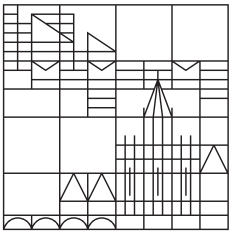
BACHELOR THESIS

Universität Konstanz



STOCHASTIC GRADIENT DESCENT AND ITS APPLICATION FOR PARAMETRIZED BOUNDARY VALUE PROBLEMS UNDER UNCERTAINTIES

by
FLORIAN WOLF

Written and submitted at the
DEPARTMENT OF MATHEMATICS AND STATISTICS
of University Konstanz

Supervised by
PROF. DR. STEFAN VOLKWEIN at University Konstanz

Florian Wolf: Bachelor Thesis
in Mathematics.

Selbstständigkeitserklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit mit dem Thema

STOCHASTIC GRADIENT DESCENT AND ITS APPLICATION FOR PARAMETRIZED BOUNDARY VALUE PROBLEMS UNDER UNCERTAINTIES

selbständig verfasst und keine anderen Hilfsmittel als die angegebenen verwendet habe. Die Stellen, die anderen Werken dem Wortlaut oder dem Sinne nach entnommen sind, habe ich in jedem einzelnen Fall durch Angaben der Quelle, auch der benutzten Sekundärliteratur, als Entlehnung kenntlich gemacht. Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Ort, Datum	Florian Wolf

Zusammenfassung

Diese Arbeit behandelt eine theoretische und praktische Einführung in das stochastische Gradientenverfahren. Im theoretischen Teil zeigen wir zwei wichtige Konvergenzaussagen, welche unter gewissen Bedingung, z. B. einer stark konvexen Zielfunktion, gelten. Zunächst beweisen wir ein allgemeines Konvergenzresultat, welches eine obere Schranke für den erwarteten Fehler unter der Verwendung einer konstanten Schrittweite bereitstellt. In einem zweiten Schritt wird dieses Resultat auf eine Strategie mit kleiner werdender Schrittweite verbessert. Beide Resultate werden auf Kosten der Konvergenzgeschwindigkeit auf nicht-konvexe Zielfunktionen verallgemeinert.

Der praktische Teil behandelt die Anwendung des stochastischen Gradientenverfahrens für parametrisierte Randwertprobleme unter Unsicherheiten. Zuerst wenden wir eine Finite-Differenzen-Diskretisierung auf eine gewöhnliche Differentialgleichung mit Dirichlet-Randwerten an. Durch die Verwendung eines Vorkonditionierers und einer gewichteten Norm verbessern wir die Ergebnisse. Anschließend werden die Resultate auf eine elliptische partielle Differentialgleichung, welche mit einem Finite-Elemente-Verfahren diskretisiert wird, verallgemeinert. In beiden Fällen erweist sich das stochastische Gradientenverfahren als sehr stabiles Optimierungsverfahren, welches Konvergenz im Erwartungswert liefert.

Abstract

In this thesis we want to give a theoretical and practical introduction to stochastic gradient descent (SGD) methods. In the theoretical part, we prove two fundamental convergence results that hold under certain assumptions, like a strongly convex objective function. The first result covers the convergence behaviour of SGD running with a fixed step size sequence and is expanded to the second result, which deals with SGD running with a diminishing step size sequence. For both cases, we provide an upper bound for the expected optimality gap. At the expense of a concrete convergence rate, we then generalize both results to non-convex objective functions.

The practical part of this thesis deals with the application of SGD as a convincing and stable optimizer for parametrized boundary value problems under uncertainties. Firstly, we discretize an ordinary differential equation (ODE) Dirichlet problem using finite differences (FD) and improve the results by using preconditioning techniques and a weighted norm. Secondly, we generalize the results to an elliptic partial differential equation (PDE) Dirichlet problem and aim for a weak solution using a finite element (FE) discretization. For both problems, the SGD algorithm convinces with stable results and provides convergence in expectation.

Contents

List of Figures

List of Tables

Chapter 1.

Introduction and problem formulation

In statistical learning we normally have a set of data $\{(x_i, y_i)\}_{i=1}^n$ with $n \in \mathbb{N}$ training examples, where for $i \in \{1, \dots, n\}$ the value x_i is a feature (e.g. a vector containing pixels of an image) and y_i represents the desired label or output (e.g. if the image contains a cat or not). Our goal is to determine a prediction function $h : \mathcal{X} \to \mathcal{Y}$, such that for $x \in \mathcal{X}$ the value $h(x)$ is a sufficiently accurate prediction of the desired value y. Here, $\mathcal{X}$ represents the input space and $\mathcal{Y}$ the output space. Furthermore, in defining the function h, we want to avoid memorization and instead generalize the concepts learned from the provided examples or the training data. Thus, it turns out that the choice of the function h is key to our learning process.

In general, we want to choose the function h as a minimizer of a risk measure over a selected family of prediction functions $\mathcal{H}$. Assume that we have a joint probability distribution $\mathbb{P}(x, y)$ that on the one hand represents the distribution $\mathbb{P}(x)$ of the inputs, as well as on the other hand the conditional probability $\mathbb{P}(y|x)$ of a label y given the feature x. Our (up to this point still very general) goal is to minimize the *expected risk* of misclassification *over all possible inputs*. More formally,

$$\min_{h \in \mathcal{H}} R(h), \text{ with } R(h) := \mathbb{P}(h(x) \neq y) = \mathbb{E}[\chi_{\{h(x) \neq y\}}] \tag{1.1}$$

where χ is the indicator function and $h \in \mathcal{H}$ belongs to a suitable (and later specified) function space. The last equality follows from the definition of the expected value for indicator functions.

