \leq \gamma$.

$$f(\theta(\alpha), \varphi) = f(\theta^*, \varphi) + \frac{1}{2}\alpha^2 \mathbf{d}^{\top} \nabla_{\theta}^2 f(\theta^*, \varphi)\mathbf{d} + \underbrace{\mathcal{O}(\alpha^3)}_{\to 0 \text{ for sufficiently small } \alpha} \tag{3.13}$$

The inequality $f(\theta(\alpha), \varphi) \geq f(\theta^*, \varphi)$ holds if and only if $\mathbf{d}^{\top} \nabla_{\theta}^2 f(\theta^*, \varphi)\mathbf{d} \geq 0$, which means that $\nabla_{\theta}^2 f(\theta^*, \varphi)$ needs to be positive semi-definite. Because we assume that the second derivative is non-degenerate (Assumption 3.5), we arrive at the sufficient condition that

$$\nabla_{\theta}^2 f(\theta^*, \varphi) \succ \mu_{\theta} \mathbf{I} \quad \text{, with } \mu_{\theta} > 0 \tag{3.14}$$

which concludes the proof. □

3.2 Assumptions

For the upcoming theoretical analysis of different optimization procedures on the saddle point problem, we need to make several assumptions on the function $f(\theta, \varphi)$.

Smoothness We assume that the function $f(\theta, \varphi)$ is smooth, which is formalized in the following assumption.

Assumption 3.4 (Smoothness) *We assume that $f(\mathbf{z}) := f(\theta, \varphi)$ is a C^2 function, and that its gradient and Hessian are Lipschitz with respect to the*

parameters $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$, i.e., we assume that the following inequalities hold:

$$\|\nabla f(\mathbf{z}) - \nabla f(\widetilde{\mathbf{z}})\| \leq L_{\mathbf{z}} \|\mathbf{z} - \widetilde{\mathbf{z}}\| \tag{3.15}$$
$$\|\nabla^2 f(\mathbf{z}) - \nabla^2 f(\widetilde{\mathbf{z}})\| \leq \rho_{\mathbf{z}} \|\mathbf{z} - \widetilde{\mathbf{z}}\| \tag{3.16}$$
$$\|\nabla f(\boldsymbol{\theta}, \boldsymbol{\varphi}) - \nabla f(\widetilde{\boldsymbol{\theta}}, \boldsymbol{\varphi})\| \leq L_{\boldsymbol{\theta}} \|\boldsymbol{\theta} - \widetilde{\boldsymbol{\theta}}\| \tag{3.17}$$
$$\|\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) - \nabla^2_{\boldsymbol{\theta}} f(\widetilde{\boldsymbol{\theta}}, \boldsymbol{\varphi})\| \leq \rho_{\boldsymbol{\theta}} \|\boldsymbol{\theta} - \widetilde{\boldsymbol{\theta}}\| \tag{3.18}$$
$$\|\nabla f(\boldsymbol{\theta}, \boldsymbol{\varphi}) - \nabla f(\boldsymbol{\theta}, \widetilde{\boldsymbol{\varphi}})\| \leq L_{\boldsymbol{\varphi}} \|\boldsymbol{\varphi} - \widetilde{\boldsymbol{\varphi}}\| \tag{3.19}$$
$$\|\nabla^2_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) - \nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \widetilde{\boldsymbol{\varphi}})\| \leq \rho_{\boldsymbol{\varphi}} \|\boldsymbol{\varphi} - \widetilde{\boldsymbol{\varphi}}\| \tag{3.20}$$
$$\|\nabla_{\boldsymbol{\theta}} f(\mathbf{z})\| \leq \ell_{\boldsymbol{\theta}},\ \|\nabla_{\boldsymbol{\varphi}} f(\mathbf{z})\| \leq \ell_{\boldsymbol{\varphi}},\ \|\nabla_{\mathbf{z}} f(\mathbf{z})\| \leq \ell_{\mathbf{z}} \tag{3.21}$$

Non-Degenerate For our theoretical guarantees, we require a non-degenerate assumption on the Hessian matrix of f.

Assumption 3.5 (Non-degenerate Hessian) *We assume that the matrices $\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ and $\nabla^2_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ are non-degenerate for all $(\boldsymbol{\theta} \in \mathbb{R}^n, \boldsymbol{\varphi} \in \mathbb{R}^m)$.*

Chapter 4

Saddle Point Optimization

4.1 Gradient-Based Optimization

The saddle point problem of Eq. (1.1) is usually solved by a gradient-based optimization method. In particular, the problem is split into two individual optimization problems that can be solved simultaneously with gradient steps in different directions, i.e.,

$$\begin{bmatrix} \boldsymbol{\theta}_{t+1} \\ \boldsymbol{\varphi}_{t+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\theta}_t \\ \boldsymbol{\varphi}_t \end{bmatrix} + \eta \begin{bmatrix} -\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \\ \nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \end{bmatrix}. \tag{4.1}$$

This method is used in various applications of saddle point optimization, as explained in chapter 2. Stationary points of the above dynamics are critical points of the function f (cf. 3.1.2). More precisely, the point $\mathbf{z}^* := (\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ is a stationary point of the above iterates if the mapping function, implicitly defined through the recursion, maps z^* to itself, i.e., if $\nabla_{\mathbf{z}} f(\mathbf{z}^*) = \mathbf{0}$. With $\mathcal{F}_{\mathcal{G}} \subset \mathbb{R}^n \times \mathbb{R}^m$ we denote the set of all stationary points of the gradient iterates, i.e.,

$$\mathcal{F}_{\mathcal{G}} := \{\mathbf{z} \in \mathbb{R}^n \times \mathbb{R}^m | \nabla_{\mathbf{z}} f(\mathbf{z}^*) = \mathbf{0}\}. \tag{4.2}$$

4.2 Linear transformed Gradient Steps

Linear transformation of the gradient updates has been shown to accelerate optimization for various types of problems, including the saddle point problem. Such methods can be expressed by the following recursion

$$\begin{bmatrix} \boldsymbol{\theta}_{t+1} \\ \boldsymbol{\varphi}_{t+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\theta}_t \\ \boldsymbol{\varphi}_t \end{bmatrix} + \eta \mathbf{A}_{\boldsymbol{\theta},\boldsymbol{\varphi}} \begin{bmatrix} -\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \\ \nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \end{bmatrix} \tag{4.3}$$

where $\mathbf{A}_{\boldsymbol{\theta},\boldsymbol{\varphi}}$ is a symmetric $((n+m) \times (n+m))$-matrix. Different optimization methods use a different linear transformation $\mathbf{A}_{\boldsymbol{\theta},\boldsymbol{\varphi}}$. Most common op-

timization schemes use a block diagonal matrix $\mathbf{A}_{\theta,\varphi} = \begin{bmatrix} \mathcal{A} & 0 \\ 0 & \mathcal{B} \end{bmatrix}$. Table 4.1 illustrates the choice of $\mathbf{A}_{\theta,\varphi}$ for different optimizer.

Table 4.1: Update matrices for commonly used optimization schemes.

	Formula	positive definite?
Gradient Descent	$\mathcal{A}_t = I$ $\mathcal{B}_t = I$	Yes.
Adagrad [9]	$\mathcal{A}_{t,ii} = \left(\sqrt{\sum_{\tau=1}^{t} \left(\nabla_{\theta_i} f(\theta_\tau, \varphi_\tau)\right)^2 + \varepsilon}\right)^{-1}$ $\mathcal{B}_{t,ii} = \left(\sqrt{\sum_{\tau=1}^{t} \left(\nabla_{\varphi_i} f(\theta_\tau, \varphi_\tau)\right)^2 + \varepsilon}\right)^{-1}$	Yes.
Newton	$\mathcal{A}_t = \left(\nabla_\theta^2 f(\theta_t, \varphi_t)\right)^{-1}$ $\mathcal{B}_t = \left(\nabla_\varphi^2 - f(\theta_t, \varphi_t)\right)^{-1}$	Around local min of $f(\theta, \varphi)$. Around local max of $f(\theta, \varphi)$.
Saddle-Free Newton [8]	$\mathcal{A}_t = \left\lvert\nabla_\theta^2 f(\theta_t, \varphi_t)\right\rvert^{-1}$ $\mathcal{B}_t = \left\lvert\nabla_\varphi^2 f(\theta_t, \varphi_t)\right\rvert^{-1}$	Yes.

4.3 Asymptotic Behavior of Gradient Iterations

There are three different asymptotic scenarios for the gradient iterations in Eq. (4.1):

1. Divergence, i.e., $\lim_{t\to\infty} \|\nabla f(\theta_t, \varphi_t)\| \to \infty$.
2. Being trapped in a loop, i.e., $\lim_{t\to\infty} \|\nabla f(\theta_t, \varphi_t)\| > 0$.
3. Convergence to a stationary point of the gradient updates, i.e., $\lim_{t\to\infty} \|\nabla f(\theta_t, \varphi_t)\| = 0$.

Up to the best of our knowledge, there is no global convergence guarantee for general saddle point optimization. Typical convergence guarantees require convex-concavity or somehow quasiconvex-quasiconcavity of f [6]. Instead of global convergence, we consider the rather modest analysis of convergent iterations for the upcoming chapters of the thesis. Hence, we focus on the third outlined case where we are sure that the method converges to some stationary point (θ, φ) for which holds $\nabla f(\theta, \varphi) = \mathbf{0}$. The question of interest for the upcoming chapters is whether such a sequence always yields a locally optimal saddle point as defined in 1.1?

4.4 Convergence Analysis on a Convex-Concave Objective

In this section, we want to analyze the differences between common optimization methods for the saddle point problem in Eq. (1.1). As described before, it is very difficult to make any global argument for a non convex-concave function f. In order to still get an intuition about the different optimization schemes on the saddle point problem, we consider (in this section) a function $f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ that is convex in $\boldsymbol{\theta}$ and concave in $\boldsymbol{\varphi}$, i.e., we assume that

$$\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \succeq \alpha \mathbf{I} \quad \text{and} \quad \nabla^2_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \preceq -\alpha \mathbf{I} \qquad \alpha > 0 \tag{4.4}$$

for all $(\boldsymbol{\theta}, \boldsymbol{\varphi}) \in \mathbb{R}^n \times \mathbb{R}^m$. Moreover, we assume that the function f has a unique critical point $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$, which is the solution to the saddle point problem (1.1). In this simplified setting, we want to obtain insights to the convergence behavior of commonly used optimization methods. In particular, we are interested in the convergence rate to the optimal saddle point $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$.

4.4.1 Gradient-Based Optimizer

The following Lemma shows that the convergence rate of gradient descent is independent of cross-dependencies between the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ in the strictly convex-concave setting. By choosing a proper step size γ we are guaranteed to converge to the saddle point $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$.

Lemma 4.1 *Suppose that $\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \succeq \alpha \mathbf{I}$ and $\nabla^2_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \preceq -\alpha \mathbf{I}$ with $\alpha > 0$, and assumptions 3.4 and 3.5 hold. Let $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ be the unique solution of the saddle point problem, then t gradient steps obtain*

$$\| \begin{bmatrix} \boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \leq (1 + \gamma(L_{\mathbf{z}}\gamma - 2\alpha))^t \| \begin{bmatrix} \boldsymbol{\theta}^{(0)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(0)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \tag{4.5}$$

Proof See appendix 7.5. □

4.4.2 Gradient-Based Optimizer with Preconditioning Matrix

In non-convex minimization it is common practice to use a gradient-based optimizer with a preconditioning matrix, such as for example Adagrad [9]. Usually this matrix is diagonal and positive definite. These methods have been adapted to be used in the saddle point problem, which yields the simultaneous update step of Eq. (4.3). In this section, we try to understand the convergence behavior of this more general update step on the convex-concave objective. The following lemma establishes an expression for the distance to the unique saddle point after one iteration of the update in Eq. (4.3).

An interesting observation is that, as opposed to gradient descent before, the convergence depends on the cross-dependencies between the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$.

Lemma 4.2 *Suppose that* $\nabla^2_{\boldsymbol{\theta}} f \succeq \eta \mathbf{I}$ *and* $\nabla^2_{\boldsymbol{\varphi}} f \preceq -\gamma \mathbf{I}$ *with* $\gamma > 0$*, and assumptions 3.4 and 3.5 hold. The step size matrices A and B are diagonal, positive semi-definite matrices with*

$$\alpha_{min}\mathbf{I} \preceq A \preceq \alpha_{max}\mathbf{I} \tag{4.6}$$

$$\beta_{min}\mathbf{I} \preceq B \preceq \beta_{max}\mathbf{I} \tag{4.7}$$

with a constant

$$R := \frac{\min(\alpha_{min}, \beta_{min})}{\max(\alpha_{max}, \beta_{max})}. \tag{4.8}$$

Let $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ *be the unique solution of the saddle point problem, then one modified gradient step of Eq.* (4.3) *with step size*

$$\eta = \frac{\gamma R}{L_{\mathbf{z}} \max(\alpha_{max}, \beta_{max})} \tag{4.9}$$

leads to

$$\begin{aligned} \| \begin{bmatrix} \boldsymbol{\theta}^+ - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^+ - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 = (1 - \frac{\gamma^2}{L_{\mathbf{z}}} R^2) \| \begin{bmatrix} \boldsymbol{\theta} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \\ - 2\gamma \sum_{i=1}^{|\boldsymbol{\theta}|} \sum_{j=1}^{|\boldsymbol{\varphi}|} (A_{ii} - B_{jj})(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)(\boldsymbol{\varphi}_j - \boldsymbol{\varphi}_j^*) \nabla_{\boldsymbol{\theta}_i \boldsymbol{\varphi}_j} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) \end{aligned} \tag{4.10}$$

Proof See appendix 7.5. □

The Lemma provides an upper bound for the convergence of a general optimization scheme that uses diagonal step-matrices, as for example Adagrad. As we can see, the right-hand-side of inequality 4.10 consists of a summation of two terms. First, one that, depending on the problem and the chosen step size, is smaller than $\| \begin{bmatrix} \boldsymbol{\theta} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2$. It therefore drives the new parameters closer to the optimum. The value of the second summand is rather unclear. In general we can not make any statement about the sign of the products $(\mathbf{A}_{ii} - B_{jj})(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)(\boldsymbol{\varphi}_i - \boldsymbol{\varphi}_i^*) \nabla_{\boldsymbol{\theta}_i \boldsymbol{\varphi}_j} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})$. Hence, from this form it is not possible to tell whether this cross-dependency term increases or decreases the upper bound. What we can see, however, is that the product of any *ij-combination* vanishes, if either $\boldsymbol{\theta}_i$ or $\boldsymbol{\varphi}_j$ gets close to its optimum value. This ensures that for $\boldsymbol{\theta}$, $\boldsymbol{\varphi}$ close enough around the optimum the algorithm converges to $\boldsymbol{\theta}^*$, $\boldsymbol{\varphi}^*$, respectively.

4.4.3 Experiment

Let's consider a simple convex-concave saddle point problem of the form

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^n}\max_{\boldsymbol{\varphi}\in\mathbb{R}^n}(f(\boldsymbol{\theta},\boldsymbol{\varphi}) = \frac{\alpha}{2}\|\boldsymbol{\theta}\|^2 - \frac{\beta}{2}\|\boldsymbol{\varphi}\|^2 + \boldsymbol{\theta}^\top \mathbf{M}\boldsymbol{\varphi}) \tag{4.11}$$

where $\mathbf{M} \in \mathbb{R}^{n\times n}$ is a diagonal matrix. The Jacobian and Hessian of the objective are given, respectively, by

$$\nabla f(\boldsymbol{\theta},\boldsymbol{\varphi}) = \begin{bmatrix} \alpha\boldsymbol{\theta} + \mathbf{M}\boldsymbol{\varphi} \\ -\beta\boldsymbol{\varphi} + \mathbf{M}^\top\boldsymbol{\theta} \end{bmatrix} \qquad \nabla^2 f(\boldsymbol{\theta},\boldsymbol{\varphi}) = \begin{bmatrix} \alpha\mathbf{I} & \mathbf{M} \\ \mathbf{M}^\top & -\beta\mathbf{I} \end{bmatrix} \tag{4.12}$$

Experimentally, we want to compare the convergence to the saddle point of Eq. (4.11) for gradient descent of Eq. (4.1) (GD) and Adagrad (special form of Eq. (4.3)).
Lemma 4.1 indicates that with a sufficiently small step size GD always converges to the saddle point. Since Adagrad, however, is part of the class of algorithms that use a more general update step matrix, we need to use the convergence upper bound given by Lemma 4.2 - which, in general, does not provide the same guarantees. The problematic part about this bound is the additive cross-dependency term given by

$$\gamma \sum_{i=1}^{n} (\mathbf{B}_{ii} - \mathbf{A}_{ii})(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)(\boldsymbol{\varphi}_i - \boldsymbol{\varphi}_i^*)\mathbf{M}_{ii}. \tag{4.13}$$

In general, it is not possible to make a statement about the sign of this term.