Our next goal is to reduce the optimization problem from an infinite dimensional problem (optimizing over a space of functions), to a very high-, but finite-dimensional

optimization problem. Therefore, we optimize over a fixed function h, which is parametrized by a real vector $w \in \mathbb{R}^d$, the so-called weights, instead of optimizing over a closed set of functions or even a function space. Hence, we can write $\mathcal{H}$ as

$$\mathcal{H} = \{h(\cdot; w) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_y} | w \in \mathbb{R}^d\},$$

where the parametrized functions $h(\cdot;\cdot) : \mathbb{R}^{d_x} \times \mathbb{R}^d \to \mathbb{R}^{d_y}$ will be specified later. In our notation d_x is the dimension of the input space and d_y the dimension of the output space.

To measure how good a current prediction function $h(\cdot; w)$ for a concrete $w \in \mathbb{R}^d$ is, we assume having a *loss function* $\ell : \mathbb{R}^{d_y} \times \mathbb{R}^{d_y} \to \mathbb{R}$. Given a pair (x, y), the function ℓ yields the loss $\ell(h(x; w), y)$, where $h(x; w)$ is our prediction output and y is the desired output. In reality, we usually define this function by ourselves.

Now we will consider two cases: the theoretical case and the case given by the applicants.

Ideally Looking at the problem (1.1) from a theoretical perspective, we want to find a minimizer $w_* \in \mathbb{R}^d$ considering *every possible* in- and output case, which leads to a "perfect result". Therefore, we need the, at the start already addressed, probability distribution $\mathbb{P}$. This probability measure represents the *true relationship* between the in- and output

$$\mathbb{P} : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \to [0, 1].$$

Then, we can start minimizing the objective function $R : \mathbb{R}^d \to \mathbb{R}$, given by

$$w \mapsto R(w) := \int_{\mathbb{R}^{d_x} \times \mathbb{R}^{d_y}} \ell(h(x; w), y) \, \mathrm{d}\mathbb{P}(x, y) = \mathbb{E}\left[\ell(h(\cdot; w), \cdot)\right]. \tag{1.2}$$

For a given parameter $w \in \mathbb{R}^d$, we call (1.2) the **expected risk** with respect to the distribution $\mathbb{P}$. In reality we do not have access to the real relationship stated above. This leads to a more practical approach.

Reality In reality, there is no complete information about $\mathbb{P}$ given. Instead we have to

work with the available data.[1] Hence, we try to estimate the expected risk using our given (independently drawn) input-output samples $\{(x_i, y_i)\}_{i=1}^n \subset \mathbb{R}^{d_x} \times \mathbb{R}^{d_y}$, to get the **empirical risk**

$$R_n(w) := \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i; w), y_i), \tag{1.3}$$

which is a way more practical optimization problem.

1.1. Motivation for a stochastic gradient approach

In this section, we want to take a closer look at a concrete optimization problem of the form (1.3), to see which problems arise using standard optimization algorithms.

Let us consider a data set $\{(x_i, y_i)\}_{i=1}^n \subset \mathbb{R}^d \times \{0, 1\}$. We have d-dimensional vectors (e.g. hours a student studied for an exam) as input values and outputs taking boolean values (e.g. the student passed the exam represented by the value 1 and failing corresponds to the value 0). We now assume this given data correlates to a logistical curve. To obtain a classifier, we want to solve the following minimization problem

$$\min_{w \in \mathbb{R}^d} f(w) = \frac{1}{n} \underbrace{\sum_{i=1}^{n} \left(-y_i x_i^\mathsf{T} w + \log\left(1 + \exp(x_i^\mathsf{T} w)\right)\right)}_{=: f_i(w)}, \tag{1.4}$$

for $w \in \mathbb{R}^d$. A (full) gradient approach with an initial guess $w_0 \in \mathbb{R}^d$ is for $k \in \mathbb{N}_0$ of the form

$$w^{(k+1)} \leftarrow w^{(k)} - \alpha_k \cdot \nabla f(w^{(k)}), \tag{1.5}$$

with a sequence of stepwidths $\{\alpha_k\}_{k \in \mathbb{N}_0}$ generated by a line search strategy like *Armijo* or *Powell-Wolfe*. Due to the fact that the gradient is a linear operator, we can write

$$\nabla f(w) = \nabla \left(\frac{1}{n} \sum_{i=1}^{n} f_i(w) \right) = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w). \tag{1.6}$$

[1] Another import reason to not consider the optimization problem over an integral equation, is the amount of computational effort needed to approximate the integral. Recalling the lecture *Numerische Mathematik*, we assume approximating the integral using the trapezoidal rule and *only* 10 grid points. In total, this leads to 10^D grid points, whereby $D = d_x + d_y$. This is way too expensive to be practically realizable.

At this point the problem becomes evident: the calculation of the full gradient is at the order of $\mathcal{O}(nd)$. This is doable when n and d are moderate, but *not when n is huge*. In this case huge means, that n is, of the order of magnitude, up to 100 million or even larger. This is common for large-scale machine learning projects executed by companies like Google or Amazon.

Now we can see that in large-scale machine learning, a full gradient approach is too expensive or not even doable. Considering other approaches like *Newton* or *Quasi-Newton* methods, we even have more computational costs due to the calculation of the Hessian matrix (or its approximation). Consequently, this is not doable either.

This leads to the idea of a **stochastic gradient descent (SGD)** approach. In each iteration $k \in \mathbb{N}$ we *randomly* (without replacement) choose an index $i_k \in \{1, \dots, n\}$ to get the new iteration process via

$$w^{(k+1)} \leftarrow w^{(k)} - \alpha_k \cdot \nabla f_{i_k}(w^{(k)}). \tag{1.7}$$

In particular, note that not each direction $-\nabla f_{i_k}(w^{(k)})$ has to be a descent direction, but in *expectation*, the sequence $\{w_k\}_{k\in\mathbb{N}}$ can be guided towards a local minimizer. To achieve this goal, choosing the step size sequence $\{\alpha_k\}_{k\in\mathbb{N}}$ is crucial.