To experimentally show that the cross-dependency term can either improve or worsen the convergence rate of Adagrad, we design specific problems based on the knowledge about the factors in expression 4.13. Since it is an additive term, we are interested in its sign to determine whether it increases or decreases the bound. It consists of four different factors, with each of it being either positive or negative. Therefore, we need to make some assumptions in order to investigate the sign change behavior of the whole term. In particular, we would like to fix the sign of $(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)(\boldsymbol{\varphi}_i - \boldsymbol{\varphi}_i^*)\mathbf{M}_{ii}$ for all $i \in 1,\dots,n$ such that by varying the diagonal values of the step matrices $\mathbf{A}$ and $\mathbf{B}$ we can achieve a change in the sign of the term. While this is trivial to achieve for the cross-dependency matrix $\mathbf{M}$ (we can choose its values arbitrarily), it is not obvious how to fix the sign of the other two factors. However, we can approximately make them positive by initializing all parameters, such that

$$\boldsymbol{\theta}_i^{(0)} \gg \boldsymbol{\theta}_i^* \quad \text{and} \quad \boldsymbol{\varphi}_i^{(0)} \gg \boldsymbol{\varphi}_i^* \quad i \in 1,\dots,n. \tag{4.14}$$

Hence, by choosing all diagonal values of $\mathbf{M}$ to be greater than zero, we can see that the cross-term dependency improves the algorithm when $\mathbf{A}_{ii} > \mathbf{B}_{ii}$

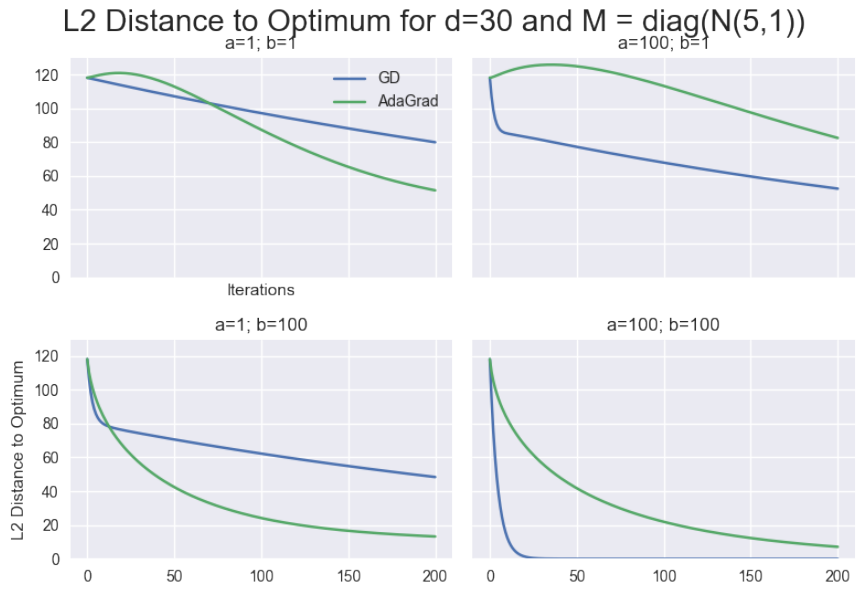


Figure 4.1: L2 Distance to the saddle-point over iterations of the optimization procedure of the problem formulation in equation 4.11 with $\boldsymbol{\theta}, \boldsymbol{\varphi} \in \mathbb{R}^{30}$. The different plots show varying values for $\alpha = a$ and $\beta = b$ while the diagonal cross-dependency matrix $\mathbf{M}$ has fixed values coming from $\mathcal{N}(5,1)$.

for all $i \in 1, \ldots, n$. The preconditioning matrices $\mathbf{A}$ and $\mathbf{B}$ for Adagrad depend on the history of the optimization process (cf. table 4.1). Their diagonal values are approximately inversely proportional to the sum of squared past gradients. Therefore, extreme parameter updates get dampened while parameters that get few or small updates receive higher learning rates. By using the knowledge about the Jacobian of the objective from equation 4.12, we can influence the values of the step matrices $\mathbf{A}$ and $\mathbf{B}$ by varying the parameters α and β. Intuitively, a large value of α together with a small β should lead to a positive sign of the cross-dependency term and, therefore, worsen Adagrad compared to GD.
The results of the experiment, shown in figure 4.1, are in accordance with this intuition. The parameters of this experiment are chosen based on the argumentation above, i.e., large positive initial values for $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ and positive values for the matrix $\mathbf{M}$. The variation of the α and β parameterization shows the anticipated behavior. If $\alpha \gg \beta$ (plot top right), the cross-dependency term hinders Adagrad to find the optimal parameters, and thus it is outperformed by GD. Conversely, if $\beta \gg \alpha$ (plot bottom left), the additive term becomes negative and improves the performance of Adagrad.

The designed experiment provides evidence to the argument arising from Lemma 4.2, namely that Adagrad can either benefit or be hurt by the cross-dependency term, depending on the specific parameterization of the convex-concave saddle point problem.

Chapter 5

Stability Analysis

5.1 Local Stability

In general, a stationary point of the gradient iterates in Eq. (4.1) can be either stable or unstable, where the notion of stability is defined as in Def. 3.1. Stability characterizes the behavior of the optimization dynamics in a local region around a stationary point. Within an epsilon-ball around a stable stationary point, successive iterates of the method are not able to escape the region. Conversely, for unstable points, successive iterates can escape this region.

Let's consider the gradient iterations of Eq. (4.1) as a dynamical system with the following mapping function

$$\mathcal{G}(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) = \begin{bmatrix} \boldsymbol{\theta}_t \\ \boldsymbol{\varphi}_t \end{bmatrix} + \eta \begin{bmatrix} -\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \\ \nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \end{bmatrix} \tag{5.1}$$

and assume that it has a stationary point at $\mathbf{z}^* := (\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$. Then, we can linearize the system by a first-order Taylor approximation around this point, i.e.,

$$\begin{aligned} \mathcal{G}(\mathbf{z}) &\approx \mathcal{G}(\mathbf{z}^*) + \nabla^\top \mathcal{G}(\mathbf{z}^*)(\mathbf{z} - \mathbf{z}^*) && (5.2) \\ &= \mathbf{z}^* + \nabla^\top \mathcal{G}(\mathbf{z}^*)(\mathbf{z} - \mathbf{z}^*). && (5.3) \end{aligned}$$

Obviously, this approximation is only valid for points $\mathbf{z}$ in a small neighborhood around the critical point $\mathbf{z}^*$. To quantify the previously stated intuition about stability, we can say that a fixed-point $\mathbf{z}^*$ is unstable if successive iterates of (4.1) increase the distance to the point $\mathbf{z}^*$. Conversely, a fixed point $\mathbf{z}^*$ is stable if

$$\left\| \frac{\mathbf{z}_t - \mathbf{z}^*}{\mathbf{z}_{t-1} - \mathbf{z}^*} \right\| \leq 1 \tag{5.4}$$

which, with the help of the linearization (5.2), leads to the following expression:

$$\left\| \frac{\mathbf{z}_t - \mathbf{z}^*}{\mathbf{z}_{t-1} - \mathbf{z}^*} \right\| = \left\| \frac{\mathcal{G}(\mathbf{z}_{t-1}) - \mathbf{z}^*}{\mathbf{z}_{t-1} - \mathbf{z}^*} \right\| \tag{5.5}$$

$$\approx \left\| \frac{\nabla \mathcal{G}(\mathbf{z}^*)(\mathbf{z}_{t-1} - \mathbf{z}^*)}{\mathbf{z}_{t-1} - \mathbf{z}^*} \right\| = \|\nabla \mathcal{G}(\mathbf{z}^*)\| \leq 1. \tag{5.6}$$

Hence, we can analyze the stability of a stationary point of the gradient iterates by the absolute value of its Jacobian $\nabla \mathcal{G}(\mathbf{z}^*)$. In particular, a stationary point $\mathbf{z}^*$ is locally stable if all the eigenvalues of the corresponding Jacobian $\nabla \mathcal{G}(\mathbf{z}^*)$ lie inside the unit sphere.

5.2 Convergence to Non-Stable Stationary Points

With the notion of stability, we are able to analyze the asymptotic behavior of the gradient method. It has been shown by [14] that for a non-convex minimization problem of the form

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^n} f(\boldsymbol{\theta}) \tag{5.7}$$

gradient descent with random initialization almost surely converges to a stable stationary point of its dynamics. The next lemma extends this result to gradient iterations of Eq. (4.1) for saddle point problems.

Lemma 5.1 (Random Initialization) *Suppose that assumptions 3.4 and 3.5 hold. Consider gradient iterations of Eq.* (4.1) *starting from a random initial point. If the iterations converge to a stationary point, then the stationary point is almost surely stable.*

Proof The following Lemma 5.2 proves that the gradient update from Eq. (4.1) for the saddle point problem is a diffeomorphism. The remaining part of the proof follows directly from theorem 4.1 from [14]. □

Lemma 5.2 *The gradient mapping for the saddle point problem*

$$\mathcal{G}(\boldsymbol{\theta}, \boldsymbol{\varphi}) = (\boldsymbol{\theta}, \boldsymbol{\varphi}) + \eta(-\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}), \nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi})) \tag{5.8}$$

with step size $\eta < \min\left(\frac{1}{L_x}, \frac{1}{L_y}, \frac{1}{\sqrt{2}L_z}\right)$ *is a diffeomorphism.*

Proof The following proof is very much based on the proof of proposition 4.5 from [14].
A necessary condition for a diffeomorphism is bijectivity. Hence, we need to check that $\mathcal{G}$ is 1. injective, and 2. surjective for $\eta < \min\left(\frac{1}{L_x}, \frac{1}{L_y}, \frac{1}{\sqrt{2}L_z}\right)$.

1. Consider two points $\mathbf{z} := (\boldsymbol{\theta}, \boldsymbol{\varphi}), \widetilde{\mathbf{z}} := (\widetilde{\boldsymbol{\theta}}, \widetilde{\boldsymbol{\varphi}}) \in \mathcal{K}_\gamma$ for which

$$\mathcal{G}(\mathbf{z}) = \mathcal{G}(\widetilde{\mathbf{z}}) \tag{5.9}$$

holds. Then, we have that

$$\mathbf{z} - \widetilde{\mathbf{z}} = \eta \begin{bmatrix} -\nabla_{\boldsymbol{\theta}} f(\mathbf{z}) \\ \nabla_{\boldsymbol{\varphi}} f(\mathbf{z}) \end{bmatrix} - \eta \begin{bmatrix} -\nabla_{\boldsymbol{\theta}} f(\widetilde{\mathbf{z}}) \\ \nabla_{\boldsymbol{\varphi}} f(\widetilde{\mathbf{z}}) \end{bmatrix} \tag{5.10}$$

$$= \eta \begin{bmatrix} -\nabla_{\boldsymbol{\theta}} f(\mathbf{z}) + \nabla_{\boldsymbol{\theta}} f(\widetilde{\mathbf{z}}) \\ \nabla_{\boldsymbol{\varphi}} f(\mathbf{z}) - \nabla_{\boldsymbol{\varphi}} f(\widetilde{\mathbf{z}}) \end{bmatrix}. \tag{5.11}$$

Note that

$$\|\nabla_{\boldsymbol{\theta}} f(\mathbf{z}) - \nabla_{\boldsymbol{\theta}} f(\widetilde{\mathbf{z}})\| \leq \|\nabla f(\mathbf{z}) - \nabla f(\widetilde{\mathbf{z}})\| \leq L_z \|\mathbf{z} - \widetilde{\mathbf{z}}\| \tag{5.12}$$

$$\|\nabla_{\boldsymbol{\varphi}} f(\mathbf{z}) - \nabla_{\boldsymbol{\varphi}} f(\widetilde{\mathbf{z}})\| \leq \|\nabla f(\mathbf{z}) - \nabla f(\widetilde{\mathbf{z}})\| \leq L_z \|\mathbf{z} - \widetilde{\mathbf{z}}\|, \tag{5.13}$$

from which follows that

$$\|\mathbf{z} - \widetilde{\mathbf{z}}\| \leq \eta \sqrt{L_z^2 \|\mathbf{z} - \widetilde{\mathbf{z}}\|^2 + L_z^2 \|\mathbf{z} - \widetilde{\mathbf{z}}\|^2} \tag{5.14}$$

$$= \sqrt{2} \eta L_z \|\mathbf{z} - \widetilde{\mathbf{z}}\|. \tag{5.15}$$

For $0 < \eta < \frac{1}{\sqrt{2} L_z}$ this means $\mathbf{z} = \widetilde{\mathbf{z}}$, and therefore g is injective.

2. We're going to show that $\mathcal{G}$ is surjective by constructing an explicit inverse function for both optimization problems individually. As suggested by [14], we make use of the proximal point algorithm on the function $-f$ for the parameters $\boldsymbol{\theta}, \boldsymbol{\varphi}$, individually.
For the parameter $\boldsymbol{\theta}$ the proximal point mapping of $-f$ centered at $\widetilde{\boldsymbol{\theta}}$ is given by

$$\boldsymbol{\theta}(\widetilde{\boldsymbol{\theta}}) = \arg\min_{\boldsymbol{\theta}} \underbrace{\frac{1}{2} \|\boldsymbol{\theta} - \widetilde{\boldsymbol{\theta}}\|^2 - \eta f(\boldsymbol{\theta}, \boldsymbol{\varphi})}_{h(\boldsymbol{\theta})} \tag{5.16}$$

Moreover, note that $h(\boldsymbol{\theta})$ is strongly convex in $\boldsymbol{\theta}$ if $\eta < \frac{1}{L_x}$:

$$(\nabla_{\boldsymbol{\theta}} h(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} h(\widehat{\boldsymbol{\theta}}))^\top (\boldsymbol{\theta} - \widehat{\boldsymbol{\theta}}) \tag{5.17}$$

$$= (\boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) - \widehat{\boldsymbol{\theta}} + \eta \nabla_{\boldsymbol{\theta}} f(\widehat{\boldsymbol{\theta}}, \boldsymbol{\varphi}))^\top (\boldsymbol{\theta} - \widehat{\boldsymbol{\theta}}) \tag{5.18}$$

$$= \|\boldsymbol{\theta} - \widehat{\boldsymbol{\theta}}\|^2 - \eta (\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) - \nabla_{\boldsymbol{\theta}} f(\widetilde{\boldsymbol{\theta}}, \boldsymbol{\varphi}))^\top (\boldsymbol{\theta} - \widehat{\boldsymbol{\theta}}) \tag{5.19}$$

$$\geq (1 - \eta L_x) \|\boldsymbol{\theta} - \widehat{\boldsymbol{\theta}}\|^2 \tag{5.20}$$

Hence, the function $h(\boldsymbol{\theta})$ has a unique minimizer, given by

$$0 \overset{!}{=} \nabla_{\boldsymbol{\theta}} h(\boldsymbol{\theta}) = \boldsymbol{\theta} - \widetilde{\boldsymbol{\theta}} - \eta \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \tag{5.21}$$

$$\Rightarrow \widetilde{\boldsymbol{\theta}} = \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \tag{5.22}$$

which means that there is a unique mapping from $\boldsymbol{\theta}$ to $\widetilde{\boldsymbol{\theta}}$ under the gradient mapping $\mathcal{G}$ if $\eta < \frac{1}{L_x}$.
The same line of reasoning can be applied to the parameter $\boldsymbol{\varphi}$ with the negative proximal point mapping of $-f$ centered at $\widetilde{\boldsymbol{\varphi}}$, i.e.

$$\boldsymbol{\varphi}(\widetilde{\boldsymbol{\varphi}}) = \arg\max_{\boldsymbol{\varphi}} \underbrace{-\frac{1}{2}\|\boldsymbol{\varphi} - \widetilde{\boldsymbol{\varphi}}\|^2 - \eta f(\boldsymbol{\theta}, \boldsymbol{\varphi})}_{h(\boldsymbol{\varphi})} \tag{5.23}$$

Similarly as before, we can observe that $h(\boldsymbol{\varphi})$ is strictly concave for $\eta < \frac{1}{L_y}$ and that the unique minimizer of $h(\boldsymbol{\varphi})$ yields the $\boldsymbol{\varphi}$ update step of $\mathcal{G}$. From this, we conclude that the mapping $\mathcal{G}$ is surjective for $(\boldsymbol{\theta}, \boldsymbol{\varphi})$ if $\eta < \min\left(\frac{1}{L_x}, \frac{1}{L_y}\right)$

Observing that under assumption 3.5, $\mathcal{G}^{-1}$ is continuously differentiable concludes the proof that $\mathcal{G}$ is a diffeomorphism. □

Conclusion The result of lemma 5.1 implies that a convergent sequence of the gradient iterates almost surely yields a stable stationary point of its dynamics. Remembering that our analysis is solely concerned with convergent sequences (cf. section 4.3), the result suggests that we only need to consider stable stationary points of the gradient dynamics. In a minimization problem, as in Eq. (5.7), this result implies that if the gradient dynamics of the form

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t) \tag{5.24}$$

converge to a solution $\boldsymbol{\theta}^*$, this solution is not just a stable stationary point but also a minimizer of the function f. This is due to the fact that the Jacobian of the update in Eq. (5.24) is given by:

$$\mathbf{J}(\boldsymbol{\theta}_t) = \mathbf{I} - \eta \nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t) \tag{5.25}$$

As mentioned earlier, a point is stable for a certain dynamic only if the eigenvalues of its Jacobian lie within the unit-sphere. From the above equation, we see that this is only possible - with a sufficiently small step size - if $\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t)$ is a Hurwitz matrix. Since any Hessian matrix is symmetric, the eigenvalues of $\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t)$ are real numbers. Therefore, the stability condition reduces to $\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t)$ being a positive definite matrix. As defined in 3.1.2, this implies that $\boldsymbol{\theta}_t$ is a minimum of the function f.
Based on the intuition from the minimization problem, one can jump to a quick conclusion and claim that since we do simultaneous minimization/ maximization on two individual problems, the gradient iterates of the saddle point problem yield a locally optimal saddle point. We will show in the upcoming sections that this claim is **not** true, and outline the need of a more complex analysis of the dynamics of the gradient iterates for the saddle point problem.

5.3 Undesired Stable Points

As explained in the previous section, a convergent sequence of gradient iterations on the minimization problem (Eq. 5.24) yields – almost surely – a solution that locally minimizes the function value. This is a desirable property of the optimizer, because (with random initialization) we are guaranteed to obtain a solution to the posed minimization problem. However, this section will show that the gradient dynamics for the saddle point problem do not share the same property, and therefore, we can not be certain that a convergent series yields a solution to the problem.

It is already known that every locally optimal saddle point is a stable stationary point of the gradient dynamics in Eq. (4.1) [18, 20]. However, the gradient dynamic might introduce additional stable points that are not locally optimal saddle points. This fact is in a clear contrast to GD for minimization where the set of stable stationary points of GD is the same as the set of local minima. In the next example we use an example to illustrate our claim.

Example Consider the function

$$f(x,y) = 2x^2 + y^2 + 4xy + \frac{4}{3}y^3 - \frac{1}{4}y^4 \tag{5.26}$$

with $x, y \in \mathbb{R}$[1]. The saddle point problem defined with this function is given by

$$\min_{x\in\mathbb{R}} \max_{y\in\mathbb{R}} f(x,y) \tag{5.27}$$

The critical points of the function, i.e., points for which $\nabla f(x,y) = 0$, are

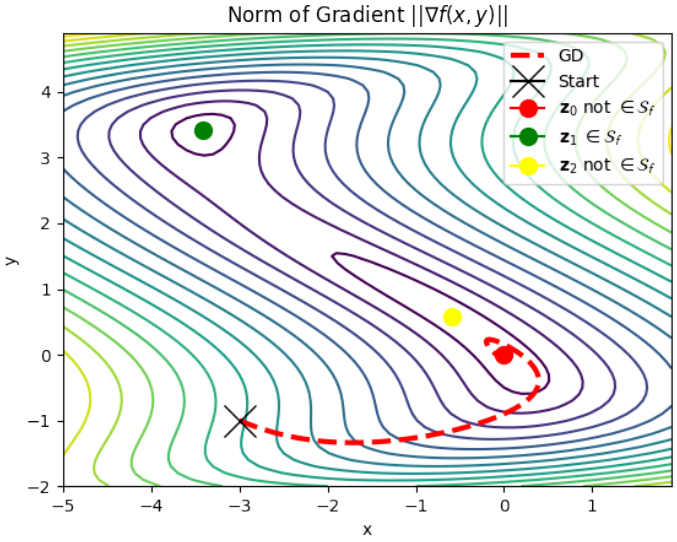
$$\mathbf{z}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad \mathbf{z}_1 = \begin{bmatrix} -2-\sqrt{2} \\ 2+\sqrt{2} \end{bmatrix} \qquad \mathbf{z}_2 = \begin{bmatrix} -2+\sqrt{2} \\ 2-\sqrt{2} \end{bmatrix} \tag{5.28}$$

Evaluating the Hessian at the three critical points gives rise to the following three matrices:

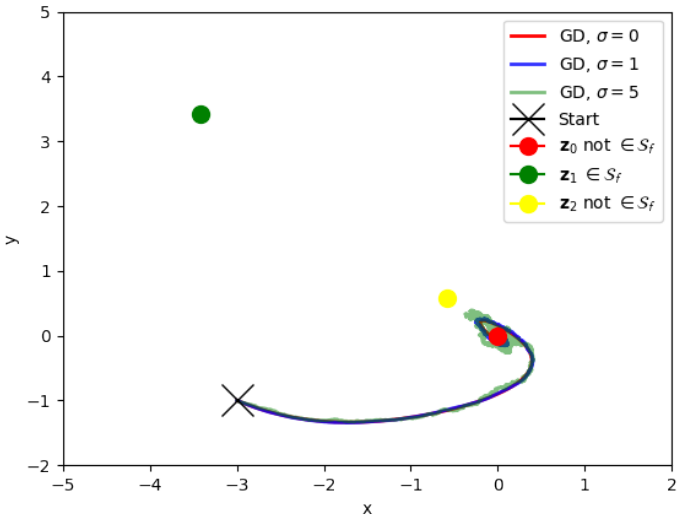
$$\mathbf{H}(\mathbf{z}_0) = \begin{bmatrix} 4 & 4 \\ 4 & 2 \end{bmatrix} \qquad \mathbf{H}(\mathbf{z}_1) = \begin{bmatrix} 4 & 4 \\ 4 & -4\sqrt{2} \end{bmatrix} \qquad \mathbf{H}(\mathbf{z}_2) = \begin{bmatrix} 4 & 4 \\ 4 & 4\sqrt{2} \end{bmatrix}. \tag{5.29}$$

We see that only $\mathbf{z}_1$ is a locally optimal saddle point, because $\nabla_x^2 f(\mathbf{z}_1) = 4 > 0$ and $\nabla_y^2 f(\mathbf{z}_1) = -4\sqrt{2} < 0$ (cf. lemma 3.3), whereas the two other points are both a local minimum in the parameter y, rather than a maximum. Figure 5.1 (a) illustrates gradient steps converging to the undesired stationary point $\mathbf{z}_0$. Due to the stability of this point, even a small perturbation of the gradient in each step can not avoid convergence to it (see Figure 5.1 (b)).

[1] To guarantee smoothness, one can restrict the domain of f to a bounded set.



(a) The background shows a contour plot of the norm of gradient $\|\nabla f(x,y)\|$.



(b) Optimization trajectory when adding gaussian noise from $\mathcal{N}(0,\sigma)$ to the gradient in every step.

Figure 5.1: Optimization trajectory of the gradient method (4.1) on the function in Eq. (5.26) with a step size of $\eta = 0.001$. The method converges to the critical point (0,0), even though it's not a locally optimal saddle point and, therefore, not a solution to the problem defined in (5.27).

Chapter 6

Curvature Exploitation

A recent approach for solving non-convex minimization problems, such as (5.7), involves curvature exploitation [7, 27]. The main idea behind this method is to follow – if possible – the negative curvature of the function. To get an intuition on why this might be a useful direction, consider first an ordinary gradient-based minimization. It ensures convergence to a first-order stationary point, i.e., a point for which the gradient vanishes. However, to find a solution to the minimization problem (5.7), we require second-order stationarity, i.e., the Hessian of the function is positive definite (cf. section 3.1.2 about critical points). By following the negative curvature, on the other hand, it is possible to escape from stationary points that are no local minima. Interestingly, we will show in this chapter how we can use the approach of extreme curvature exploitation not to escape from saddle points, but to find locally optimal saddle points.

6.1 Curvature Exploitation for the Saddle Point Problem

The example in section 5.3 has shown that gradient iterations on the saddle point problem introduce undesired stable points. In this section, we propose a strategy to escape from these stationary points. Our approach is based on extreme curvature exploitation [7].

Extreme curvature direction Let $(\lambda_\theta, \mathbf{v}_\theta)$ be the minimum eigenvalue of $\nabla^2_\theta f(\mathbf{z})$ with its associated eigenvector, and $(\lambda_\varphi, \mathbf{v}_\varphi)$ be the maximum eigen-

value of $\nabla^2_{\boldsymbol{\varphi}} f(\mathbf{z})$ with its associated eigenvector. Then, we define

$$\mathbf{v}_{\mathbf{z}}^{(-)} = \begin{cases} \frac{\lambda_{\boldsymbol{\theta}}}{2\rho_{\boldsymbol{\theta}}} \operatorname{sgn}(\mathbf{v}_{\boldsymbol{\theta}}^\top \nabla_{\boldsymbol{\theta}} f(\mathbf{z})) \mathbf{v}_{\boldsymbol{\theta}} & \text{if } \lambda_{\boldsymbol{\theta}} < 0 \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

$$\mathbf{v}_{\mathbf{z}}^{(+)} = \begin{cases} \frac{\lambda_{\boldsymbol{\varphi}}}{2\rho_{\boldsymbol{\varphi}}} \operatorname{sgn}(\mathbf{v}_{\boldsymbol{\varphi}}^\top \nabla_{\boldsymbol{\varphi}} f(\mathbf{z})) \mathbf{v}_{\boldsymbol{\varphi}} & \text{if } \lambda_{\boldsymbol{\varphi}} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6.2}$$

$$\mathbf{v}_{\mathbf{z}} = (\mathbf{v}_{\mathbf{z}}^{(-)}, \mathbf{v}_{\mathbf{z}}^{(+)}) \tag{6.3}$$

where $\operatorname{sgn} : \mathbb{R} \to \{-1, 1\}$ is the sign function. We call the vector $\mathbf{v}_{\mathbf{z}}$ extreme curvature direction at $\mathbf{z}$.

Algorithm Using the extreme curvature direction, we modify gradient steps to obtain our new update as

$$\begin{bmatrix} \boldsymbol{\theta}_{t+1} \\ \boldsymbol{\varphi}_{t+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\theta}_t \\ \boldsymbol{\varphi}_t \end{bmatrix} + \mathbf{v}_{\mathbf{z}_t} + \eta \begin{bmatrix} -\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \\ \nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \end{bmatrix} \tag{6.4}$$

The new update is constructed by adding the extreme curvature direction to the gradient method of Eq. (4.1). From now on we will refer to this modified update as the **CESP** method (**c**urvature **e**xploitation for the **s**addle **p**oint problem).

6.2 Theoretical Analysis

6.2.1 Stability

Extreme curvature exploitation has already been used for escaping from unstable stationary points (i.e., saddle points) of gradient descent for minimization problems. In saddle point problems, curvature exploitation is advantageous not only for escaping from *unstable* stationary points but also for escaping from undesired *stable* stationary points. In the next lemma, we prove that all stationary points of the CESP method are locally optimal saddle points.

Lemma 6.1 *The point* $\mathbf{z} := (\boldsymbol{\theta}, \boldsymbol{\varphi})$ *is a stationary point of the iterates in Eq.* (6.4) *if and only if* $\mathbf{z}$ *is a locally optimal saddle point.*

Proof The point $\mathbf{z}^* := (\boldsymbol{\theta}, \boldsymbol{\varphi})$ is a stationary point of the iterates if and only if $\mathbf{v}_{\mathbf{z}^*} + \eta(-\nabla_{\boldsymbol{\theta}} f(\mathbf{z}^*), \nabla_{\boldsymbol{\varphi}} f(\mathbf{z}^*)) = \mathbf{0}$. Let's consider w.l.o.g. only the stationary point condition with respect to $\boldsymbol{\theta}$, i.e.

$$\mathbf{v}_{\mathbf{z}^*} = \eta \nabla_{\boldsymbol{\theta}} f(\mathbf{z}^*) \tag{6.5}$$

We prove that the above equation holds only if $\nabla_{\boldsymbol{\theta}} f(\mathbf{z}^*) = \mathbf{v}_{\mathbf{z}^*} = \mathbf{0}$. This can be proven by a simple contradiction: suppose that $\nabla_{\boldsymbol{\theta}} f(\mathbf{z}^*) \neq \mathbf{0}$, then multiplying the both sides of the above equation by $\nabla_{\boldsymbol{\theta}} f(\mathbf{z}^*)$ yields

$$\underbrace{\lambda_{\boldsymbol{\theta}^*} / (2\rho_{\boldsymbol{\theta}})}_{<0} \underbrace{\operatorname{sgn}(\mathbf{v}_{\boldsymbol{\theta}^*}^\top \nabla_{\boldsymbol{\theta}} f(\mathbf{z}^*)) \mathbf{v}_{\boldsymbol{\theta}^*}^\top \nabla_{\boldsymbol{\theta}} f(\mathbf{z}^*)}_{\geq 0} = \eta \|\nabla_{\boldsymbol{\theta}} f(\mathbf{z}^*)\|^2 \tag{6.6}$$

Since the left-hand side is negative and the right-hand side is positive, the above equation leads to a contradiction. Therefore, $\nabla f(\mathbf{z}^*) = \mathbf{0}$ and $\mathbf{v}_{\mathbf{z}^*} = \mathbf{0}$. This means that $\lambda_{\boldsymbol{\theta}^*} \geq 0$ and $\lambda_{\boldsymbol{\varphi}^*} \leq 0$ and therefore, according to lemma 3.3, $\mathbf{z}^*$ is a locally optimal saddle point. □

From lemma 6.1, we know that every stationary point of the CESP dynamics is a locally optimal saddle point. The next lemma establishes a stability condition for the gradient iterates, namely that every locally optimal saddle point is a stable point.

Lemma 6.2 *Suppose that assumptions 3.4 and 3.5 hold. Let* $\mathbf{z}^* := (\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ *be a locally optimal saddle point, i.e.,*

$$\nabla f(\mathbf{z}) = 0,\ \nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}^*) \succeq \mu_{\boldsymbol{\theta}} \mathbf{I},\ \nabla^2_{\boldsymbol{\varphi}} f(\mathbf{z}^*) \preceq -\mu_{\boldsymbol{\varphi}} \mathbf{I},\ (\mu_{\boldsymbol{\theta}}, \mu_{\boldsymbol{\varphi}} > 0) \tag{6.7}$$

Then iterates of Eq. (6.4) *are stable in* $\mathcal{K}^*_\gamma$ *as long as*

$$\gamma \leq \min\{\mu_{\boldsymbol{\theta}} / (\sqrt{2}\rho_{\boldsymbol{\theta}}), \mu_{\boldsymbol{\varphi}} / (\sqrt{2}\rho_{\boldsymbol{\varphi}})\} \tag{6.8}$$

Proof The proof is based on a simple idea: In a $\mathcal{K}^*_\gamma$ neighbourhood of a locally optimal saddle point f can not have extreme curvature, i.e., $\mathbf{v}_{\mathbf{z}} = \mathbf{0}$. Hence, within $\mathcal{K}^*_\gamma$ the update of Eq. (6.4) reduces to the gradient update in Eq. (4.1), which is stable according to [20, 18].
To prove our claim that negative curvature does not exist in $\mathcal{K}^*_\gamma$, we make use of the smoothness assumption. Suppose that $\mathbf{z} := (\boldsymbol{\theta}, \boldsymbol{\varphi}) \in \mathcal{K}^*_\gamma$, then the smoothness assumption 3.4 implies

$$\begin{aligned}
\nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}) &= \nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}_*) - \left(\nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}_*) - \nabla^2_{\boldsymbol{\theta}} f(\mathbf{z})\right) && (6.9)\\
&\succeq \nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}_*) - \rho_{\boldsymbol{\theta}} \|\mathbf{z} - \mathbf{z}_*\| \mathbf{I} && (6.10)\\
&\succeq \nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}_*) - \sqrt{2}\rho_{\boldsymbol{\theta}} \gamma \mathbf{I} && (6.11)\\
&\succeq (\mu_{\boldsymbol{\theta}} - \sqrt{2}\rho_{\boldsymbol{\theta}} \gamma) \mathbf{I} && (6.12)\\
&\succ 0 \qquad [\gamma < \mu_{\boldsymbol{\theta}} / (\sqrt{2}\rho_{\boldsymbol{\theta}})] && (6.13)
\end{aligned}$$

Similarly, one can show that

$$\nabla^2_{\boldsymbol{\varphi}} f(\mathbf{z}) \prec 0 \qquad [\gamma < \mu_{\boldsymbol{\varphi}} / (\sqrt{2}\rho_{\boldsymbol{\varphi}})]. \tag{6.14}$$

Therefore, the extreme curvature direction is zero according to the definition in Eq. (6.1). □

Guarantees for a convergent series Combining the statements of the previous two lemmas, we conclude that the set of locally optimal saddle points and the set of stable stationary points of our new method are the same. As a consequence, we prove that a convergent series of our method is guaranteed to reach a locally optimal saddle point.

Corollary 6.3 *Suppose that assumptions 3.4 and 3.5 hold. Then, a convergent series of the iterates of Eq. 6.4 yields a locally optimal saddle point.*

Proof It is a direct consequence of lemma 6.1 and 6.2. □

6.2.2 Guaranteed Improvement

The gradient update step of Eq. (4.1) has the property that an update with respect to $\boldsymbol{\theta}$ ($\boldsymbol{\varphi}$) decreases (increases) the function value. The next lemma proves that the modified method of Eq. (6.4) shares the same desirable property.

Lemma 6.4 *In each iteration of Eq.* (6.4) f *decreases in* $\boldsymbol{\theta}$ *with*

$$f(\boldsymbol{\theta}_{t+1}, \boldsymbol{\varphi}_t) \leq f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) - (\eta/2)\|\nabla_{\boldsymbol{\theta}} f(\mathbf{z}_t)\|^2 + \lambda_{\boldsymbol{\theta}}^3/(24\rho_{\boldsymbol{\theta}^2}), \tag{6.15}$$

and increases in $\boldsymbol{\varphi}$ *with*

$$f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_{t+1}) \geq f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) + (\eta/2)\|\nabla_{\boldsymbol{\varphi}} f(\mathbf{z}_t)\|^2 + \lambda_{\boldsymbol{\varphi}}^3/(24\rho_{\boldsymbol{\varphi}^2}). \tag{6.16}$$

as long as the step size is chosen as

$$\eta \leq \min\{\frac{1}{2L_{\boldsymbol{\theta}}\ell_{\boldsymbol{\theta}}\rho_{\boldsymbol{\theta}}}, \frac{1}{2L_{\boldsymbol{\varphi}}\ell_{\boldsymbol{\varphi}}\rho_{\boldsymbol{\varphi}}}\} \tag{6.17}$$

Proof The Lipschitzness of the Hessian (Assumption 3.4) implies that for $\Delta \in \mathbb{R}^n$

$$\|f(\boldsymbol{\theta}+\Delta, \boldsymbol{\varphi}) - f(\boldsymbol{\theta}, \boldsymbol{\varphi}) - \Delta^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) - \frac{1}{2}\Delta^\top \nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}, \boldsymbol{\varphi})\Delta\| \leq \frac{\rho_x}{6}\|\Delta\|^3 \tag{6.18}$$

holds. The update in Eq. (6.4) for $\boldsymbol{\theta}$ is given by $\Delta = \alpha\mathbf{v} - \eta\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi})$, where $\alpha = -\lambda/(2\rho_{\boldsymbol{\theta}})$ (where we assume w.l.o.g. that $\mathbf{v}^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) < 0$) and $\mathbf{v}$ is the eigenvector associated with the minimum eigenvalue λ of the Hessian matrix $\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t)$. In the following, we use the shorter notation: $\nabla f := \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ and $\mathbf{H} := \nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}, \boldsymbol{\varphi})$. We can construct a lower bound on the left

hand side of Eq. (6.18) as

$$\|f(\boldsymbol{\theta}+\Delta,\boldsymbol{\varphi}) - f(\boldsymbol{\theta},\boldsymbol{\varphi}) - \Delta^\top \nabla f - \frac{1}{2}\Delta^\top \mathbf{H}\Delta\| \tag{6.19}$$

$$\geq f(\boldsymbol{\theta}+\Delta,\boldsymbol{\varphi}) - f(\boldsymbol{\theta},\boldsymbol{\varphi}) - \Delta^\top \nabla f - \frac{1}{2}\Delta^\top \mathbf{H}\Delta \tag{6.20}$$

$$\geq f(\boldsymbol{\theta}+\Delta,\boldsymbol{\varphi}) - f(\boldsymbol{\theta},\boldsymbol{\varphi}) - \alpha \mathbf{v}^\top \nabla f + (\eta - \frac{\eta^2}{2}L_{\boldsymbol{\theta}})\|\nabla f\|^2 - \frac{1}{2}\alpha^2\lambda + \alpha\eta\lambda \mathbf{v}^\top \nabla f \tag{6.21}$$

which leads to the following inequality

$$f(\boldsymbol{\theta}+\Delta,\boldsymbol{\varphi}) - f(\boldsymbol{\theta},\boldsymbol{\varphi}) \tag{6.22}$$

$$\leq \alpha \mathbf{v}^\top \nabla f - (\eta - \frac{\eta^2}{2}L_{\boldsymbol{\theta}})\|\nabla f\|^2 + \frac{1}{2}\alpha^2\lambda - \alpha\eta\lambda \mathbf{v}^\top \nabla f + \frac{\rho_x}{6}\|\Delta\|^3 \tag{6.23}$$

$$\leq \frac{1}{2}\alpha^2\lambda - (\eta - \frac{\eta^2}{2}L_{\boldsymbol{\theta}})\|\nabla f\|^2 + \frac{\rho_x}{6}\|\Delta\|^3 \tag{6.24}$$

By using the triangular inequality we obtain the following bound on the cubic term

$$\|\Delta\|^3 \leq (\eta\|\nabla f\| + \alpha\|\mathbf{v}\|)^3 \tag{6.25}$$

$$\leq 4\eta^3\|\nabla f\|^3 + 4\alpha^3\|\mathbf{v}\|^3 \tag{6.26}$$

$$= 4\eta^3\|\nabla f\|^3 + 4\alpha^3. \tag{6.27}$$

Replacing this bound into the upper bound of Eq. (6.22) yields

$$f(\boldsymbol{\theta}+\Delta,\boldsymbol{\varphi}) - f(\boldsymbol{\theta},\boldsymbol{\varphi}) \leq \frac{1}{2}\alpha^2\lambda - (\eta - \frac{\eta^2}{2}L_{\boldsymbol{\theta}})\|\nabla f\|^2 + \frac{\rho_x}{6}\left(4\eta^3\|\nabla f\|^3 + 4\alpha^3\right) \tag{6.28}$$

The choice of $\alpha = -\lambda/(2\rho_{\boldsymbol{\theta}})$ leads to further simplification of the above bound:

$$f(\boldsymbol{\theta}+\Delta,\boldsymbol{\varphi}) - f(\boldsymbol{\theta},\boldsymbol{\varphi}) \leq \frac{\lambda^3}{24\rho_{\boldsymbol{\theta}}^2} - \eta(1 - \frac{\eta}{2}L_{\boldsymbol{\theta}} - \frac{2}{3}\rho_{\boldsymbol{\theta}}\eta^2\ell_{\mathbf{x}})\|\nabla f\|^2 \tag{6.29}$$

Now, we choose step size η such that

$$1 - \frac{\eta}{2}L_{\boldsymbol{\theta}} - \frac{2}{3}\rho_{\boldsymbol{\theta}}\eta^2\ell_{\boldsymbol{\theta}} \geq 1/2 \tag{6.30}$$

For $\eta \leq 1/(2L\rho_{\boldsymbol{\theta}}\ell_{\boldsymbol{\theta}})$, the above inequality holds. Therefore, the following decrease in the function value is guaranteed

$$f(\boldsymbol{\theta}+\Delta,\boldsymbol{\varphi}) - f(\boldsymbol{\theta},\boldsymbol{\varphi}) \leq \lambda^3/(24\rho_{\boldsymbol{\theta}^2}) - (\eta/2)\|\nabla f\|^2 \tag{6.31}$$

Similarly, one can derive the lower-bound for the function increase in $\boldsymbol{\varphi}$. □

6.3 Efficient Implementation

6.3.1 Hessian-Vector Product

Storing and computing the Hessian in high dimensions is very costly; therefore, we need to find a way to efficiently extract the extreme curvature direction. The most prominent approach for obtaining the eigenvector, corresponding to the largest absolute eigenvalue, (and the eigenvalue itself) of $\nabla^2_{\boldsymbol{\theta}} f(\mathbf{z})$ is power iterations on $\mathbf{I} - \beta \nabla^2_{\boldsymbol{\theta}} f(\mathbf{z})$ as

$$\mathbf{v}_{t+1} = (\mathbf{I} - \beta \nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}))\mathbf{v}_t \tag{6.32}$$

where $\mathbf{v}_{t+1}$ is normalized after every iteration and $\beta > 0$ is chosen such that $\mathbf{I} - \beta \nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}) \succeq 0$. Since this method only requires implicit Hessian computation through a Hessian-vector product, it can be implemented about as efficiently as gradient evaluations [23]. The results of [13] imply that for the case of $\lambda_{\min}(\nabla^2_{\boldsymbol{\theta}} f(\mathbf{z})) \leq -\gamma$, we need at most $\frac{1}{\gamma} \log(k/\delta^2) L_x$ iterations of the power method to find a vector $\hat{\mathbf{v}}$ such that $\hat{\mathbf{v}}^\top \nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}) \hat{\mathbf{v}} \leq -\frac{\gamma}{2}$ with probability $1 - \delta$ (cf. [14]).