We can clearly see that $\nabla f_{i_k}(w)$ is an unbiased estimate of the full gradient $\nabla f(w)$. Using the formula for the expected value with respect to the uniform distribution,[2] we obtain

$$\mathbb{E}[\nabla f_{i_k}(w)] = \frac{1}{n}\sum_{j=1}^{n} \nabla f_j(w) = \nabla f(w). \tag{1.8}$$

The advantages of this stochastic method can be recognized immediately. In particular, the calculation of one iteration is at the order of $\mathcal{O}(d)$ instead of $\mathcal{O}(nd)$, which is for really large values of n significantly cheaper. However, the method is not deterministic, not easy parallelizable and, as we will see later, leads only to convergence in expectation.

Having the idea of SGD in mind and by reconsidering the gradient descent method, one can combine the advantages both methods in a so-called **(mini) batch approach**.

[2]We want to clarify the calculation. Assume having a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, with a finite set Ω. Let $X : \Omega \to \{x_i\}_{i=1}^n$ be a uniformly distributed random variable, cf. [Kle13, p. 13, 33], in particular $\mathbb{P}(X = x_i) = \frac{1}{n}$, for all $i = 1, \dots, n$. Then by definition $\mathbb{E}[X] = \sum_{i=1}^n x_i \cdot \mathbb{P}(X = x_i) = \sum_{i=1}^n x_i \frac{1}{n} = \frac{1}{n}\sum_{i=1}^n x_i$. In our case, we have $\Omega = \{1, \dots, n\}$ and $X : \Omega \to \{\nabla f_1(w), \dots, \nabla f_n(w)\}$, with $\mathbb{P}(X = \nabla f_i(w)) = \frac{1}{n}$ for all $i = 1, \dots, n$.

This method corresponds to the iteration process

$$w^{(k+1)} \leftarrow w^{(k)} - \alpha_k \cdot \frac{1}{|\mathcal{S}_k|} \sum_{i \in \mathcal{S}_k} \nabla f_i(w^{(k)}),$$

where $\mathcal{S}_k \subseteq \{1, \ldots, n\}$ is randomly chosen in each iteration. In this process, one can either keep $|\mathcal{S}_k|$ fixed, for all iterations $k \in \mathbb{N}$, or can even vary this quantity. Using a batch approach, we can use the advantages of SGD due to randomization and additionally expect a better step computation because more samples are used. The batch approach offers a lot of different possibilities where we can decide on the trade of between per-iteration costs and per-iteration improvement.

1.2. Notation

Before we take a closer look at SGD and its analysis, we need some notation to make our algorithms and analysis more generic.

First of all, we want to represent a sample or a set of samples by a random seed ξ. This means that ξ represents either a single sample $(x, y) \in \mathbb{R}^{d_x} \times \mathbb{R}^{d_y}$ or a set of samples $\{(x_i, y_i)\}_{i=1}^n \subset \mathbb{R}^{d_x} \times \mathbb{R}^{d_y}$, where we receive the sample (x_i, y_i) for $i \in \{1, \ldots, n\}$ via $\xi_{[i]}$.

Now, for even more simplicity, we want to define the function f as the composition of the loss function ℓ and the prediction function h, leading to $f := \ell \circ h$. Using this notation, we can redefine the expected and empirical risk, obtaining the following formulas

$$\textbf{(expected risk)} \quad R(w) := \mathbb{E}\left[f(w; \xi)\right]. \tag{1.9}$$

Furthermore, using $f_i(w) := f(w; \xi_{[i]})$ for $i = 1, \ldots n$, we get the definition

$$\textbf{(empirical risk)} \quad R_n(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w). \tag{1.10}$$

In the following study we will denote the k-th element of a sequence of random variables by ξ_k.

Chapter 2.

Stochastic gradient descent

In this section we will take a closer look at the stochastic gradient method for strongly convex objective functions. We will analyse its behaviour regarding convergence and its convergence rate. Our goal is to determine convergence properties and give inequalities for worst case complexity bounds. The structure of this chapter is based on [BCN18, Sec. 4]. Two main results including some additional information will be shown below. Additional proofs can be found in [BCN18, Sec. 4].

2.1. Generic stochastic gradient descent algorithm

To make the analysis of the algorithm more general, we will define a generic objective function.

Definition 2.1.1 (Generic objective function)**.** For analysing both, expected and empirical risk, let us define the **generic objective function** F as

$$F : \mathbb{R}^d \to \mathbb{R}, w \mapsto \begin{cases} R(w) & = \mathbb{E}\left[f(w;\xi)\right] \\ \text{or} & \\ R_n(w) & = \frac{1}{n}\sum_{i=1}^{n} f_i(w) \end{cases} . \tag{2.1}$$

No, we will specify a generic version of the stochastic gradient descent algorithm and take a closer look on the components of the algorithm in Remark 2.1.2.
Except for the stochastic part included in the gradient calculation, this generic algorithm is close to the original generic algorithm for general descent methods, which we analysed in the lecture *Optimization 1*.

Remark 2.1.2 (Interpretation of Algorithm 1)**.** By analysing Algorithm 1, we can remark a few important characteristics.

Algorithm 1 Stochastic gradient descent method (SGD method)

Require: initial guess $w_1 \in \mathbb{R}^d$
1: **for** $k = 1, 2, \ldots$ **do**
2: generate realization of the random variable ξ_k
3: compute the stochastic vector $g(w_k; \xi_k) \in \mathbb{R}^d$
4: choose a step size α_k
5: $w_{k+1} \leftarrow w_k - \alpha_k \cdot g(w_k; \xi_k)$
6: **end for**

(i) The generation of the realization of the random variable ξ_k directly corresponds to the risk measure we use. If we uniformly choose ξ_k over the finite training set, this corresponds to the empirical risk R_n. Alternatively selecting samples according to the distribution $\mathbb{P}$, we obtain the expected risk R. Furthermore, we assume that the random variable sequence $\{\xi_k\}_{k\in\mathbb{N}}$, generated by the algorithm, is independent.