The pseudo code in algorithm 2 shows an efficient realization of the CESP method by using the Hessian-vector product[1].

6.3.2 Implicit Curvature Exploitation

Recent results on the non-convex minimization problem have shown that perturbed gradient steps with an isotropic noise can approximately move along the extreme curvature direction [11]. Suppose that $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0, \boldsymbol{\varphi}) = 0$, $\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0, \boldsymbol{\varphi}) \not\succ 0$. Perturbed gradient steps can be written as

$$\boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 + r\xi \tag{6.33}$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}). \tag{6.34}$$

where ξ is drawn uniformly from the unit sphere. There is a choice for r, the noise term ξ, and the step size η such that $\boldsymbol{\theta}_T$ exploits the negative curvature of $\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0, \boldsymbol{\varphi})$. In other words, one can naturally exploit the negative curvature by perturbation of parameters and gradient steps on $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$. Nonetheless, this approach requires a very careful adjustment of hyperparameters and number of iterations. In our experiments, we only use Hessian-vector products to approximate the extreme curvature direction.

[1] The source code for an implementation of this algorithm in tensorflow is available on GitHub.

Algorithm 2 Efficient CESP implementation

Require: Smooth f, $(\boldsymbol{\theta}, \boldsymbol{\varphi})$, β, $\rho_{\boldsymbol{\theta}}$, $\rho_{\boldsymbol{\varphi}}$

function MinimumEigenvalue(function: f, parameters: $\mathbf{x}$)
  $\mathbf{v}_0 \leftarrow$ random vector $\in \mathbb{R}^{|\mathbf{x}|}$ with unit length.
  for i in 1, ..., k **do**
    $\mathbf{v}_i \leftarrow (\mathbf{I} - \beta \nabla^2_{\mathbf{x}} f)\mathbf{v_{i-1}}$
    $\mathbf{v}_i \leftarrow \mathbf{v}_i / \|\mathbf{v}_i\|$
  end for
  $\mathbf{v} \leftarrow -\mathbf{v}_k \times \operatorname{sgn}(\mathbf{v}_k^\top \nabla_{\mathbf{x}} f)$
  $\lambda \leftarrow \mathbf{v}^\top \nabla^2_{\mathbf{x}} f \mathbf{v}$
  if $\lambda \geq 0$ **then**
    $\lambda \leftarrow 0$
    $\mathbf{v} \leftarrow \mathbf{0}$
  end if
  return $\mathbf{v}, \lambda$
end function
for t in 1, ..., T **do**
  $\mathbf{v}_{\boldsymbol{\theta}}, \lambda_{\boldsymbol{\theta}} \leftarrow$MinimumEigenvalue$(f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t), \boldsymbol{\theta}_t)$
  $\mathbf{v}_{\boldsymbol{\varphi}}, -\lambda_{\boldsymbol{\varphi}} \leftarrow$MinimumEigenvalue$(-f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t), \boldsymbol{\varphi}_t)$
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_t + \frac{\lambda_{\boldsymbol{\theta}}}{2\rho_{\boldsymbol{\theta}}} \mathbf{v}_{\boldsymbol{\theta}} - \eta \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t)$
  $\boldsymbol{\varphi}_t \leftarrow \boldsymbol{\varphi}_t + \frac{\lambda_{\boldsymbol{\varphi}}}{2\rho_{\boldsymbol{\varphi}}} \mathbf{v}_{\boldsymbol{\varphi}} + \eta \nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t)$
end for

6.4 Curvature Exploitation for linear-transformed Gradient Steps

As described in section 4.2, it is common practice to use linear-transformed gradient update steps as an optimization procedure on the saddle point problem. This gives rise to optimization methods like Adagrad that have been shown to accelerate learning. In this section, we propose a modified CESP method that adds the extreme curvature direction to the linear-transformed gradient update of Eq. (4.3), and prove that it keeps the same desirable properties concerning its stable points.

We can adapt our CESP method to a linear-transformed variant as

$$\begin{bmatrix} \boldsymbol{\theta}_{t+1} \\ \boldsymbol{\varphi}_{t+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\theta}_t \\ \boldsymbol{\varphi}_t \end{bmatrix} + \mathbf{v}_{\mathbf{z}_t} + \eta \mathbf{A}_{\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t} \begin{bmatrix} -\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \\ \nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t) \end{bmatrix} \tag{6.35}$$

where we choose the linear transformation matrix $\mathbf{A}_{\boldsymbol{\theta}_t, \boldsymbol{\varphi}_t}$ to be positive definite. Even for this adapted method we can show that it is able to filter out the undesired stable stationary points of the gradient method for the saddle point problem. Moreover, we can show that for a convergent series it holds the same guarantees as its non-transformed counterpart, namely that the set

of its stable stationary points and the set of locally optimal saddle points are the same.

Lemma 6.5 *A point $\mathbf{z}^*$ is a stationary point of the linear-transformed CESP update in Eq.* (6.35) *if and only if $\mathbf{z}^*$ is a locally optimal saddle point.*

Proof As a direct consequence of lemma 6.1 and the positive definiteness property of the linear transformation matrix follows that all stationary points of the linear-transformed CESP update in Eq. (6.35) are locally optimal saddle points. □

Lemma 6.6 *Every locally optimal saddle point is a stable stationary point of the linear-transformed CESP update in Eq.* (6.35).

Proof Let's consider some locally optimal saddle point $\mathbf{z}^* := (\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ in the sense of definition 1.1. From lemma 3.3 follows that

$$\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) \succeq \mu_{\boldsymbol{\theta}} \mathbf{I}, \qquad \nabla^2_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) \preceq -\mu_{\boldsymbol{\varphi}} \mathbf{I}. \tag{6.36}$$

for $\mu_{\boldsymbol{\theta}}, \mu_{\boldsymbol{\varphi}} > 0$. As a direct consequence, the extreme curvature direction is zero, i.e., $\mathbf{v}_{\mathbf{z}} = \mathbf{0}$. Hence, the Jacobian of the update in Eq. (6.35) is given by

$$\mathbf{I} + \eta \mathbf{A}_{\mathbf{z}^*} \begin{bmatrix} -\nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}^*) & -\nabla_{\boldsymbol{\theta},\boldsymbol{\varphi}} f(\mathbf{z}^*) \\ \nabla_{\boldsymbol{\varphi},\boldsymbol{\theta}} f(\mathbf{z}^*) & \nabla^2_{\boldsymbol{\varphi}} f(\mathbf{z}^*) \end{bmatrix}. \tag{6.37}$$

The point $\mathbf{z}^*$ is a stable point of the dynamic if all eigenvalues of its Jacobian lie within the unit-sphere. This condition can be fulfilled, with a sufficiently small step size, if and only if all the real parts of the eigenvalues of $\mathbf{J}(\mathbf{z}^*)$ are negative.
Hence, to prove stability of the update for a locally optimal saddle point $\mathbf{z}^*$, we have to show that the following expression is a Hurwitz matrix:

$$\mathbf{J}(\mathbf{z}^*) = \underbrace{\mathbf{A}_{\mathbf{z}^*}}_{:=\mathbf{A}} \underbrace{\begin{bmatrix} -\nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}^*) & -\nabla_{\boldsymbol{\theta},\boldsymbol{\varphi}} f(\mathbf{z}^*) \\ \nabla_{\boldsymbol{\varphi},\boldsymbol{\theta}} f(\mathbf{z}^*) & \nabla^2_{\boldsymbol{\varphi}} f(\mathbf{z}^*) \end{bmatrix}}_{:=\mathbf{H}} := \mathbf{J} \tag{6.38}$$

Since $\mathbf{A}$ is a positive definite matrix, we can construct its square root $\mathbf{A}^{\frac{1}{2}}$ such that $\mathbf{A} = \mathbf{A}^{\frac{1}{2}}\mathbf{A}^{\frac{1}{2}}$. The matrix product $\mathbf{AH}$ can be re-written as

$$\mathbf{AH} = \mathbf{A}^{\frac{1}{2}}(\mathbf{A}^{\frac{1}{2}}\mathbf{H}\mathbf{A}^{\frac{1}{2}})\mathbf{A}^{-\frac{1}{2}}. \tag{6.39}$$

Since we are multiplying the matrix $\tilde{\mathbf{J}} := \mathbf{A}^{\frac{1}{2}}\mathbf{H}\mathbf{A}^{\frac{1}{2}}$ from the left with the inverse of the matrix from which we are multiplying from the right side, we

can observe that $\mathbf{J}$ has the same eigenvalues as $\tilde{\mathbf{J}}$. The symmetric part of $\tilde{\mathbf{J}}(\mathbf{z}^*)$ is given by

$$\frac{1}{2}\left(\tilde{\mathbf{J}}+\tilde{\mathbf{J}}^\top\right) = \frac{1}{2}(\mathbf{A}^{\frac{1}{2}}\mathbf{H}\mathbf{A}^{\frac{1}{2}} + \mathbf{A}^{\frac{1}{2}}\mathbf{H}^\top\mathbf{A}^{\frac{1}{2}}) = \mathbf{A}^{\frac{1}{2}}(\mathbf{H}+\mathbf{H}^\top)\mathbf{A}^{\frac{1}{2}} \tag{6.40}$$

$$= \mathbf{A}^{\frac{1}{2}}\begin{bmatrix} -\nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}^*) & 0 \\ 0 & \nabla^2_{\boldsymbol{\varphi}} f(\mathbf{z}^*) \end{bmatrix}\mathbf{A}^{\frac{1}{2}} \tag{6.41}$$

From assumption 6.36 follows that the block diagonal matrix $(\mathbf{H}+\mathbf{H}^\top)$ is a symmetric, negative definite matrix for which, therefore, holds $\mathbf{x}^\top(\tilde{\mathbf{J}}+\tilde{\mathbf{J}}^\top)\mathbf{x} \leq 0$ for any $\mathbf{x} \in \mathrm{R}^{n+m}$. The remaining part of the proof follows the argument from [4] Theorem 3.6.

Let $(\lambda, \mathbf{v})$ be an eigenpair of $\tilde{\mathbf{J}}$. Then, the following two equalities hold:

$$\mathbf{v}^*\tilde{\mathbf{J}}\mathbf{v} = \lambda \tag{6.42}$$

$$(\mathbf{v}^*\tilde{\mathbf{J}}\mathbf{v})^* = \mathbf{v}^*\tilde{\mathbf{J}}^\top\mathbf{v} = \bar{\lambda} \tag{6.43}$$

Therefore, we can re-write the real part of the eigenvalue λ as:

$$\mathrm{Re}(\lambda) = \frac{\lambda+\bar{\lambda}}{2} = \frac{1}{2}\mathbf{v}^*(\tilde{\mathbf{J}}+\tilde{\mathbf{J}}^\top)\mathbf{v}. \tag{6.44}$$

By observing that

$$\mathbf{v}^*(\tilde{\mathbf{J}}+\tilde{\mathbf{J}}^\top)\mathbf{v} = \mathrm{Re}(\mathbf{v})^\top(\tilde{\mathbf{J}}+\tilde{\mathbf{J}}^\top)\mathrm{Re}(\mathbf{v}) + \mathrm{Im}(\mathbf{v})^\top(\tilde{\mathbf{J}}+\tilde{\mathbf{J}}^\top)\mathrm{Im}(\mathbf{v}) \tag{6.45}$$

is a real, negative quantity, we can be sure that the real part of any eigenvalue of $\mathbf{J}$ is negative. Therefore it directly follows that, with a sufficiently small step size $\eta > 0$, any locally optimal saddle point $\mathbf{z}^*$ is a stable statoinary point of the linear-transformed update method in Eq. (6.35). □

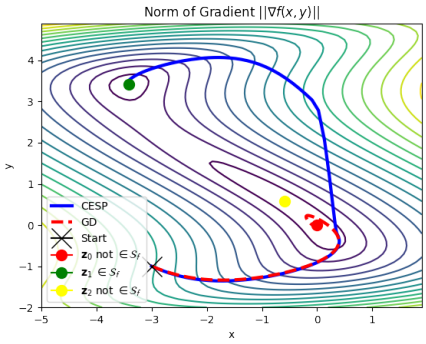
Corollary 6.7 *The set of locally optimal saddle points as defined in 1.1 and the set of stable points of the linear-transformed CESP update method in Eq.* (6.35) *are the same.*

Proof It is a direct consequence of lemma 6.5 and 6.6. □

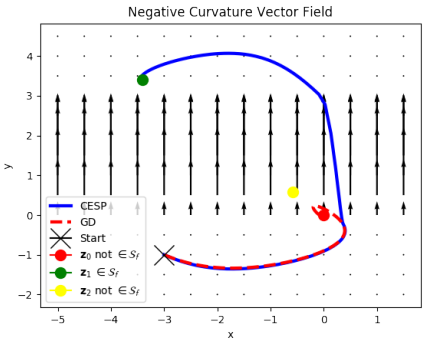
6.5 Experiments

6.5.1 Escaping from Undesired Stationary Points of the Toy-Example

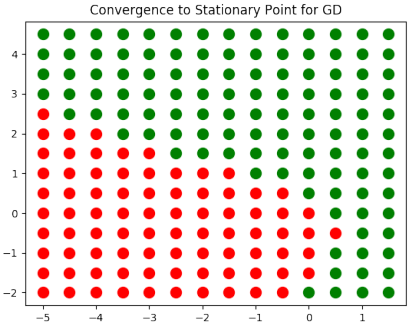
Previously, we saw that for the two-dimensional saddle point problem on the function of Eq. (5.26), gradient iterates might converge to an undesired stationary point that is not locally optimal. Figure 6.1 illustrates that CESP



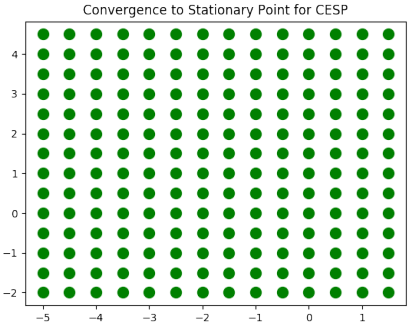
(a)



(b) The background shows the vector field of the extreme curvature, as defined in Eq. (6.1). Note that for the function in Eq. (5.26) the curvature in x-dimension is constant positive and therefore $\mathbf{v}_{\mathbf{z}}^{(-)}$ is always zero.



(c)



(d)

Figure 6.1: Comparison of GD and the new CESP method on the function in Eq. (5.26). While plots (a) and (b) show the two trajectories from the fixed starting point $(-3, -1)$, the plots in (c) and (d) show the basin of attraction of the locally optimal saddle point (green area) and the undesired saddle point (red area).

solves this issues. In this example simultaneous gradient iterates converge to the undesired stationary point $\mathbf{z}_0 = (0, 0)$ for many different initialization parameters, whereas our method always converges to the locally optimal saddle point. The source code of this experiment is available on github.

6.5.2 Generative Adversarial Networks

This experiment evaluates the performance of the CESP method on a Generative Adversarial Network (GAN), which reduces to the saddle point problem

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\varphi}} \left(f(\boldsymbol{\theta}, \boldsymbol{\varphi}) = \mathbb{E}_{\mathbf{x} \sim p_d} \log(D_{\boldsymbol{\theta}}(\mathbf{x})) + \mathbb{E}_{\mathbf{z} \sim p_z} \log(1 - D_{\boldsymbol{\theta}}(G_{\boldsymbol{\varphi}}(\mathbf{z}))) \right) \quad (6.46)$$

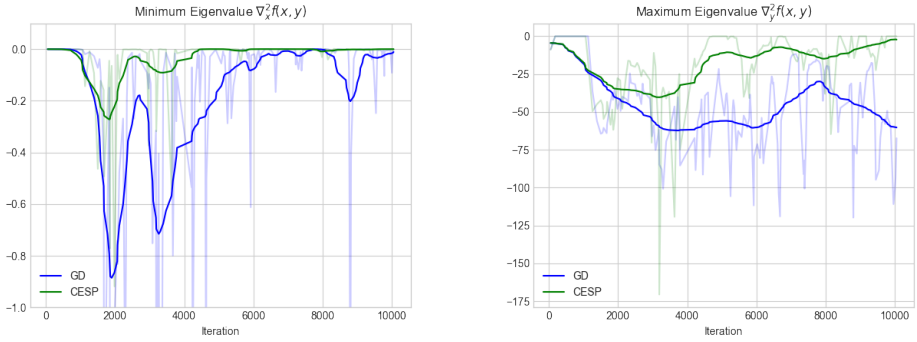
where the functions $D : \mathbb{R}^{|\mathbf{x}|} \to [0, 1]$ and $G : \mathbb{R}^{|\mathbf{z}|} \to \mathbb{R}^{|\mathbf{x}|}$ are neural networks, parameterized with the variables $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$, respectively. We use the MNIST data set and a simple GAN architecture, that is described in table 6.1. To accelerate training we use the linear transformed CESP method, where the transformation matrix corresponds to the Adagrad update.

Table 6.1: Parameters of the GAN model.