(ii) The computation of the stochastic vector in line 3 is also very generic. This computation could represent SGD or a (mixed) batch approach

$$g(w_k, \xi_k) = \begin{cases} \nabla f(w_k; \xi_k) \\ \frac{1}{n_k} \sum_{i=1}^{n_k} \nabla f(w_k; \xi_{k,i}) \\ H_k \cdot \frac{1}{n_k} \sum_{i=1}^{n_k} \nabla f(w_k; \xi_{k,i}) \end{cases}, \tag{2.2}$$

with $H_k \in \mathbb{R}^{d\times d}$ being a symmetric positive definite matrix.

(iii) The choice of α_k in line 4 allows us to use a fixed step size as well as a sequence of diminishing step sizes. In the following analysis, we will take a closer look on both options for the step size and its effect on convergence.

2.2. Two fundamental lemmas

The following two fundamental lemmas are build on the smoothness of the objective function. Hence, we need some assumptions regarding the first and second moments.

Assumption 2.2.1 (Lipschitz continuity of the objective functions gradient)**.** *In the following report, we assume that the objective function $F : \mathbb{R}^d \to \mathbb{R}$ is continuously differentiable and the gradient function $\nabla F : \mathbb{R}^d \to \mathbb{R}^d$ is Lipschitz continuous with*

Lipschitz constant $L > 0$. In particular, for all $w, \bar{w} \in \mathbb{R}^d$ it holds

$$\|\nabla F(w) - \nabla F(\bar{w})\|_2 \leq L \cdot \|w - \bar{w}\|_2 .$$

Recalling the convergence proof of the gradient descent method in *Optimization 1*, we can see that Assumption 2.2.1 is typical for a gradient based descent method. Otherwise, the gradient would not be a good indicator for how far to move to achieve a decrease in F. We obtain the following proposition.

Proposition 2.2.2. *In the setting of Assumption 2.2.1, we obtain for all $w, \bar{w} \in \mathbb{R}^d$ the inequality*

$$F(w) \leq F(\bar{w}) + \nabla F(\bar{w})^\mathsf{T}(w - \bar{w}) + \frac{L}{2}\|w - \bar{w}\|_2^2 . \tag{2.3}$$

Proof. See [BCN18, Appendix B]. □

Before stating the first of two fundamental lemmas, we need some more notations. We use $\mathbb{E}_{\xi_k}[\cdot]$ to denote the expected value with respect to the distribution of the random variable ξ_k given by w_k. As a consequence, the expression $\mathbb{E}_{\xi_k}[F(w_{k+1})]$ makes sense because w_{k+1} depends on ξ_k (cf. Algorithm 1).

Lemma 2.2.3. *Under Assumption 2.2.1, the iterates generated by Algorithm 1 satisfy for all $k \in \mathbb{N}$ the inequality*

$$\mathbb{E}_{\xi_k}[F(w_{k+1})] - F(w_k) \leq -\alpha_k \nabla F(w_k)^\mathsf{T} \mathbb{E}_{\xi_k}[g(w_k, \xi_k)] + \frac{\alpha_k^2 L}{2} \mathbb{E}_{\xi_k}[\|g(w_k, \xi_k)\|_2^2]. \tag{2.4}$$

Proof. See [BCN18, Lemma 4.2]. □

Lemma 2.2.3 shows that Algorithm 1 behaves similarly to a *Markov chain*. Regardless of how the iterates arrived at the step w_k, the expected descent just depends on two properties:

(i) the expected directional derivative of F at w_k along the direction $g(w_k, \xi_k)$, and

(ii) the second moment[1] of $g(w_k, \xi_k)$.

[1] If we recall (1.8) and assume that $g(w_k, \xi_k)$ is an unbiased estimate of $\nabla F(w_k)$, Lemma 2.2.3 yields

$$\mathbb{E}_{\xi_k}[F(w_{k+1})] - F(w_k) \leq -\alpha_k \|\nabla F(w_k)\|_2^2 + \frac{1}{2}\alpha_k^2 L \mathbb{E}_{\xi_k}[\|g(w_k, \xi_k)\|_2^2].$$

If we could ensure that the right-hand side of the inequality in (2.4) is negative and bounded from above by a *deterministic* quantity, this would lead to an asymptotically sufficient descent in F. Therefore, we need more assumptions regarding the first and second moments of the stochastic directions. But first of all, we define the variance with respect to the distribution of ξ_k as

$$\begin{aligned}\mathbb{V}\mathrm{ar}_{\xi_k}[g(w,\xi_k)] &:= \mathbb{E}_{\xi_k}\left[(g(w,\xi_k) - \mathbb{E}_{\xi_k}[g(w,\xi_k)])^2\right] \\ &\overset{(*)}{=} \mathbb{E}_{\xi_k}\left[\|g(w,\xi_k)\|_2^2\right] - \|\mathbb{E}_{\xi_k}[g(w,\xi_k)]\|_2^2\,, \end{aligned} \tag{2.5}$$

for $w \in \mathbb{R}^d$. The equality $(*)$ was proven in the course *Probability Theory* [Kle13, p. 103] and is a basic property of the variance.

Assumption 2.2.4 (Limits for first and second moments)**.** *We assume that the objective function $F : \mathbb{R}^d \to \mathbb{R}$ satisfies the following quantities:*

(i) *$\{w_k\}_k \subseteq \mathcal{O}$, for an open set $\mathcal{O} \subseteq \mathbb{R}^d$, in which the objective function is bounded from below. In particular: for all $w \in \mathcal{O}$ it holds $F(w) \geq F_{\inf}$.*

(ii) *There exist constants $\mu_G \geq \mu > 0$, such that*

$$\nabla F(w_k)^{\mathsf{T}} \mathbb{E}_{\xi_k}[g(w_k,\xi_k)] \geq \mu \left\|\nabla F(w_k)\right\|_2^2 \tag{2.6a}$$

$$\left\|\mathbb{E}_{\xi_k}[g(w_k,\xi_k)]\right\|_2 \leq \mu_G \left\|\nabla F(w_k)\right\|_2 \tag{2.6b}$$

hold for all $k \in \mathbb{N}$.

(iii) *There are constants $M, M_V \geq 0$, such that the inequality*

$$\mathbb{V}\mathrm{ar}_{\xi_k}\left[g(w_k,\xi_k)\right] \leq M + M_V \left\|\nabla F(w_k)\right\|_2^2 \tag{2.7}$$

is satisfied for all $k \in \mathbb{N}$.

We want to briefly interpret Assumption 2.2.4:

(i) The first assumption states, that the objective function F is bounded from below in the area explored by the algorithm. Basically, we had the same assumption in the analysis of a generic descent algorithm in *Optimization 1.*

(ii) This point is also similar to a condition of the full gradients analysis. We want to make sure that, in expectation, the algorithm's directions are of sufficient descent. We will see this in Lemma 2.2.5.

(iii) The last assumption restricts the variance in a minor way. If, for example F is a convex quadratic function, the inequality (2.7) allows the variance to be non-zero in every stationary point of F and to grow quadratically in each direction.

Combining Assumption 2.2.4 and Eq. (2.5) lead to

$$\begin{aligned}\mathbb{E}_{\xi_k}\left[\|g(w_k,\xi_k)\|_2^2\right] &= \mathbb{V}\mathrm{ar}_{\xi_k}[g(w_k,\xi_k)] + \|\mathbb{E}_{\xi_k}[g(w_k,\xi_k)]\|_2^2 \\ &\leq M + M_G\|\nabla F(w_k)\|_2^2, \end{aligned} \tag{2.8}$$

for a constant

$$M_G := M_V + \mu_G^2 \geq \mu^2 > 0. \tag{2.9}$$

Lemma 2.2.5. *Under Assumptions 2.2.1 and 2.2.4, the iterates $\{w_k\}_{k\in\mathbb{N}}$ of SGD produced by Algorithm 1 satisfy*

$$\mathbb{E}_{\xi_k}[F(w_{k+1})] - F(w_k) \leq -\mu\alpha_k\|\nabla F(w_k)\|_2^2 + \frac{1}{2}\alpha_k^2 L\mathbb{E}_{\xi_k}\left[\|g(w_k,\xi_k)\|_2^2\right] \tag{2.10a}$$

$$\leq -\left(\mu - \frac{1}{2}\alpha_k L M_G\right)\alpha_k\|\nabla F(w_k)\|_2^2 + \frac{1}{2}\alpha_k^2 LM \tag{2.10b}$$

for all $k \in \mathbb{N}$.

Proof. See [BCN18, Lemma 4.4]. □

If we take a closer look at the inequalities above and reconsider the interpretation of Lemma 2.2.3, we can see again how SGD behaves in a *Markovian manner*. No matter how SGD arrived at w_k, the random variable w_{k+1} only depends on w_k and ξ_k and *not* on past iterates.

Notice that the difference on the left-hand side is bounded from above by a deterministic quantity.

2.3. SGD for strongly convex objective functions

In this section, we will proceed our analysis of Lemma 2.2.5, by assuming that our objective function is strongly convex. In a first observation, we will see that in expectation the difference of the left-hand side will converge to a deterministic (in general non-zero) quantity, while using SGD with a fixed step size.

Naturally, we will try to refine this result by using a diminishing step size sequence to achieve convergence in expectation[2].

Assumption 2.3.1 (Strongly convex objective function)**.** *We assume that for our generic objective function $F : \mathbb{R}^d \to \mathbb{R}$ there exists a constant $c > 0$ (the so-called modulus), such that for all $(\bar{w}, w) \in \mathbb{R}^d \times \mathbb{R}^d$ the inequality*

$$F(\bar{w}) \geq F(w) + \nabla F(w)^{\mathsf{T}}(\bar{w} - w) + \frac{1}{2}c \left\|\bar{w} - w\right\|_2^2 \tag{2.11}$$

is satisfied. In particular, in that case F has a unique minimizer $w_* \in \mathbb{R}$; compare Optimization 1. Furthermore, we will define $F_* := F(w_*)$. Taking a look at Eq. (2.3)*, we can also see, that $c \leq L$ has to hold.*

The following proposition provides a useful result, which relates the minimal value F_* with any other value $F(w)$, using the Euclidean-Norm of the gradient.

Proposition 2.3.2. *Using Assumption 2.3.1, the inequality*

$$2c\,(F(w) - F_*) \leq \left\|\nabla F(w)\right\|_2^2 \tag{2.12}$$

follows for all $w \in \mathbb{R}^d$.

Proof. See [BCN18, Appendix B]. □

As mentioned above, in the following first convergence theorem we will not get convergence towards a solution. Instead, we will only get convergence to a neighbourhood of the optimal value.

Remark 2.3.3 (Total expectation)**.** In step $k \in \mathbb{N}$ of the algorithm, we will write $\mathbb{E}[\cdot]$ to denote the expected value with respect to the joint distribution of *all* random variables. Particularly, the total expectation of $F(w_k)$ can be written as

$$\mathbb{E}[F(w_k)] := \mathbb{E}_{\xi_1}\left[\mathbb{E}_{\xi_2}\left[\ldots\left[\mathbb{E}_{\xi_{k-1}}\left[F(w_k)\right]\right]\ldots\right]\right],$$

because w_k is completely determined by the independent random variables $\{\xi_1, \ldots \xi_{k-1}\}$. The goal of our analysis is to achieve convergence *in expectation* for the sequence $\{F(w_k)\}_{k\in\mathbb{N}}$ towards a local minimizer. As usual, the choice of the step size is crucial

[2]It is also possible to achieve almost sure convergence, but this requires way more work and a change of perspective, using martingale techniques; cf. [BCN18, Inset 4.1].

to obtain convergence. As we will see in the numerical analysis, due to the stochastic approach, SGD methods are more sensitive to the step size sequence than the (exact) gradient method.

Now we are able to prove our first convergence result.