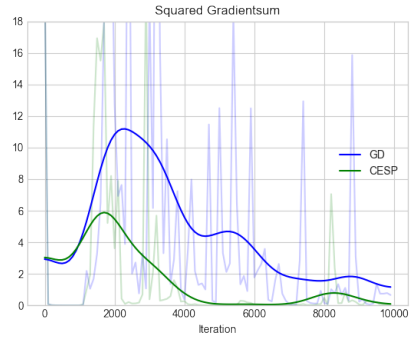
	Discriminator	Generator
Input Dimension	784	10
Hidden Layers	1	1
Hidden Units / Layer	100	100
Activation Function	Leaky ReLU	Leaky ReLU
Output Dimension	1	784
Batch Size	1000	
Learning Rate η	0.05	
Learning Rate $\alpha := \frac{1}{2\rho_{\boldsymbol{\theta}}} = \frac{1}{2\rho_{\boldsymbol{\varphi}}}$	0.01	

On this objective, we evaluate the influence of the negative curvature step of our new method. In particular, we compare the plain Adagrad optimizer (GD) with the newly proposed linear transformed CESP method regarding the spectrum of f at a convergent solution $\mathbf{z}^*$. Note that we are interested in any solution that gives rise to a locally optimal saddle point, as defined in 1.1. In this regard, we track the smallest eigenvalue of $\nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}^*)$ and the largest eigenvalue of $\nabla^2_{\boldsymbol{\varphi}} f(\mathbf{z}^*)$ through the optimization. If the former is positive and the latter negative, the optimization procedure has achieved its goal and found a locally optimal saddle point.

The results are shown in figure 6.2. The decrease in the squared norm of gradient indicates that both methods converge to a solution. Moreover, both fulfill the condition for a locally optimal saddle point for the parameter $\boldsymbol{\varphi}$, i.e., the maximum eigenvalue of $\nabla^2_{\boldsymbol{\varphi}} f(\mathbf{z}^*)$ is negative. However, the graph of the minimum eigenvalue of $\nabla^2_{\boldsymbol{\theta}} f(\mathbf{z}^*)$ shows that the CESP method converges



(a) (b)



(c)

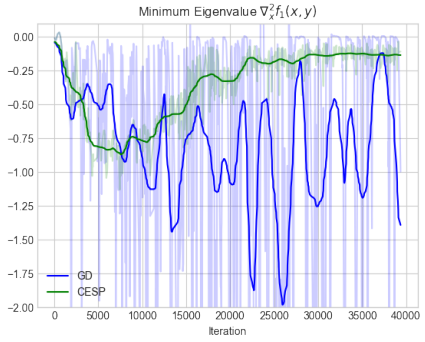
Figure 6.2: The first two plots show the minimum eigenvalue of $\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}, \boldsymbol{\varphi})$ and the maximum eigenvalue of $\nabla_{\boldsymbol{\varphi}}^2 f(\boldsymbol{\theta}, \boldsymbol{\varphi})$, respectively. The third plot shows $\|\nabla f(\mathbf{z}_t)\|$. The transparent graph shows original values, whereas the solid graph is smoothed with a Gaussian filter.

faster, and with less frequent and severe spikes, to a solution where the minimum eigenvalue is zero. Hence, the negative curvature step seems to be able to drive the optimization procedure to regions that yield points closer to a locally optimal saddle point.

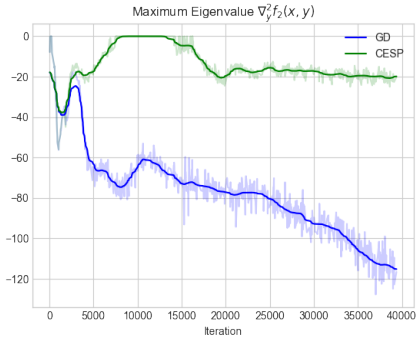
Using two individual loss functions It is common practice in GAN training to not consider the saddle point problem as defined in Eq. (6.46), but rather split the training in two individual optimization problems over different functions (cf. section 2.2.1). In particular, one usually considers

$$\min_{\boldsymbol{\theta}}(f_1(\boldsymbol{\theta}, \boldsymbol{\varphi}) = -\mathbb{E}_{\mathbf{z}\sim p_z} \log D_{\boldsymbol{\theta}}(G_{\boldsymbol{\varphi}}(\mathbf{z}))) \tag{6.47}$$

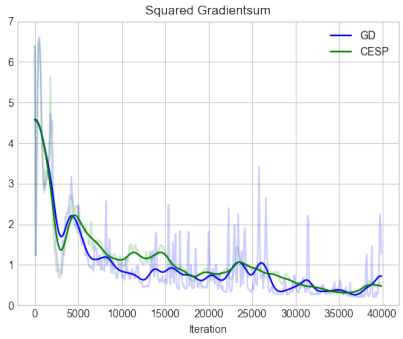
$$\max_{\boldsymbol{\varphi}}(f_2(\boldsymbol{\theta}, \boldsymbol{\varphi}) = \mathbb{E}_{\mathbf{x}\sim p_d} \log D_{\boldsymbol{\theta}}(\mathbf{x}) + \mathbb{E}_{\mathbf{z}\sim p_z} \log(1 - D_{\boldsymbol{\theta}}(G_{\boldsymbol{\varphi}}(\mathbf{z})))) \tag{6.48}$$



(b)



(b)



(c)

Figure 6.3: The first two plots show the minimum eigenvalue of $\nabla^2_{\boldsymbol{\theta}} f_1(\boldsymbol{\theta}, \boldsymbol{\varphi})$ and the maximum eigenvalue of $\nabla^2_{\boldsymbol{\varphi}} f_2(\boldsymbol{\theta}, \boldsymbol{\varphi})$, respectively. The third plot shows $\|\nabla f(\mathbf{z}_t)\|$. The transparent graph shows the original values, whereas the solid graph is smoothed with a Gaussian filter.

Our CESP optimization method is defined individually for the two parameter sets $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ and can therefore also be applied to a setting with two individual objectives. Figure 6.3 shows the results on the GAN problem trained with two individual losses. In this experiment, CESP decreases the negative curvature of $\nabla^2_{\boldsymbol{\theta}} f_1$, while the gradient method can not exploit the negative curvature properly (the negative curvature is oscillating in all iterations). More details on the parameters of the experiments are provided by table 6.1.

6.5.3 Robust Optimization

Setup The approach of Robust Optimization [3], as defined in section 2.3, gives rise to a general saddle point problem. Even though robust optimization is often formulated as a convex-concave saddle point problem, we con-

sider it on a neural network that does not fulfill this assumption. The optimization problem that we target here is an application of robust optimization in empirical risk minimization [21], namely solving

$$\min_{\boldsymbol{\theta}} \sup_{P \in \mathcal{P}} \left[f(\mathbf{X}; \boldsymbol{\theta}, \mathcal{P}) = \left\{ \mathbb{E}_P[l(\mathbf{X}; \boldsymbol{\theta})] : D(P \| \hat{P}_n) \leq \frac{\rho}{n} \right\} \right] \tag{6.49}$$

where $l(\mathbf{X}; \boldsymbol{\theta})$ denotes the cost function to minimize, $\mathbf{X}$ the data, and $D(P\|\hat{P}_n)$ a divergence measure between the true data distribution P and the empirical data distribution $\hat{P}_n$.
We use this framework on the Wisconsin breast cancer data set [16], which is a binary classification task with 30 attributes and 569 samples. Our classification model is chosen to be a Multilayer Perceptron with the following prediction formula:

$$\hat{y}(\mathbf{x}) = \sigma\left(\mathbf{W}_2\, \sigma\left(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1\right) + b_2\right) \tag{6.50}$$

The non-convex sigmoid activation function $\sigma : \mathbb{R} \to (0, 1)$ is applied elementwise. The weights and biases of the layers are the parameter of the model, i.e.,

$$\boldsymbol{\theta} = \left\{ \mathbf{W}_1 \in \mathbb{R}^{2\times 30}, \mathbf{b_1} \in \mathbb{R}^2, \mathbf{W}_2 \in \mathbb{R}^{1\times 2}, b_2 \in \mathbb{R} \right\}. \tag{6.51}$$

The robust loss function is given by

$$l(\mathbf{X}; \boldsymbol{\theta}, P^*) = -\sum_{i=1}^{n} p_i^* \left[y_i \log(\hat{y}(\mathbf{x}_i)) + (1 - y_i) \log(1 - \hat{y}(\mathbf{x}_i)) \right] \tag{6.52}$$

where $\mathbf{x}_i$ and y_i correspond to the i-th datapoint and label, respectively. The values of $P^* = \{p_i^*\}_{i=1}^n$ form a trainable, not-normalized distribution over which we maximize the training loss. To enforce that the distance between the normalized version of P^* and the empirical distribution $P_n = \left\{\frac{1}{n}\right\}_{i=1}^n$ is bounded, we add the following regularization term:

$$r(P^*) = \sum_{i=1}^{n} (p_i^* - \frac{1}{n})^2. \tag{6.53}$$

By subtracting the regularization $r(P^*)$, scaled by $\lambda > 0$, from the robust loss $l(X; \theta, P^*)$, we construct the optimization objective for this experiment as

$$\min_{\boldsymbol{\theta}} \max_{P^*} l(\mathbf{X}; \boldsymbol{\theta}, P^*) - \lambda r(P^*). \tag{6.54}$$

The subtraction comes from the fact that the regularization depends only on P^* and since we are maximizing over this parameter we need to subtract it from the loss.

(a) (b)

Figure 6.4: The left plot shows the minimum eigenvalue of $\nabla^2_{\boldsymbol{\theta}} f(\mathbf{X}; \boldsymbol{\theta}, \mathcal{P})$ and the right plot the squared sum of gradients. The solid line shows the mean value over 10 runs with different initialization, whereas the blurred area indicates the 90th percentile.

Results Figure 6.4 shows the comparison of the gradient method (GD) and our CESP optimizer on this problem in terms of the minimum eigenvalue of $\nabla^2_{\boldsymbol{\theta}} f(\mathbf{X}; \boldsymbol{\theta}, \mathcal{P})$. Note that f is concave with respect to $\mathcal{P}$ and therefore its Hessian is constant negative. The results indicate the anticipated behavior that CESP is able to more reliably drive a convergent series towards a solution where the minimum eigenvalue of $\nabla^2_{\boldsymbol{\theta}} f(\mathbf{X}; \boldsymbol{\theta}, \mathcal{P})$ is positive.

Chapter 7

Second-order Optimization

7.1 Introduction

In the previous chapter, we used extreme curvature exploitation to design an optimizer that provably avoids undesired stable points of gradient-based optimization. Hence, with the information about the most extreme curvature alone, we were already able to significantly improve the theoretical guarantees and practical performance of an optimizer on the saddle point problem. This naturally raises the question if an optimization procedure can be further advanced by considering additional curvature information. Taking this idea to the extreme means having access to the full Hessian, which leads us to the field of second-order optimization. In this chapter, we explore the properties of second-order methods for the saddle point problem, identify issues that arise in this context, and propose a modified Newton method for non convex-concave saddle point optimization.

Benefits of Second-order Optimization Second-order optimization methods, like the Newton method, hold great potential for accelerating training in many learning tasks. Making use of the curvature information of the function, these methods can often find optima in fewer steps than their first-order counterparts. Even though the additional computational complexity, these methods introduce, make them in-practical for most contemporary Deep Learning tasks, the rapid growth of available computational power may pave the way for them – enabling them to become the standard for general optimization tasks. Currently, the family of quasi-Newton methods is bridging the gap between first- and second-order methods. While technically still being an (efficient) first-order method, they approximate the curvature information and try to imitate the update of the second-order method. Hence, with the rise of quasi-Newton methods and the growing computational power of modern systems, the analysis of second-order methods

becomes an essential task.

For solving the saddle point problem in practice, it is the conventional approach to change different optimization schemes like GD, Adagrad or Adam in a modular way, assuming that they are all capable of optimizing the function as intended. However, as already seen in previous chapters, this assumption is not true in general and, therefore, optimizer need to be analyzed carefully on their suitability for the saddle point problem. This chapter shows that Newton's method suffers from two major problems, that can be circumvented by two individual modifications. The first problem is closely related to the attraction issue of Newton's method to saddle points in non-convex minimization [8]. While this can be solved with the *Saddle Free Newton* (SFN) approach from [8], the second problem is caused by the simultaneous optimization procedure itself and, thus, yields a need for a new problem formulation. Instead of splitting the problem into two individual minimization problems, we need to apply the Newton method on the *full* objective to make sure that we obtain the best estimate of the update directions.

7.2 Dynamics of Newton's Method

We can apply Newton's method on the saddle point problem in a straightforward manner by using the simultaneous optimization approach. The particular update equations for this method are given in table 4.1. Stacking the two equations together yields the optimization step in matrix form as:

$$\begin{bmatrix} \boldsymbol{\Delta\theta} \\ \boldsymbol{\Delta\varphi} \end{bmatrix} = \eta \begin{bmatrix} \nabla^2_{\boldsymbol{\theta}} f & 0 \\ 0 & -\nabla^2_{\boldsymbol{\varphi}} f \end{bmatrix}^{-1} \begin{bmatrix} -\nabla_{\boldsymbol{\theta}} f \\ \nabla_{\boldsymbol{\varphi}} f \end{bmatrix}, \tag{7.1}$$

where we used the equivalence between the inverse of a block diagonal matrix with the inverse of all its elements. With the help of the following lemma, we show that the simultaneous Newton update step shares a desirable property with Gradient Descent, namely that all locally optimal saddle points are stable stationary points of its dynamics [18, 20].

Lemma 7.1 *Consider a locally optimal saddle point* $\mathbf{z}^* := (\boldsymbol{\theta}, \boldsymbol{\varphi})$*. This point is a stable stationary point to the general update dynamics of Eq.* (4.3) *if* $\boldsymbol{\mathcal{A}}_t$ *and* $\boldsymbol{\mathcal{B}}_t$ *are positive definite matrices.*

Proof It is a direct consequence from lemma 6.6. □

As mentioned in table 4.1, $\left(\nabla^2_{\boldsymbol{\theta}} f\right)^{-1}$ and $(\nabla^2_{\boldsymbol{\varphi}} - f)^{-1}$ are positive definite matrices around any locally optimal saddle point. Therefore, we can apply the result of lemma 7.1 – that every locally optimal saddle point is a stable stationary point – to the update dynamics in Eq. (7.1).

7.2.1 Avoiding Undesired Saddle Points

Minimizing non-convex functions with Newton's method is known to be problematic due to the prevalence of saddle points in high dimensional error surfaces [8]. In this section, we establish a connection between this issue in non-convex minimization and the problem of converging to non-optimal saddle points in saddle point optimization. We follow the argument of [8] to evaluate the optimization step direction of Newton's method around critical points. We analyze a critical point $(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})$ locally by using a second-order Taylor expansion, together with a re-parameterization coming from Moore's Lemma. Because we are optimizing two functions separately with respect to different sets of parameters, the expansions are also done individually:

$$f(\bar{\boldsymbol{\theta}} + \boldsymbol{\Delta\theta}, \bar{\boldsymbol{\varphi}}) = f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) + \frac{1}{2} \sum_{i=1}^{n} \lambda_i^{\theta} (\Delta v_i^{\theta})^2 \tag{7.2}$$

$$f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}} + \boldsymbol{\Delta\varphi}) = f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) + \frac{1}{2} \sum_{i=1}^{m} \lambda_i^{\varphi} (\Delta v_i^{\varphi})^2 \tag{7.3}$$

where λ_i^{θ} and λ_i^{φ} are the eigenvalues from $\nabla^2_{\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})$ and $\nabla^2_{\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})$, respectively. The values of Δv_i^{θ} and Δv_i^{φ} are the updates of the parameters, corresponding to motion along the eigenvectors of their respective Hessian, i.e.,

$$\Delta v_i^{\theta} = (\mathbf{e}_i^{\theta})^{\top} \boldsymbol{\Delta\theta} \tag{7.4}$$

$$\Delta v_i^{\varphi} = (\mathbf{e}_i^{\varphi})^{\top} \boldsymbol{\Delta\varphi} \tag{7.5}$$

where $\mathbf{e}_i^{\theta}$ and $\mathbf{e}_i^{\varphi}$ are the eigenvectors from $\nabla^2_{\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})$ and $\nabla^2_{\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})$, respectively.

The way we defined the simultaneous Newton method (Table 4.1) makes the update of the parameters $\boldsymbol{\theta}$ independent of the change in $\boldsymbol{\varphi}$, and vice versa. Therefore, both individual updates re-scale the step along their eigenvector with the inverse of the corresponding eigenvalue. Using the Taylor expansions from above, this yields the following update steps along their respective eigenvectors:

$$-\frac{\lambda_i^{\theta}}{\lambda_i^{\theta}} \Delta v_i^{\theta} \quad \text{and} \quad -\frac{\lambda_i^{\varphi}}{\lambda_i^{\varphi}} \Delta v_i^{\varphi} \tag{7.6}$$

We can make two interesting observations from these updates, when comparing it to the corresponding Gradient Descent (GD) update. The GD steps along the eigendirections of the two Hessians are given by

$$-\lambda_i^{\theta} \Delta v_i^{\theta} \quad \text{and} \quad \lambda_i^{\varphi} \Delta v_i^{\varphi}. \tag{7.7}$$

The two differences between the methods are that (i) Newton's method re-scales the step length with the absolute value of the eigenvalue and (ii) the direction of the step changes when $\lambda_i^\theta < 0$ ($\lambda_i^\varphi > 0$). While the re-scaling of the step length avoids slowing down in directions with small absolute eigenvalue, which gives Newton's method an advantage over GD, the change in direction can lead to a problem. Let's consider (without loss of generality) the update of the minimization over one parameter $\theta_i \in \boldsymbol{\theta}$. If the ith eigenvalue λ_i^θ of $\nabla^2_{\boldsymbol{\theta}} f$ is negative, the updates of GD and Newton, along the eigenvector $\mathbf{e}_i^\theta$, point in different directions.

$$\text{GD: } |\lambda_i^\theta| \Delta v_i^\theta \qquad \text{Newton: } -\Delta v_i^\theta \tag{7.8}$$

The authors of [8] argue that for the Newton method, we are moving in a direction of increasing error, towards the critical point. Hence, any critical point of the individual dynamics $\bar{\boldsymbol{\theta}}$ ($\bar{\boldsymbol{\varphi}}$) becomes an attractor, even if it is no local minimum (maximum) of the objective, but rather a saddle point, which means that $\nabla^2_{\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})$ is not positive semi-definite ($\nabla^2_{\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})$ is not negative semi-definite). Summarizing this finding for our problem yields that the simultaneous Newton method is attracted to **all** critical points, and not just locally optimal saddle points.

The authors of [8] propose the *Saddle Free Newton* (SFN) method to circumvent the identified problem of the *vanilla* Newton method for non-convex optimization. The SFN method modifies the inverse Hessian $\mathbf{H}^{-1}$ in the update equation by $|\mathbf{H}|^{-1}$, where $|\mathbf{H}|$ is the matrix obtained by taking the absolute value of each of the eigenvalues of $\mathbf{H}$. Through this change, we can make sure that the update of GD and Newton, along the eigendirection, in equation 7.8 always have the same sign. Applying this concept to our simultaneous Newton method for for the saddle point problem is straightforward and results in the following update equation:

$$\begin{bmatrix} \Delta\boldsymbol{\theta} \\ \Delta\boldsymbol{\varphi} \end{bmatrix} = \eta \left| \begin{bmatrix} \nabla^2_{\boldsymbol{\theta}} f & 0 \\ 0 & -\nabla^2_{\boldsymbol{\varphi}} f \end{bmatrix} \right|^{-1} \begin{bmatrix} -\nabla_{\boldsymbol{\theta}} f \\ \nabla_{\boldsymbol{\varphi}} f \end{bmatrix}, \tag{7.9}$$

which yields the two parameter updates along the corresponding eigenvectors as

$$-\frac{\lambda_i^\theta}{|\lambda_i^\theta|} \Delta v_i^\theta \quad \text{and} \quad \frac{\lambda_i^\varphi}{|\lambda_i^\varphi|} \Delta v_i^\varphi. \tag{7.10}$$

As we can see, this method combines the benefits of the re-scaling with the eigenvalue, but still keeps the same direction as GD.

7.2.2 Simultaneous vs. full Newton Updates

The simultaneous Newton update equation 7.1 presents the major simplification of this method. Namely, that we approximate the Hessian of the objective with a block diagonal matrix, disregarding all curvature information of the *cross-terms*. In this section we are going to show that this gives rise to different update directions and visualize its consequences for a simple convex-concave objective.

With the *full* Newton method we denote the parameter updates according to the following equation:

$$\begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} = -\eta \begin{bmatrix}\nabla^2_{\boldsymbol{\theta}} f & \nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f\\ \nabla^\top_{\boldsymbol{\theta}\boldsymbol{\varphi}} f & \nabla^2_{\boldsymbol{\varphi}} f\end{bmatrix}^{-1} \begin{bmatrix}\nabla_{\boldsymbol{\theta}} f\\ \nabla_{\boldsymbol{\varphi}} f\end{bmatrix} \tag{7.11}$$

We update both sets of parameters at once by using the Hessian of the full problem, which includes the curvature information about the *cross terms*, i.e., $\nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f$ and $\nabla^\top_{\boldsymbol{\varphi}\boldsymbol{\theta}} f$. Note that the update equation for the simultaneous method in Eq. (7.1) is a special instance of this where $\nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f = 0$. Therefore, in general, the two methods yield different update equations, which means that they change the parameters along different directions.

Obviously, with a shrinking relative magnitude of the cross term curvature $\nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f$, the block diagonal approximation gets closer to the true Hessian. Intuitively, it seems clear that with a better approximation the update directions of the simultaneous and the full method become more similar. Hence, the similarity of the update directions of the two methods can be controlled by the relative magnitude of $\nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f$. The following paragraph gives a quantitative analysis of this argument, using a simple convex-concave example.

Convex-Concave Example To study the difference between the simultaneous Newton update in Eq. (7.1) and the full Newton update in Eq. (7.11) on an actual example, we consider the following simple convex-concave function

$$f(\boldsymbol{\theta},\boldsymbol{\varphi}) = \frac{\alpha}{2}\|\boldsymbol{\theta}\|^2 + \boldsymbol{\theta}^\top \mathbf{M}\boldsymbol{\varphi} - \frac{\beta}{2}\|\boldsymbol{\varphi}\|^2 \tag{7.12}$$

with $\boldsymbol{\theta} \in \mathbb{R}^n$, $\boldsymbol{\varphi} \in \mathbb{R}^m$, $\mathbf{M} \in \mathbb{R}^{n\times m}$ and $\alpha, \beta > 0$. Since this function is convex in $\boldsymbol{\theta}$ and concave in $\boldsymbol{\varphi}$, the only critical point $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*) = (\mathbf{0}, \mathbf{0})$ is a locally optimal saddle point. The gradient and Hessian of the function are given by

$$\nabla f(\boldsymbol{\theta},\boldsymbol{\varphi}) = \begin{bmatrix}\alpha\boldsymbol{\theta} + \mathbf{M}\boldsymbol{\varphi}\\ -\beta\boldsymbol{\varphi} + \mathbf{M}^\top\boldsymbol{\theta}\end{bmatrix} \qquad \mathbf{H}(\boldsymbol{\theta},\boldsymbol{\varphi}) = \begin{bmatrix}\alpha\mathbb{I} & \mathbf{M}\\ \mathbf{M}^\top & -\beta\mathbb{I}\end{bmatrix}. \tag{7.13}$$

To compare the simultaneous method to the full method, we evaluate the parameter update step for both at a random point $(\boldsymbol{\theta}, \boldsymbol{\varphi})$.

First, observe that derivatives of f of a higher order than two are zero. Hence, the second order Taylor expansion is exact, i.e.,

$$f\left(\begin{bmatrix}\boldsymbol{\theta}+\Delta\boldsymbol{\theta}\\ \boldsymbol{\varphi}+\Delta\boldsymbol{\varphi}\end{bmatrix}\right) = f\left(\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}\right) + \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix}\begin{bmatrix}\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta},\boldsymbol{\varphi})\\ \nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta},\boldsymbol{\varphi})\end{bmatrix} + \frac{1}{2}\begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix}\mathbf{H}(\boldsymbol{\theta},\boldsymbol{\varphi})\begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix}. \quad (7.14)$$

Taking the derivative of the expansion with respect to $\begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix}$, and setting it to zero yields

$$\begin{bmatrix}\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta},\boldsymbol{\varphi})\\ \nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta},\boldsymbol{\varphi})\end{bmatrix} + \mathbf{H}(\boldsymbol{\theta},\boldsymbol{\varphi})\begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} = 0 \quad (7.15)$$

$$\Rightarrow \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} = -\mathbf{H}^{-1}\begin{bmatrix}\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta},\boldsymbol{\varphi})\\ \nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta},\boldsymbol{\varphi})\end{bmatrix} \quad (7.16)$$

which is exactly the update equation for Newton's method. Since our particular function has zero gradients only at the origin, the parameter update of the full Newton method at any point $(\boldsymbol{\theta}, \boldsymbol{\varphi})$ is

$$\begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} = -\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \quad (7.17)$$

leading straight to the origin.

For the simultaneous update, on the other hand, we have the following two update equations

$$\Delta\boldsymbol{\theta} = -\left(\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta},\boldsymbol{\varphi})\right)^{-1}\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta},\boldsymbol{\varphi}) \quad (7.18)$$

$$= -\frac{1}{\alpha}(\alpha\boldsymbol{\theta} + \mathbf{M}\boldsymbol{\varphi}) = -\boldsymbol{\theta} - \frac{1}{\alpha}\mathbf{M}\boldsymbol{\varphi} \quad (7.19)$$

$$\Delta\boldsymbol{\varphi} = -\left(\nabla^2_{\boldsymbol{\varphi}} f(\boldsymbol{\theta},\boldsymbol{\varphi})\right)^{-1}\nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta},\boldsymbol{\varphi}) \quad (7.20)$$

$$= \frac{1}{\beta}(-\beta\boldsymbol{\varphi} + \mathbf{M}^\top\boldsymbol{\theta}) = -\boldsymbol{\varphi} + \frac{1}{\beta}\mathbf{M}^\top\boldsymbol{\theta} \quad (7.21)$$

which results in the overall update

$$\begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} = -\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix} + \begin{bmatrix}-\frac{1}{\alpha}\mathbf{M}\boldsymbol{\varphi}\\ \frac{1}{\beta}\mathbf{M}^\top\boldsymbol{\theta}\end{bmatrix}. \quad (7.22)$$

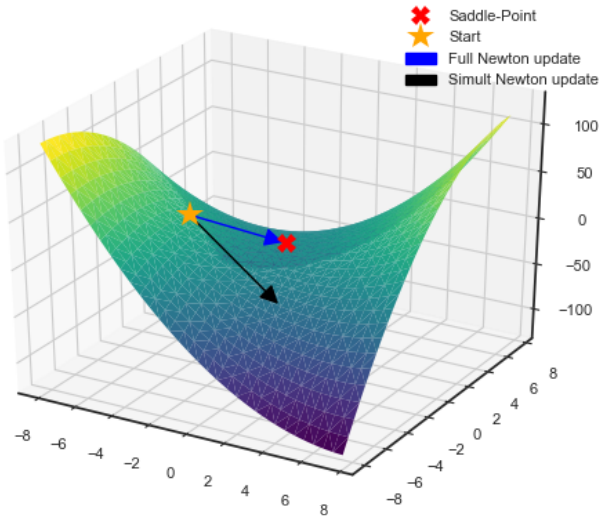


Figure 7.1: Function 7.12 with $\alpha, \beta = 2$ and $\mathbf{M} = 1$. The blue and black arrow show one optimization step of the full Newton and simultaneous Newton, respectively, from the start point $(\theta, \varphi) = (-4, -2)$.

Hence, we introduce an additive error term to the optimization scheme, when disregarding the cross terms, that depends on the matrix $\mathbf{M}$.

In figure 7.1, we see one example where the additive error term leads to a significantly worse update. This example shows that even in a simple convex-concave setting, the simultaneous Newton method, with its block diagonal approximation, can lead to bad updates when the cross-term dependencies in the matrix $\mathbf{M}$ are large.

7.3 Generalized Trust Region Method for the Saddle Point Problem

In the previous section, we have seen that the naïve replacement of a common GD optimizer with the Newton method introduces two severe problems. First, that it is attracted to any critical point and secondly that the simultaneous approach can yield bad approximations to the true eigendirections of the full Hessian. We already identified the solutions to both of the problems, namely, using the SFN method for the former and a full Newton update for the latter. However, combining these two approaches

is not straightforward, as the SFN method is only defined for minimization problems. Therefore, in this section, we extend the *generalized trust region* approach from [8] to the saddle point problem, yielding a modified second-order method to find structured saddle points.

To extend the generalized trust region method, we must account for the fact that we are doing a min-max operation over the function f with respect to the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$, respectively. Therefore, we formulate two individual optimization problems within this framework:

$$\Delta\boldsymbol{\theta} = \arg\min_{\Delta\boldsymbol{\theta}} \mathcal{T}_1 \left(f, \begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} \right) \text{ s.t. } d\left(\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\theta}+\Delta\boldsymbol{\theta}\\ \boldsymbol{\varphi}+\Delta\boldsymbol{\varphi}\end{bmatrix} \right) \leq \Delta \tag{7.23}$$

$$\Delta\boldsymbol{\varphi} = \arg\min_{\Delta\boldsymbol{\varphi}} \mathcal{T}_1 \left(-f, \begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} \right) \text{ s.t. } d\left(\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\theta}+\Delta\boldsymbol{\theta}\\ \boldsymbol{\varphi}+\Delta\boldsymbol{\varphi}\end{bmatrix} \right) \leq \Delta \tag{7.24}$$

The expression $\mathcal{T}_1 \left(f, \begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} \right)$ denotes the first-order Taylor approximation of f around the point $\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}$, evaluated at $\begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix}$. In the generalized trust region framework, the authors of [8] use the discrepancy between the first-order and second-order Taylor approximation as the distance measure d. To extend this idea to our saddle point problem, with two individual optimization objectives, we use the mean sum of the discrepancies to ensure that both problems have the same constraint. Note that the discrepancy is defined as the absolute difference, and therefore it is independent of the sign of f. Thus, the distance simplifies to the discrepancy of f, which is given by:

$$\begin{aligned} &d\left(\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\theta}+\Delta\boldsymbol{\theta}\\ \boldsymbol{\varphi}+\Delta\boldsymbol{\varphi}\end{bmatrix} \right) \\ &\quad = \left| f(\boldsymbol{\theta},\boldsymbol{\varphi}) + \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} \begin{bmatrix}\nabla_{\boldsymbol{\theta}} f\\ \nabla_{\boldsymbol{\varphi}} f\end{bmatrix} + \frac{1}{2} \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} \mathbf{H} \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} \right. \\ &\qquad\qquad \left. - f(\boldsymbol{\theta},\boldsymbol{\varphi}) - \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} \begin{bmatrix}\nabla_{\boldsymbol{\theta}} f\\ \nabla_{\boldsymbol{\varphi}} f\end{bmatrix} \right| \\ &\quad = \frac{1}{2} \left| \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} \mathbf{H} \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} \right| \leq \Delta \end{aligned} \tag{7.25}$$

The Hessian, denoted by $\mathbf{H}$, is defined as

$$\mathbf{H} = \begin{bmatrix}\nabla^2_{\boldsymbol{\theta}} f & \nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f\\ \nabla_{\boldsymbol{\varphi}\boldsymbol{\theta}} f & \nabla^2_{\boldsymbol{\varphi}} f\end{bmatrix}. \tag{7.26}$$

Lemma 7.2 ***(Lemma 1 from [8])*** *Let* $\mathbf{A}$ *be a nonsingular square matrix in* $\mathrm{R}^n \times \mathrm{R}^n$*, and* $\mathbf{x} \in \mathrm{R}^n$ *be some vector. Then it holds that*

$$\left|\mathbf{x}^\top \mathbf{A}\mathbf{x}\right| \leq \mathbf{x}^\top \left|\mathbf{A}\right| \mathbf{x}, \tag{7.27}$$

where $|A|$ *is the matrix obtained by taking the absolute value of each of the eigenvalues of* $\mathbf{A}$*.*

Proof Let $\mathbf{e}_1, \ldots, \mathbf{e}_n$ be the different eigenvectors of $\mathbf{A}$ and $\lambda_1, \ldots, \lambda_n$ the corresponding eigenvalues. By decomposing the matrix $\mathbf{A}$, we can re-write the left-hand side of the inequality as follows:

$$\left|\mathbf{x}^\top \mathbf{A}\mathbf{x}\right| = \left|\mathbf{x}^\top \left(\sum_{i=1}^{n} \lambda_i \mathbf{e}_i \mathbf{e}_i^\top\right) \mathbf{x}\right| = \left|\sum_{i=1}^{n} \lambda_i \mathbf{x}^\top \mathbf{e}_i \mathbf{e}_i^\top \mathbf{x}\right| \tag{7.28}$$

$$= \left|\sum_{i=1}^{n} \lambda_i \left(\mathbf{e}_i^\top \mathbf{x}\right)^\top \left(\mathbf{e}_i^\top \mathbf{x}\right)\right| = \left|\sum_{i=1}^{n} \lambda_i \left(\mathbf{e}_i^\top \mathbf{x}\right)^2\right| \tag{7.29}$$

With the use of the triangular inequality

$$\left|\sum_i \mathbf{x}_i\right| \leq \sum_i |\mathbf{x}_i| \tag{7.30}$$

we obtain the upper bound of the expression as:

$$\left|\mathbf{x}^\top \mathbf{A}\mathbf{x}\right| \leq \sum_{i=1}^{n} \left|\lambda_i \left(\mathbf{e}_i^\top \mathbf{x}\right)^2\right| = \mathbf{x}^\top \left(\sum_{i=1}^{n} |\lambda_i| \, \mathbf{e}_i \mathbf{e}_i^\top\right) \mathbf{x} \tag{7.31}$$

$$= \mathbf{x}^\top \left|\mathbf{A}\right| \mathbf{x} \tag{7.32}$$

□

With the help of Lemma 7.2, we can construct an upper bound of the distance measure d with

$$d\left(\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\theta}+\Delta\boldsymbol{\theta}\\ \boldsymbol{\varphi}+\Delta\boldsymbol{\varphi}\end{bmatrix}\right) \leq \frac{1}{2}\begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} |\mathbf{H}| \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} \leq \Delta \tag{7.33}$$

where $|\mathbf{H}| = \begin{bmatrix}\mathbf{H}_1 & \mathbf{H}_2\\ \mathbf{H}_3 & \mathbf{H}_4\end{bmatrix}$ is the matrix obtained by taking the absolute value of each of the eigenvalues of the Hessian of f.

To solve the two constrained optimization problems, we use Lagrange multipliers which yield the following two functions:

$$\mathcal{L}_1 = f(\boldsymbol{\theta}, \boldsymbol{\varphi}) + \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} \begin{bmatrix}\nabla_\theta f\\ \nabla_\varphi f\end{bmatrix} + \lambda\left(\frac{1}{2}\begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} |\mathbf{H}| \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} - \Delta\right) \tag{7.34}$$

$$\mathcal{L}_2 = -f(\boldsymbol{\theta}, \boldsymbol{\varphi}) - \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} \begin{bmatrix}\nabla_{\boldsymbol{\theta}} f \\ \nabla_{\boldsymbol{\varphi}} f\end{bmatrix} + \lambda \left(\frac{1}{2} \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} |\mathbf{H}| \begin{bmatrix}\Delta\boldsymbol{\theta} \\ \Delta\boldsymbol{\varphi}\end{bmatrix} - \Delta \right) \tag{7.35}$$

Minimizing $\mathcal{L}_1$ with respect to $\Delta\boldsymbol{\theta}$, yields

$$\frac{\partial \mathcal{L}_1}{\partial \Delta\boldsymbol{\theta}} = \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) + \lambda(\mathbf{H}_1 \Delta\boldsymbol{\theta} + \frac{1}{2}\mathbf{H}_2 \Delta\boldsymbol{\varphi} + \frac{1}{2}\mathbf{H}_3^\top \Delta\boldsymbol{\varphi}) \tag{7.36}$$

Similarly, minimizing $\mathcal{L}_2$ with respect to $\Delta\boldsymbol{\varphi}$, yields

$$\frac{\partial \mathcal{L}_1}{\partial \Delta\boldsymbol{\varphi}} = -\nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) + \lambda(\mathbf{H}_4 \Delta\boldsymbol{\varphi} + \frac{1}{2}\mathbf{H}_3 \Delta\boldsymbol{\theta} + \frac{1}{2}\mathbf{H}_2^\top \Delta\boldsymbol{\theta}) \tag{7.37}$$

Lemma 7.3 *Let $\mathbf{A} \in \mathrm{R}^n \times \mathrm{R}^n$ be a symmetric matrix. Then, $|\mathbf{A}|$ is also symmetric, where $|\mathbf{A}|$ is the matrix obtained by taking the absolute value of each of the eigenvalues of $\mathbf{A}$.*

Proof Let $\mathbf{e}_i, \ldots, \mathbf{e}_n$ be the different eigenvectors of $\mathbf{A}$ and $\lambda_1, \ldots, \lambda_n$ the corresponding eigenvalues. Using eigendecomposition for symmetric real matrices, we can re-write the matrix $\mathbf{A}$ as

$$\sum_{i=1}^{n} \lambda_i \mathbf{e}_i \mathbf{e}_i^\top. \tag{7.38}$$

Taking the absolute values of every eigenvalue leads to the following expression for $|\mathbf{A}|$:

$$\sum_{i=1}^{n} |\lambda_i| \, \mathbf{e}_i \mathbf{e}_i^\top. \tag{7.39}$$

Observing that the outer product of each eigenvector forms a symmetric matrix, and that the set of symmetric matrices is closed under the weighted sum, concludes the proof. □

Using the statement of Lemma 7.3, and observing that $\nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f^\top = \nabla_{\boldsymbol{\varphi}\boldsymbol{\theta}} f$ due to the symmetry of the Hessian, leads to the conclusion, that $\mathbf{H}_2^\top = \mathbf{H}_3$. Hence, we can replace $\mathbf{H}_3^\top$ with $\mathbf{H}_2$ in equation 7.36 and $\mathbf{H}_2^\top$ with $\mathbf{H}_3$ in equation 7.37. Setting both derivatives to zero yields the following two equations, respectively.

$$-\nabla_{\boldsymbol{\theta}} f = \lambda \begin{bmatrix}\mathbf{H}_1 & \mathbf{H}_2\end{bmatrix} \begin{bmatrix}\Delta\boldsymbol{\theta} \\ \Delta\boldsymbol{\varphi}\end{bmatrix} \tag{7.40}$$

$$\nabla_{\boldsymbol{\varphi}} f = \lambda \begin{bmatrix}\mathbf{H}_3 & \mathbf{H}_4\end{bmatrix} \begin{bmatrix}\Delta\boldsymbol{\theta} \\ \Delta\boldsymbol{\varphi}\end{bmatrix} \tag{7.41}$$

This set of equations can be re-written as

$$-\begin{bmatrix} \nabla_{\boldsymbol{\theta}} f \\ -\nabla_{\boldsymbol{\varphi}} f \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{H}_1 & \mathbf{H}_2 \\ \mathbf{H}_3 & \mathbf{H}_4 \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{\theta} \\ \Delta\boldsymbol{\varphi} \end{bmatrix}, \tag{7.42}$$

from which we obtain the following solution:

$$\begin{bmatrix} \Delta\boldsymbol{\theta} \\ \Delta\boldsymbol{\varphi} \end{bmatrix} = -\frac{1}{\lambda} |\mathbf{H}|^{-1} \begin{bmatrix} \nabla_{\boldsymbol{\theta}} f \\ -\nabla_{\boldsymbol{\varphi}} f \end{bmatrix}. \tag{7.43}$$

Hence, the optimal update step of the generalized trust region method uses the inverse of the modified Hessian of f, which is constructed using the absolute value of every eigenvalue of the Hessian. Additionally, it is using a negative sign for the update over the parameters to be maximized. We call the method using this update **SPNewton** (**s**addle **p**oint **N**ewton) from now on.