Theorem 2.3.4 (Strongly convex objective function, fixed step size)**.** *Suppose that Assumption 2.2.1, 2.2.4 and 2.3.1 holds, and $F_{\inf} = F_*$. For the SGD algorithm with the fixed step size $\alpha_k = \bar{\alpha}$ for all $k \in \mathbb{N}$, satisfying*

$$0 < \bar{\alpha} \leq \frac{\mu}{LM_G}, \tag{2.13}$$

the following (contraction) inequality holds

$$\mathbb{E}\left[F(w_k) - F_*\right] \leq \frac{\bar{\alpha}LM}{2c\mu} + (1 - \bar{\alpha}c\mu)^{k-1}\left(F(w_1) - F_* - \frac{\bar{\alpha}LM}{2c\mu}\right) \overset{k\to\infty}{\longrightarrow} \frac{\bar{\alpha}LM}{2c\mu}. \tag{2.14}$$

Proof. This proof is strongly based on the proof of [BCN18, Theorem 4.6]. As this result one of the main theoretical statements, we include the proof with some additional steps.

Using Lemma 2.2.5, Proposition 2.3.2 and Eq. (2.13), we infer

$$\begin{aligned}
\mathbb{E}_{\xi_k}[F(w_{k+1}) - F(w_k)] &\overset{(2.10b)}{\leq} -\left(\mu - \frac{1}{2}\bar{\alpha}LM_G\right)\bar{\alpha}\left\|\nabla F(w_k)\right\|_2^2 + \frac{1}{2}\bar{\alpha}^2 LM \\
&\overset{(2.13)}{\leq} -\frac{1}{2}\mu\bar{\alpha}\left\|\nabla F(w_k)\right\|_2^2 + \frac{1}{2}\bar{\alpha}^2 LM \\
&\overset{(2.12)}{\leq} -\frac{1}{2}\mu\bar{\alpha}2c\left(F(w_k) - F_*\right) + \frac{1}{2}\bar{\alpha}^2 LM
\end{aligned}$$

for all $k \in \mathbb{N}$. We now subtract F_* from both sides, take the total expectation and shift $\mathbb{E}[F(w_k)]$ around, to get

$$\begin{aligned}
\mathbb{E}\left[F(w_{k+1}) - F_*\right] &\leq (1 - \mu\bar{\alpha}c)\mathbb{E}\left[F(w_k) - F_*\right] + \frac{1}{2}\bar{\alpha}^2 LM \\
\Rightarrow \mathbb{E}\left[F(w_{k+1}) - F_*\right] - \frac{\bar{\alpha}LM}{2c\mu} &\leq (1 - \mu\bar{\alpha}c)\mathbb{E}\left[F(w_k) - F_*\right] + \frac{1}{2}\bar{\alpha}^2 LM - \frac{\bar{\alpha}LM}{2c\mu} \\
&= (1 - \mu\bar{\alpha}c)\left(\mathbb{E}\left[F(w_k) - F_*\right] - \frac{\bar{\alpha}LM}{2c\mu}\right) \\
\Rightarrow \mathbb{E}\left[F(w_{k+1}) - F_*\right] - \frac{\bar{\alpha}LM}{2c\mu} &\leq (1 - \mu\bar{\alpha}c)\left(\mathbb{E}\left[F(w_k) - F_*\right] - \frac{\bar{\alpha}LM}{2c\mu}\right).
\end{aligned} \tag{2.15}$$

Eq. (2.15) is a contraction inequality, since we have

$$0 < \bar{\alpha} c \mu \overset{(2.13)}{\leq} \frac{c\mu^2}{LM_G} \overset{(2.9)}{\leq} \frac{c}{L}\frac{\mu^2}{\mu^2} \leq \frac{c}{L} \leq 1.$$

We obtain $c \leq L$ from Eq. (2.11). Now, we can apply the inequality (2.15) repeatedly, to obtain

$$\mathbb{E}\left[F(w_k) - F_*\right] \leq \frac{\bar{\alpha} L M}{2c\mu} + (1 - \bar{\alpha} c \mu)^{k-1} \left(F(w_1) - F_* - \frac{\bar{\alpha} L M}{2c\mu} \right).$$

The convergence result easily follows using the fact that the factor $1 - \bar{\alpha} c \mu$ is smaller than one and $F(w_1)$ is just a scalar, which can be factored out of the expected value, as it is independent of the joint distribution. □

Let us remark a few things, regarding the results of Theorem 2.3.4. First of all, assume $g(w_k, \xi_k)$ being an unbiased estimate of $\nabla F(w_k)$ and having no noise in $g(w_k, \xi_k)$, e.g. $\mu = 1$ and $M_G = 1$. In this case, the process of selecting a fixed step size reduces to $\alpha \in \left(0, \frac{1}{L}\right]$ – a classical step size for SGD. Furthermore, Theorem 2.3.4 indicates the important interplay between the step size and the bound of the variance. If we have no noise in the computation of the gradient, Theorem 2.3.4 states a linear convergence rate of SGD. But, if the computation is too noisy, the fixed step size only leads to a neighbourhood of the optimum.

Consequently, we will now consider SGD methods with descending step size sequences.

Remark 2.3.5 (SGD in practice)**.** Let us assume running SGD with an initial step size $\alpha_1 \in \left(0, \frac{\mu}{LM_G}\right]$ for the iterations $k_1 = 1$ to k_2, whereas w_{k_2} is the first iterate to satisfy

$$\mathbb{E}[F(w_{k_2}) - F_*] \leq 2 \cdot F_{\alpha_1},$$

with $F_\alpha := \frac{\alpha L M}{2c\mu}$. Recalling Theorem 2.3.4, we obtain that F_α is the limit of the expected optimality gap, if we run SGD with fixed step size $\bar{\alpha} = \alpha_1$. Now, we cut the step size in half, meaning $\alpha_2 := \frac{\alpha_1}{2}$. This results in a step size sequence $\{\alpha_{r+1}\} = \{\alpha_1 2^{-r}\}$ with an index sequence $\{k_r\}_{r \in \mathbb{N}}$. Now we can see that the sequence

$$\{F_{\alpha_r}\}_{r \in \mathbb{N}} = \left\{ \frac{\alpha_r L M}{2c\mu} \right\}_{r \in \mathbb{N}}$$

satisfies

$$\lim_{r\to\infty} F_{\alpha_r} = 0,$$

as the diminishing step size sequence converges to zero.