7.3.1 Support for Different Loss Functions

So far, we have adapted the *Saddle Free Newton* procedure via an extended version of the generalized trust method to the saddle point problem. However, we have seen in chapter 2 that in many practical use cases we can accelerate training by considering two individual loss functions, which is not solvable by our proposed SPNewton method. In this section, we are going to generalize the saddle free Newton idea even further to the case where we have two individual loss functions (cf. section 2.2.1) f_1, f_2 over which we minimize and maximize, respectively. I.e., instead of the standard saddle point problem approach where we optimize over

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \equiv \begin{cases} \min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \\ \min_{\boldsymbol{\varphi}} -f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \end{cases}, \tag{7.44}$$

we consider the more general case with two individual loss functions:

$$\begin{cases} \min_{\boldsymbol{\theta}} f_1(\boldsymbol{\theta}, \boldsymbol{\varphi}) \\ \min_{\boldsymbol{\varphi}} -f_2(\boldsymbol{\theta}, \boldsymbol{\varphi}) \end{cases}. \tag{7.45}$$

While adapting common first-order methods to this adjustment is straightforward, our SPNewton method relies on the fact that we optimize over a single loss function for both sets of parameters. Therefore, in this section, we are going to further generalize the method to be able to handle this more generic case.

Adapting the framework The first obvious modification we have to make is to re-formulate the individual optimization problems within the generalized trust region method (7.23, 7.24). This is straightforward as we just have to replace the respective functions, which results in

$$\Delta\boldsymbol{\theta} = \arg\min_{\Delta\boldsymbol{\theta}} \mathcal{T}_1 \left(f_1, \begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix}\right) \quad \text{s.t. } d\left(\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\theta}+\Delta\boldsymbol{\theta}\\ \boldsymbol{\varphi}+\Delta\boldsymbol{\varphi}\end{bmatrix}\right) \leq \Delta \tag{7.46}$$

$$\Delta\boldsymbol{\varphi} = \arg\min_{\Delta\boldsymbol{\varphi}} \mathcal{T}_1 \left(-f_2, \begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix}\right) \quad \text{s.t. } d\left(\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\theta}+\Delta\boldsymbol{\theta}\\ \boldsymbol{\varphi}+\Delta\boldsymbol{\varphi}\end{bmatrix}\right) \leq \Delta. \tag{7.47}$$

However, now that we have two different loss functions we obviously have different Hessian matrices, which means that the two distance constraints are not the same anymore. We defined the distance as the mean absolute discrepancy between the first-order and second-order Taylor approximation, which now becomes the following expression:

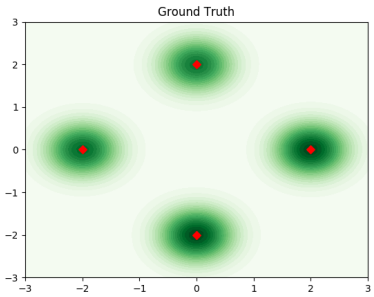
$$\begin{aligned} &d\left(\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{\varphi}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\theta}+\Delta\boldsymbol{\theta}\\ \boldsymbol{\varphi}+\Delta\boldsymbol{\varphi}\end{bmatrix}\right) \\ &\qquad = \frac{1}{2}\left| \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} \mathbf{H}^{(1)} \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix}\right| + \frac{1}{2}\left| \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} \mathbf{H}^{(2)} \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix}\right| \\ &\qquad\qquad \leq \frac{1}{2} \begin{bmatrix}\Delta\boldsymbol{\theta}^\top & \Delta\boldsymbol{\varphi}^\top\end{bmatrix} \left(|\mathbf{H}|^{(1)} + |\mathbf{H}|^{(2)} \right) \begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} \leq \Delta \end{aligned} \tag{7.48}$$

where $\mathbf{H}^{(i)}$ is the Hessian of the function f_i with $i = \{1, 2\}$, and $|\mathbf{H}|^{(i)}$ denotes the corresponding matrix constructed by taking the absolute value of each eigenvalue. The inequality of the distance is a direct consequence of Lemma 7.2.

With the given expression for the distance, we can use the same line of argument as in the previous section to arrive at the following update step equation:

$$\begin{bmatrix}\Delta\boldsymbol{\theta}\\ \Delta\boldsymbol{\varphi}\end{bmatrix} = -\frac{1}{\lambda} \left(|\mathbf{H}|^{(1)} + |\mathbf{H}|^{(2)} \right)^{-1} \begin{bmatrix}\nabla_{\boldsymbol{\theta}} f_1\\ -\nabla_{\boldsymbol{\varphi}} f_2\end{bmatrix} \tag{7.49}$$

This method uses the inverse of the sum of the positive definite Hessians of the individual functions. With the use of this optimizer (which we call AdditiveSPNewton from now on) we are able to use the the *saddle-free* Newton idea in a modified saddle point problem setting with different loss functions (as it usually arises in the training of GANs), at the cost of constructing two Hessian matrices over the full set of parameters.



(a)

	Discr.	Gen.
Input Dimension	2	2
Hidden Layers	1	1
Hidden Units / Layer	30	30
Activation Function	ReLU	ReLU
Output Dimension	1	2
Batch Size	8192	
Learning Rate	0.05	

(b)

Figure 7.2: Plot (a) shows the data distribution. The four means of the Gaussian mixtures are arranged on a circle with radius two around the origin and indicated by the red dots. All the mixtures have equal probability weight and a standard deviation of diag(0.1, 0.1). The table in (b) shows the Network architecture parameters.

7.4 Experiments

7.4.1 SPNewton vs. Newton and Gradient Descent

Objective and setup In the first experiment, we want to compare the performance of SPNewton against the Newton's method (Newton) and Gradient Descent (SGD). We asses the quality of the generative model on a simple 2D toy problem, with data coming from a mixture of four Gaussians. The data distribution is visualized in figure 7.2 (a). The detailed parameters of the model are shown in table 7.2 (b).

Results The visual outcome of the comparison of the three different optimization schemes is summarized in figure 7.3. The kernel density estimate of the generator's distribution is shown in green, while the means of the Gaussian mixtures are represented by the red dots. The results of this experiment show very clearly the aforementioned flaw of Newton's method for this type of problem. Namely, that it converges to some critical point very quickly, that is not an optimal solution. Evidence that it actually converged to a critical

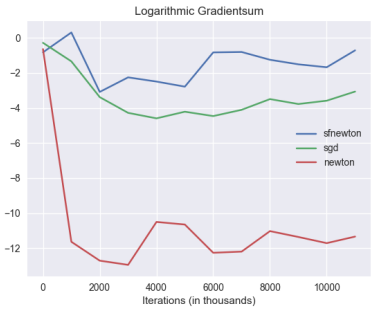


Figure 7.4: Logarithmic sum of gradients.

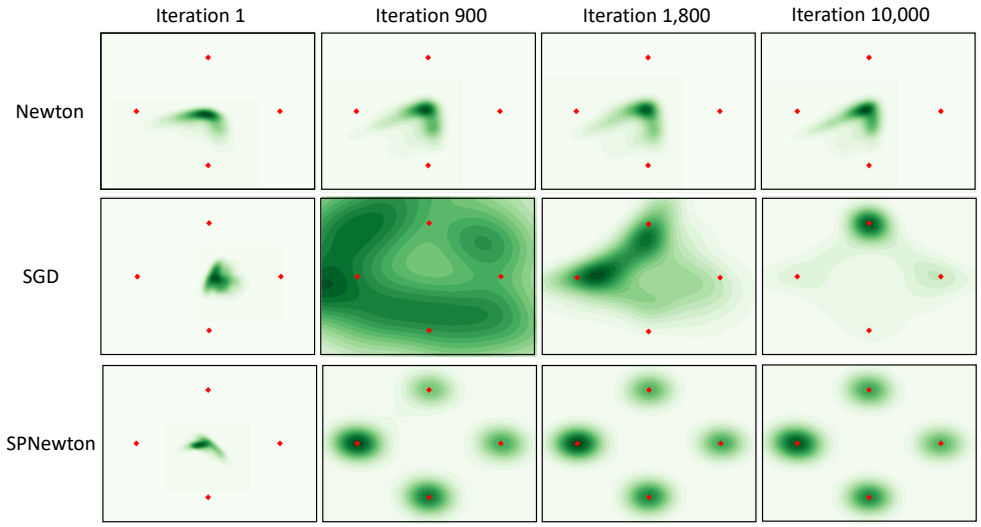


Figure 7.3: Plots of the kernel density estimate of the generator's distribution for 0 to 10000 iterations (left to right) and different optimization methods: Newton, SGD, SPNewton (top to bottom). Note that between the third and the fourth column is a relatively larger gap in iterations.

point is given by the small value of the absolute gradient sum in figure 7.4. The SPNewton method, on the other hand, is not attracted by one of these *undesired* critical points. More to the contrary, it converges quickly to something similar to the true data generating distribution, even outpacing Gradient Descent by far.

7.4.2 SPNewton vs. Simultaneous SFNewton

Objective and setup In the second experiment, we compare the SPNewton method against the simultaneously applied *Saddle Free Newton* method (SFNewton) from Dauphin et al. [8]. This comparison is important, as it gives empirical evidence to the problem of discrepancy in the eigenvectors, described in section 7.2.2. It is, therefore, closely related to the toy experiment on the convex-concave function, but on the significantly larger and more complex 2D-Gaussian GAN problem. We use the same data and model architecture for SPNewton as in the previous experiment.

Results In figure 7.5, we see a scatter plot of the resulting distribution of the generator (blue), compared to a subset of the data samples (green).The top row shows the results for the simultaneous SFNewton, and the bottom row for the SPNewton optimizer. We can see, that the SPNewton optimizer finds the modes of the gaussian a lot faster and converges quickly to an almost perfect distribution. The simultaneous approach, on the other hand,

has trouble identifying the modes and doesn't seem to converge within ten thousand iterations.
The result of this comparison confirms the theoretical argument of the superiority of a Newton method applied to the *full* objective, instead of a simultaneous approach. It illustrates the consequences of approximating the Hessian with a block diagonal matrix.

Iteration 0 Iteration 2500 Iteration 4500 Iteration 8000 Iteration 10000

Figure 7.5: Scatter plots of a subset of the data (green) and generated samples (blue) over the course of 10000 iterations. The upper row shows the results for the SFNewton applied simultaneously, and the bottom row for the proposed SPNewton optimizer.

7.4.3 Robust Optimization

Objective and Setup In this last experiment of the SPNewton method, we take a step back from the GAN objective and move to the framework of Robust Optimization (cf. section 2.3). Robust Optimization, in the sense of empirical risk minimization, gives rise to a saddle point problem that has already been described in more detail in a previous experiment in section 6.5.3. In this experiment we again consider this framework with the same architectural parameters.

Experiment 1 First, we test the SPNewton optimizer against the Newton method in both variations: simultaneously applied (7.1) and on the full objective (7.11). We compare the methods by their test set accuracy over the number of epochs. We run it for 2000 epochs with 10 different random parameter initializations and different trainings-, test-set splits. The accuracy results are reported in the left plot of figure 7.6 with their corresponding 90 percent confidence intervals. To be sure that the methods have converged we plot the squared sum of gradients over epochs in the right plot of figure

The source code for the SPNewton optimizer and the experiments is available on github.

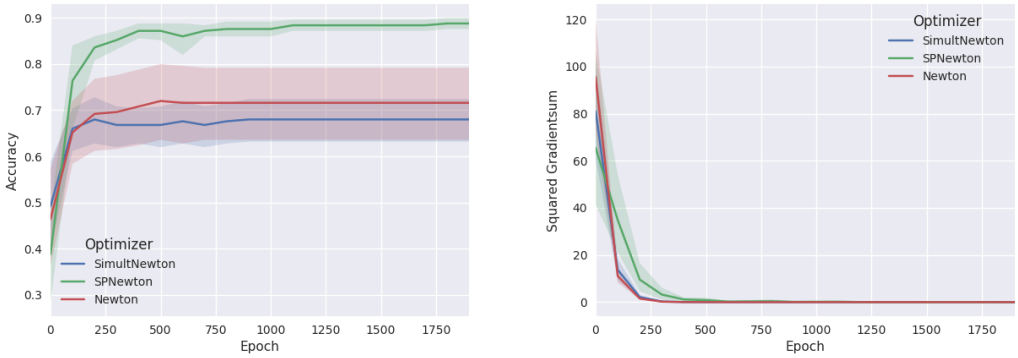


Figure 7.6: Robust optimization experiment on the breast cancer Wisconsin dataset [16] (binary classification on 30 attributes). The left plot shows the test set accuracy and the right plot the corresponding squared sum of gradients for the SPNewton, Newton and the simultaneous Newton optimization. The experiment has been run over 2000 epochs for 10 different initializations and training-, test-set splits. The solid line shows the mean over the different runs and the blurry area the corresponding 90 percent confidence interval.

7.6.
The results clearly support our intuition from the previous section. The original Newton methods, in both forms, converge to a non-ideal critical point where the test set accuracy is low. Our SPNewton optimizer, on the other hand, is able to find a significantly better solution.

Experiment 2 In this experiment, we want to analyze the behavior of the different optimizer around critical points. In the previous section we argued that the major shortcoming of Newton's method is its attraction to *any* saddle point. We constructed SPNewton with the specific goal to not be attracted to *bad* saddle points, but still to locally optimal ones. The following experiment aims to validate this property of SPNewton.
We first run Newton's method until we can be sure, based on the squared sum of gradients, that we found a critical point. Then we perturb the parameters of the model by some gaussian noise coming from $\mathcal{N}(0, 0.1)$, and compare the performance of SPNewton and Newton for the upcoming epochs. An optimizer that is attracted to the initial critical point will quickly converge to the same point again, whereas SPNewton should be able to escape and converge to a different point.
The results of this experiment are shown in figure 7.7. The test set accuracy plot on the left gives evidence that, in fact, the SPNewton optimizer is able to escape from the critical point and find a better solution. On the contrary, the simultaneous Newton method seems to converge back to the initial critical

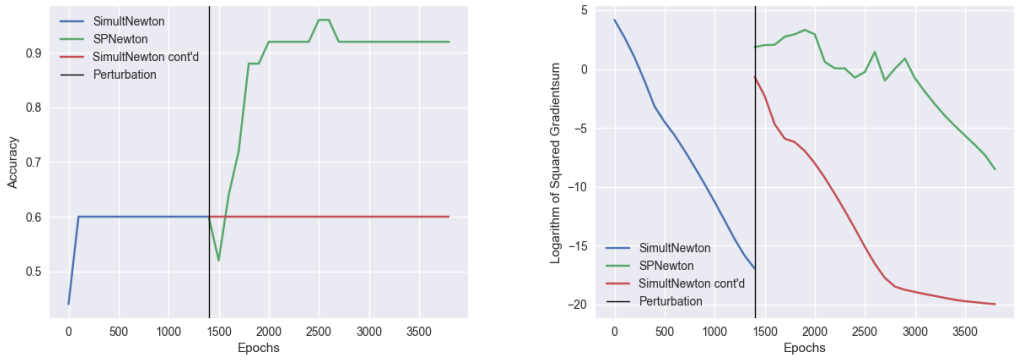


Figure 7.7: Robust optimization experiment on the breast cancer Wisconsin dataset [16] (binary classification on 30 attributes). The left plot shows the test set accuracy and the right plot the corresponding logarithm of the squared sum of gradients for the SPNewton and the simultaneous Newton optimization. The vertical black line indicates the epoch number at which the model parameters have been perturbed with additive values coming from $\mathcal{N}(0, 0.1)$.

point. The squared sum of gradient plot on the right shows that (i) we really found a critical point with the Newton *start*, (ii) the Newton and SPNewton method converge after the perturbation.

7.5 Problems with Second-order Methods and Future Work

The main drawback of any second-order method is the need to compute the Hessian. The size of this matrix depends quadratically on the number of parameters. Hence, for high-dimensional problems it is very costly or even unfeasible to compute and store the Hessian. Most of the saddle point problems we are addressing in this thesis are based on deep neural networks which are naturally very high dimensional. Therefore, the proposed SPNewton optimizer is not applicable due to the computational complexity the Hessian computation entails. However, to overcome this limitation we could relinquish the exact Hessian computation and rely on an approximation, i.e., an SPNewton optimizer within the framework of quasi-Newton methods [22].

Quasi-SPNewton Quasi-Newton methods use the alternating structure of the algorithm to update its approximation of the Hessian $\mathbf{B}$ after each step with the gained gradient information. This can be done by observing that

the change in the gradient $\nabla f(\mathbf{x})$ provides information about the second derivative of f. To justify this statement, we follow the argument of [22] – starting from Taylor's theorem – to derive the *secant* equation. First, observe that for any twice continuously differentiable function $f(\mathbf{x})$ and for some t $\in (0, 1)$ it holds that

$$\nabla f(\mathbf{x}+\mathbf{p}) = \nabla f(\mathbf{x}) + \int_0^1 \nabla^2 f(\mathbf{x}+t\mathbf{p})\mathbf{p}\,\mathrm{d}t. \tag{7.50}$$

By adding and subtracting $\nabla^2 f(\mathbf{x})\mathbf{p}$ on the right hand side we obtain

$$\nabla f(\mathbf{x}+\mathbf{p}) = \nabla f(\mathbf{x}) + \nabla^2 f(\mathbf{x})\mathbf{p} + \int_0^1 (\nabla^2 f(\mathbf{x}+t\mathbf{p}) - \nabla^2 f(\mathbf{x}))\mathbf{p}\,\mathrm{d}t. \tag{7.51}$$

and because $\nabla f(\mathbf{x})$ is continuous, the integral on the right is of size $o(\|\mathbf{p}\|)$. To derive the secant equation we set $\mathbf{x} = \mathbf{x}_k$ and $\mathbf{p} = \mathbf{x}_{k+1} - \mathbf{x}_k$, which leads to

$$\nabla f_{k+1} = \nabla f_k + \nabla^2 f_k(\mathbf{x}_{k+1} - \mathbf{x}_k) + o(\|\mathbf{x}_{k+1} - \mathbf{x}_k\|). \tag{7.52}$$

Hence, for $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ close to a solution where $\nabla^2 f$ is positive definite, we can write

$$\nabla^2 f_k(\mathbf{x}_{k+1} - \mathbf{x}_k) \approx \nabla f_{k+1} - \nabla f_k \tag{7.53}$$

leading to the secant equation:

$$\mathbf{B}_{k+1}(\mathbf{x}_{k+1} - \mathbf{x}_k) = \nabla f_{k+1} - \nabla f_k \tag{7.54}$$

which is the theoretical foundation of all quasi-Newton methods. Most prominently, the BFGS formula ((2.19) of [22]) is used to approximate the Hessian based on the secant equation. On top of BFGS, we can apply the SPNewton method to arrive at *quasi-SPNewton* that doesn't require exact Hessian computation.

L-BFGS By using quasi-Newton methods, we are able to efficiently compute the Hessian even for high-dimensional functions. However, the memory problem of storing an $n \times n$ matrix is still present. To remedy this issue, limited-memory versions of quasi-Newton methods have been developed. The idea behind these methods is to store an implicit approximation of the Hessian. Instead of maintaining the full $n \times n$ matrix, they only keep a few vectors of length n from which the approximation can be retrieved. The limited-memory algorithm that is based on the BFGS updating formula is called L-BFGS. It achieves the memory reduction by only storing the curvature information of the last L iterations in order to construct the Hessian information. Therefore, making a trade-off between approximation quality

and memory requirement by varying the value of L. At the heart of the L-BFGS algorithm is the *two-loop recursion* from Alg. 7.4 [22]. It computes the BFGS update by using only the most recent derivative information. The extension to the L-BFGS algorithm allows to apply the most famous quasi-Newton method to problems with large scale data.
Using L-BFGS to compute an approximate update direction for the SPNewton method is not straightforward. The limited-memory version never explicitly computes an approximate Hessian, but only its product with the gradient. Therefore, it is not obvious how to apply the absolute eigenvalue operator $|\cdot|$ in the update equation of the SPNewton method (Eq. 7.43) to the approximation matrix. Changing L-BFGS to handle the SPNewton update requires careful adjustment of the two-loop recursion algorithm, which is left as future work. Such a modified method would pave the way for the SPNewton optimizer for high-dimensional saddle point problems.

Bibliography

[1] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 214–223, 2017.

[2] Peter L. Bartlett, Olivier Bousquet, and Shahar Mendelson. Local rademacher complexities. *Ann. Statist.*, 33(4):1497–1537, 08 2005.

[3] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Princeton University Press, 2009.

[4] Michele Benzi, Gene H. Golub, and Jörg Liesen. Numerical solution of saddle point problems. *ACTA NUMERICA*, 14:1–137, 2005.

[5] Xi Chen, Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2172–2180. Curran Associates, Inc., 2016.

[6] Ashish Cherukuri, Bahman Gharesifard, and Jorge Cortes. Saddle-point dynamics: conditions for asymptotic stability of saddle points. *SIAM Journal on Control and Optimization*, 55(1):486–511, 2017.

[7] Frank E Curtis and Daniel P Robinson. Exploiting negative curvature in deterministic and stochastic optimization. *arXiv preprint arXiv:1703.00412*, 2017.

[8] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In

Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2933–2941. Curran Associates, Inc., 2014.

[9] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. Technical Report UCB/EECS-2010-24, EECS Department, University of California, Berkeley, Mar 2010.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[11] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. *arXiv preprint arXiv:1703.00887*, 2017.

[12] H.K. Khalil. *Nonlinear Systems*. Pearson Education. Prentice Hall, 2002.

[13] J. Kuczyński and H. Woźniakowski. Estimating the largest eigenvalue by the power and lanczos algorithms with a random start. *SIAM Journal on Matrix Analysis and Applications*, 13(4):1094–1122, 1992.

[14] Jason D Lee, Max Simchowitz, Michael I Jordan, and Benjamin Recht. Gradient descent converges to minimizers. *arXiv preprint arXiv:1602.04915*, 2016.

[15] Kevin Leyton-Brown and Yoav Shoham. *Essentials of Game Theory: A Concise, Multidisciplinary Introduction*. Morgan and Claypool Publishers, 1st edition, 2008.

[16] M. Lichman. UCI machine learning repository, 2013.

[17] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet. Are GANs Created Equal? A Large-Scale Study. *ArXiv e-prints*, November 2017.

[18] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. The numerics of gans. *arXiv preprint arXiv:1705.10461*, 2017.

[19] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. *CoRR*, abs/1611.02163, 2016.

[20] Vaishnavh Nagarajan and J Zico Kolter. Gradient descent gan optimization is locally stable. In *Advances in Neural Information Processing Systems*, pages 5591–5600, 2017.

[21] Hongseok Namkoong and John C Duchi. Variance-based regularization with convex objectives. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 2971–2980. Curran Associates, Inc., 2017.

[22] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization, second edition*. World Scientific, 2006.

[23] Barak A. Pearlmutter. Fast exact multiplication by the hessian. *Neural Computation*, 6:147–160, 1994.

[24] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.

[25] Kevin Roth, Aurélien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. *CoRR*, abs/1705.09367, 2017.

[26] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016.

[27] Y. Xu, R. Jin, and T. Yang. NEON+: Accelerated Gradient Methods for Extracting Negative Curvature for Non-Convex Optimization. *ArXiv e-prints*, December 2017.

Appendix

Convergence Analysis on a Convex-Concave Objective

Lemma 4.1 *Suppose that* $\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \succeq \alpha \mathbf{I}$ *and* $\nabla^2_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}, \boldsymbol{\varphi}) \preceq -\alpha \mathbf{I}$ *with* $\alpha > 0$*, and assumptions 3.4 and 3.5 hold. Let* $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ *be the unique solution of the saddle point problem, then t gradient steps obtain*

$$\| \begin{bmatrix} \boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \leq (1 + \eta(L_{\mathbf{z}}\eta - 2\alpha))^t \| \begin{bmatrix} \boldsymbol{\theta}^{(0)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(0)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \tag{.55}$$

Proof From the update step for simultaneous Gradient Descent/ Ascent in equation 4.1, we can derive the euclidean distance between the parameters after t steps and the optimum as

$$\begin{aligned} \| \begin{bmatrix} \boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 = \| \begin{bmatrix} \boldsymbol{\theta}^{(t-1)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t-1)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \\ + 2\eta \begin{bmatrix} \nabla_{\boldsymbol{\theta}} f & \nabla_{\boldsymbol{\varphi}} f \end{bmatrix} \begin{bmatrix} -\boldsymbol{\theta}^{(t-1)} + \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t-1)} - \boldsymbol{\varphi}^* \end{bmatrix} \\ + \eta^2 \| \begin{bmatrix} \nabla_{\boldsymbol{\theta}} f \\ \nabla_{\boldsymbol{\varphi}} f \end{bmatrix} \|^2 \end{aligned} \tag{.56}$$

Using the mean-value theorem, there exists a value pair $(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})$ in the neighbourhood of $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ such that

$$\begin{aligned} \begin{bmatrix} \nabla_{\boldsymbol{\theta}} f - \underbrace{\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)}_{0} & \nabla_{\boldsymbol{\varphi}} f - \underbrace{\nabla_{\boldsymbol{\varphi}} f(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)}_{0} \end{bmatrix} = \\ \begin{bmatrix} (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top & (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*)^\top \end{bmatrix} \begin{bmatrix} \nabla^2_{\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) & \nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) \\ \nabla_{\boldsymbol{\varphi}\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) & \nabla^2_{\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) \end{bmatrix} \end{aligned} \tag{.57}$$

From the smoothness assumption 3.4 follows that

$$\| \begin{bmatrix} \nabla_{\boldsymbol{\theta}} f \\ \nabla_{\boldsymbol{\varphi}} f \end{bmatrix} \|^2 \leq L_{\mathbf{z}} \| \begin{bmatrix} \boldsymbol{\theta} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \tag{.58}$$

With those two properties we can rewrite equation .55 as

$$\begin{aligned} \| \begin{bmatrix} \boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 &\leq \| \begin{bmatrix} \boldsymbol{\theta}^{(t-1)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t-1)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \\ - 2\eta (\boldsymbol{\theta}^{(t-1)} - \boldsymbol{\theta}^*)^\top \nabla^2_{\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) \, (\boldsymbol{\theta}^{(t-1)} - \boldsymbol{\theta}^*) + 2\eta (\boldsymbol{\varphi}^{(t-1)} - \boldsymbol{\varphi}^*)^\top \nabla^2_{\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) \, (\boldsymbol{\varphi}^{(t-1)} - \boldsymbol{\varphi}^*) \\ + \eta^2 L_{\mathbf{z}} \| \begin{bmatrix} \boldsymbol{\theta}^{(t-1)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t-1)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \end{aligned} \tag{.59}$$

Suppose that

$$\alpha \mathbf{I} \preceq \nabla^2_{\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}), \;\; \nabla^2_{\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) \preceq -\alpha \mathbf{I} \tag{.60}$$

Then

$$\| \begin{bmatrix} \boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \leq \| \begin{bmatrix} \boldsymbol{\theta}^{(t-1)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t-1)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 + \eta (L_{\mathbf{z}} \eta - 2\alpha) \| \begin{bmatrix} \boldsymbol{\theta}^{(t-1)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(t-1)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \tag{.61}$$

$$\leq (1 + \eta (L_{\mathbf{z}} \eta - 2\alpha))^t \| \begin{bmatrix} \boldsymbol{\theta}^{(0)} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^{(0)} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \tag{.62}$$

□

Lemma 4.2 *Suppose that $\nabla^2_{\boldsymbol{\theta}} f \succeq \eta \mathbf{I}$ and $\nabla^2_{\boldsymbol{\varphi}} f \preceq -\gamma \mathbf{I}$ with $\gamma > 0$, and assumptions 3.4 and 3.5 hold. The step size matrices A and B are diagonal, positive semi-definite matrices with*

$$\alpha_{min} \mathbf{I} \preceq A \preceq \alpha_{max} \mathbf{I} \tag{.63}$$

$$\beta_{min} \mathbf{I} \preceq B \preceq \beta_{max} \mathbf{I} \tag{.64}$$

with a constant

$$R := \frac{\min(\alpha_{min}, \beta_{min})}{\max(\alpha_{max}, \beta_{max})}. \tag{.65}$$

Let $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ be the unique solution of the saddle point problem, then one modified gradient step of Eq. (4.3) *with step size*

$$\eta = \frac{\gamma R}{L_{\mathbf{z}} \max(\alpha_{max}, \beta_{max})} \tag{.66}$$

leads to

$$\| \begin{bmatrix} \boldsymbol{\theta}^+ - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^+ - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 = (1 - \frac{\gamma^2}{L_{\mathbf{z}}} R^2) \| \begin{bmatrix} \boldsymbol{\theta} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 - 2\eta \sum_{i=1}^{|\boldsymbol{\theta}|} \sum_{j=1}^{|\boldsymbol{\varphi}|} (A_{ii} - B_{jj})(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)(\boldsymbol{\varphi}_j - \boldsymbol{\varphi}_j^*) \nabla_{\boldsymbol{\theta}_i \boldsymbol{\varphi}_j} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) \quad (.67)$$

Proof We can use the same proof sketch as in Lemma 4.1. But since in general $A \neq B$, the cross-dependencies in equation .55 don't vanish and we end up with the following expression:

$$\begin{aligned} \| \begin{bmatrix} \boldsymbol{\theta}^+ - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^+ - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 &= \| \begin{bmatrix} \boldsymbol{\theta} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \\ &- 2\eta (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}}^2 f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, A (\boldsymbol{\theta} - \boldsymbol{\theta}^*) + 2\eta (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*)^\top \nabla_{\boldsymbol{\varphi}}^2 f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, B (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*) \\ &+ 2\eta (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, B (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*) - 2\eta (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*)^\top \nabla_{\boldsymbol{\varphi}\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, A (\boldsymbol{\theta} - \boldsymbol{\theta}^*) \\ &+ \eta^2 \| \begin{bmatrix} A \nabla_{\boldsymbol{\theta}} f \\ B \nabla_{\boldsymbol{\varphi}} f \end{bmatrix} \|^2 \quad (.68) \end{aligned}$$

Using the assumptions on the block diagonals of the Hessian, i.e.

$$\nabla_{\boldsymbol{\theta}}^2 f \succeq \gamma \mathbf{I} \;\text{ and }\; \nabla_{\boldsymbol{\varphi}}^2 f \preceq -\gamma \mathbf{I}\,,$$

we can form an upper bound as

$$\begin{aligned} \| \begin{bmatrix} \boldsymbol{\theta}^+ - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^+ - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 &\leq \| \begin{bmatrix} \boldsymbol{\theta} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \\ &- 2\eta\gamma (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top A (\boldsymbol{\theta} - \boldsymbol{\theta}^*) - 2\eta\gamma (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*)^\top B (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*) \\ &+ 2\eta (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, B (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*) - 2\eta (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*)^\top \nabla_{\boldsymbol{\varphi}\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, A (\boldsymbol{\theta} - \boldsymbol{\theta}^*) \\ &+ \eta^2 \| \begin{bmatrix} A \nabla_{\boldsymbol{\theta}} f \\ B \nabla_{\boldsymbol{\varphi}} f \end{bmatrix} \|^2 \quad (.69) \end{aligned}$$

Let $c_{min} := \min(\alpha_{min}, \beta_{min}) \geq 0$ be the smallest value of the diagonal matrices A and B, then

$$\begin{aligned} \| \begin{bmatrix} \boldsymbol{\theta}^+ - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^+ - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 &\leq (1 - 2\eta\gamma c_{min}) \| \begin{bmatrix} \boldsymbol{\theta} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \\ &+ 2\eta (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, B (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*) - 2\eta (\boldsymbol{\varphi} - \boldsymbol{\varphi}^*)^\top \nabla_{\boldsymbol{\varphi}\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, A (\boldsymbol{\theta} - \boldsymbol{\theta}^*) \\ &+ \eta^2 \| \begin{bmatrix} A \nabla_{\boldsymbol{\theta}} f \\ B \nabla_{\boldsymbol{\varphi}} f \end{bmatrix} \|^2 \quad (.70) \end{aligned}$$

We can find an upper bound for the last term of the right-hand-side of the inequality by using the smoothness assumption 3.4, together with the

inequality

$$\| \begin{bmatrix} A\nabla_{\boldsymbol{\theta}} f \\ B\nabla_{\boldsymbol{\varphi}} f \end{bmatrix} \|^2 \leq \eta^2 c_{max}^2 \| \begin{bmatrix} \nabla_{\boldsymbol{\theta}} f \\ \nabla_{\boldsymbol{\varphi}} f \end{bmatrix} \|^2 \tag{.71}$$

where $c_{max} := \max(\alpha_{max}, \beta_{max}) \geq 0$. Hence, the upper bound becomes

$$\begin{aligned} \| \begin{bmatrix} \boldsymbol{\theta}^+ - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^+ - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 &\leq (1 - 2\eta\gamma c_{min} + \eta^2 L_{\mathbf{z}} c_{max}^2) \| \begin{bmatrix} \boldsymbol{\theta} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \\ &+ 2\eta(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, B(\boldsymbol{\varphi} - \boldsymbol{\varphi}^*) - 2\eta(\boldsymbol{\varphi} - \boldsymbol{\varphi}^*)^\top \nabla_{\boldsymbol{\varphi}\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, A(\boldsymbol{\theta} - \boldsymbol{\theta}^*) \end{aligned} \tag{.72}$$

Observing that $\nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) = (\nabla_{\boldsymbol{\varphi}\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}))^\top$ we can rewrite the cross dependency term as

$$\begin{aligned} &2\eta(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, B(\boldsymbol{\varphi} - \boldsymbol{\varphi}^*) - 2\eta(\boldsymbol{\varphi} - \boldsymbol{\varphi}^*)^\top \nabla_{\boldsymbol{\varphi}\boldsymbol{\theta}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, A(\boldsymbol{\theta} - \boldsymbol{\theta}^*) \\ &\quad = 2\eta(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top [\nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})\, B - A\, \nabla_{\boldsymbol{\theta}\boldsymbol{\varphi}} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}})](\boldsymbol{\varphi} - \boldsymbol{\varphi}^*) \\ &\quad = -2\eta \sum_{i=1}^{|\boldsymbol{\theta}|} \sum_{j=1}^{|\boldsymbol{\varphi}|} (A_{ii} - B_{jj})(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)(\boldsymbol{\varphi}_j - \boldsymbol{\varphi}_j^*) \nabla_{\boldsymbol{\theta}_i \boldsymbol{\varphi}_j} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) \end{aligned} \tag{.73}$$

Using this identity and setting the step size to

$$\eta := \frac{\gamma}{L_{\mathbf{z}} c_{max}^2}$$

leads to the upper bound

$$\begin{aligned} \| \begin{bmatrix} \boldsymbol{\theta}^+ - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi}^+ - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 &\leq (1 - \frac{\gamma^2}{L} R^2) \| \begin{bmatrix} \boldsymbol{\theta} - \boldsymbol{\theta}^* \\ \boldsymbol{\varphi} - \boldsymbol{\varphi}^* \end{bmatrix} \|^2 \\ &\quad - 2\eta \sum_{i=1}^{|\boldsymbol{\theta}|} \sum_{j=1}^{|\boldsymbol{\varphi}|} (A_{ii} - B_{jj})(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)(\boldsymbol{\varphi}_j - \boldsymbol{\varphi}_j^*) \nabla_{\boldsymbol{\theta}_i \boldsymbol{\varphi}_j} f(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\varphi}}) \end{aligned}$$

which concludes the proof. □

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Non Convex-Concave Saddle Point Optimization

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Adolphs	Leonard

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Zürich, 08.04.2018	

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.