We will now analyse the behaviour of the optimality gap with respect to the new step size sequence $\{\alpha_r\}_{r\in\mathbb{N}}$ more closely. For all $r \in \mathbb{N}_{\geq 2}$, we have

$$\mathbb{E}[F(w_{k_{r+1}}) - F_*] \leq 2 \cdot F_{\alpha_r}. \tag{2.16}$$

Due to our choice of the step size sequence, we can assume that

$$\mathbb{E}[F(w_{k_r}) - F_*] \approx 2 \cdot F_{\alpha_{r-1}} = 4F_{\alpha_r}.$$

Next, we want to calculate the *effective rate* of step size decrease we need in order to obtain the inequality (2.16). Therefore, we can invoke Theorem 2.3.4 to calculate the difference $k_{r+1} - k_r$ as

$$\begin{aligned}\mathbb{E}[F(w_{k_{r+1}}) - F_*] &\overset{(2.14)}{\leq} \frac{\alpha_r LM}{2c\mu} + (1-\alpha_r c\mu)^{k_{r+1}-k_r}\left(\mathbb{E}\left[F(w_{k_r}) - F_*\right] - \frac{\alpha_r LM}{2c\mu}\right) \overset{!}{\leq} 2F_{\alpha_r} \\ &\Rightarrow (1-\alpha_r\mu c)^{k_{r+1}-k_r}\left(4F_{\alpha_r} - F_{\alpha_r}\right) \leq F_{\alpha_r}.\end{aligned}$$

As a consequence, we get

$$k_{r+1} - k_r \geq \frac{\log\left(\frac{1}{3}\right)}{\log\left(1-\alpha_r\mu c\right)} \approx \frac{\log(3)}{\alpha_r\mu c} = \mathcal{O}(2^r),$$

where we used that the Taylor expansion of log at $x_0 = 1$ has the form

$$\log(x) = (x-1) + \mathcal{O}(\|x-1\|^2)$$

and

$$\log\left(\frac{1}{3}\right) = \log(1) - \log(3) = -\log(3)$$

holds. This result is essential because it states that cutting the step size in half, leads to a doubled number of iterations – a *sublinear rate of convergence*. An example of this behaviour is illustrated in Fig. 2.1.

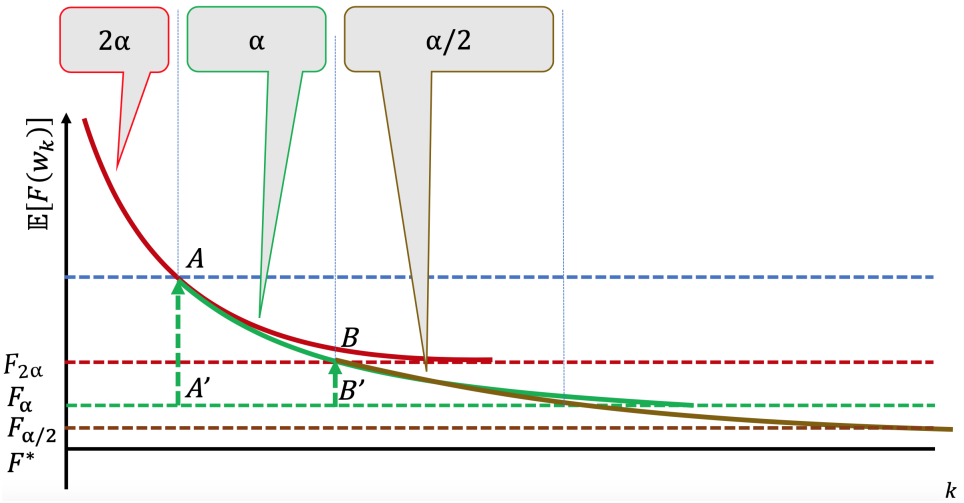


Figure 2.1.: Exemplary sublinear convergence behaviour, with a step size choice stated in Remark 2.3.5. One can see, the overall effective rate is at the order of $\mathcal{O}(\frac{1}{k})$. Source: [BCN18, Figure 4.1].

In the next theorem, we will analyse SGD methods with general diminishing step size sequences of the form

$$\sum_{k=1}^{\infty} \alpha_k = \infty \text{ and } \sum_{k=1}^{\infty} \alpha_k^2 < \infty. \tag{2.17}$$

The rate of convergence will stay the same, but one has more flexibility to choose the step size sequence.

Theorem 2.3.6 (Strongly convex objective function, diminishing step sizes)**.** *Under Assumptions 2.2.1, 2.2.4, 2.3.1 and $F_{\text{inf}} = F_*$, we consider the iterates $\{w_k\}_{k\in\mathbb{N}}$, generated by SGD with a step size sequence satisfying*

$$\{\alpha_k\}_{k\in\mathbb{N}} = \left\{\frac{\beta}{\gamma + k}\right\}_{k\in\mathbb{N}} \quad \textit{with } \alpha_1 = \frac{\beta}{\gamma+1} \leq \frac{\mu}{LM_G}, \tag{2.18}$$

for some constants

$$\beta > \frac{1}{c\mu}, \gamma > 0.$$

Then, we get for all iterations $k \in \mathbb{N}$

$$\mathbb{E}[F(w_k) - F_*] \leq \frac{\nu}{\gamma + k}, \tag{2.19}$$

as an expected optimality gap, where

$$\nu := \max\left\{\frac{\beta^2 LM}{2(\beta c\mu - 1)}, (\gamma+1)\left(F(w_1) - F_*\right)\right\}. \tag{2.20}$$

Proof. This proof is strongly based on the proof of [BCN18, Theorem 4.7]. As this one of the main theoretical results, we include the proof with some additional steps. Using $\alpha_k = \frac{\beta}{\gamma+k}$, we get for all $k \in \mathbb{N}$

$$\alpha_k L M_G \leq \alpha_1 L M_G \leq \mu. \tag{$*$}$$

Now, we can use Lemma 2.2.5 and Proposition 2.3.2 and obtain

$$\begin{aligned}
\mathbb{E}_{\xi_k}[F(w_{k+1})] - F(w_k) &\overset{(2.10b)}{\leq} -\left(\mu - \frac{1}{2}\alpha_k L M_G\right)\alpha_k \left\|\nabla F(w_k)\right\|_2^2 + \frac{1}{2}\alpha_k^2 LM \\
&\overset{(*)}{\leq} -\frac{1}{2}\mu\alpha_k \left\|\nabla F(w_k)\right\|_2^2 + \frac{1}{2}\alpha_k^2 LM \\
&\overset{(2.12)}{\leq} -\alpha_k c\mu\left(F(w_k) - F(w_*)\right) + \frac{1}{2}\alpha_k^2 LM.
\end{aligned}$$

If we subtract F_* from both sides, take the expected value and rearrange the inequality, like in the proof of Theorem 2.3.4, it follows that

$$\mathbb{E}[F(w_{k+1})] - F_*] \leq (1 - \alpha_k c\mu)\mathbb{E}\left[F(w_k) - F(w_*)\right] + \frac{1}{2}\alpha_k^2 LM \tag{2.21}$$

holds. Now we can prove the statement of the theorem by induction.
$\underline{k = 1:}$ We have to show the inequality

$$\mathbb{E}[F(w_1)] - F_*] = F(w_1) - F_* \leq \frac{\nu}{\gamma+1},$$

which is a direct consequence of the definition of ν.
$\underline{k \to k+1:}$ We assume Eq. (2.19) holds for a $k \geq 1$, using (2.21), $\hat{k} := \gamma + k$ and $\alpha_k = \frac{\beta}{\gamma+k}$, we obtain

$$\begin{aligned}
\mathbb{E}[F(w_{k+1})] - F_*] &\overset{(2.21)}{\leq} (1 - \alpha_k c\mu)\mathbb{E}\left[F(w_k) - F(w_*)\right] + \frac{1}{2}\alpha_k^2 LM \\
&\leq \left(1 - \frac{\beta}{\hat{k}} c\mu\right)\frac{\nu}{\hat{k}} + \frac{\beta^2 LM}{2\hat{k}^2} = \left(\frac{\hat{k} - \beta c\mu}{\hat{k}^2}\right)\nu + \frac{\beta^2 LM}{2\hat{k}^2}
\end{aligned}$$

$$= \left(\frac{\hat{k}-1}{\hat{k}^2}\right)\nu - \left(\frac{\beta c\mu - 1}{\hat{k}^2}\right)\nu + \frac{\beta^2 LM}{2\hat{k}^2}$$
$$\leq \left(\frac{\hat{k}-1}{\hat{k}^2}\right)\nu. \tag{2.22}$$

The last inequality in (2.22) holds, because by the definition of ν we have

$$\nu \geq \frac{\beta^2 LM}{2(\beta c\mu - 1)},$$

which implies

$$\left(\frac{\beta c\mu - 1}{\hat{k}^2}\right)\nu \geq \frac{\beta^2 LM}{2\hat{k}^2}$$
$$\Rightarrow -\left(\frac{\beta c\mu - 1}{\hat{k}^2}\right)\nu + \frac{\beta^2 LM}{2\hat{k}^2} \leq 0.$$

Using $\hat{k}^2 \geq \hat{k}^2 - 1 = (\hat{k}-1)(\hat{k}+1)$, we get

$$\frac{\hat{k}-1}{\hat{k}^2} \leq \frac{1}{\hat{k}+1}. \tag{2.23}$$

Combing the inequalities (2.22) and (2.23) leads to

$$\mathbb{E}[F(w_{k+1})] - F_*] \leq \frac{\nu}{\gamma + (k+1)},$$

which is our stated result. □

Finally, we want to interpret the results of Theorem 2.3.4 and Theorem 2.3.6.

Role of the strong convexity The parameter $c > 0$, gained by Assumption 2.3.1, occurs in both convergence theorems to achieve a contraction inequality. The impact of c on the step size differs between the strategies. While running SGD with a fixed step size, the interval to choose $\bar{\alpha}$ is independent of c, but for SG with a diminishing step size, the condition $\beta > \frac{1}{c\mu}$ is crucial to obtain a convergence rate of $\mathcal{O}\left(\frac{1}{k}\right)$.

Role of the initial guess In both theorems 2.3.4 and 2.3.6, the term $F(w_1) - F_*$ occurs. For a fixed step size, we get the initial gap with an exponentially decreasing factor.

For a diminishing step size sequence, one can spot the initial gap in the second term of the value ν. In [BCN18, p. 29] it is shown, that the first term of ν dominates the asymptotic behaviour. We will create a small example, which will illustrate this behaviour.

Assume running SGD with a fixed step size $\bar{\alpha}$ until step k_1, where

$$F(w_{k_1}) - F_* \leq \frac{\bar{\alpha} LM}{2c\mu},$$

as in Remark 2.3.5. Then choosing β, γ, as in Theorem 2.3.6 and $\bar{\alpha} = \alpha_1$, we get

$$(\gamma + 1)\mathbb{E}\left[F(w_{k_1}) - F_*\right] = \frac{\beta}{\alpha_1}\mathbb{E}\left[F(w_{k_1}) - F_*\right] \leq \frac{\beta}{\alpha_1}\frac{\alpha_1 LM}{2c\mu} = \frac{\beta LM}{2c\mu} < \frac{\beta^2 LM}{2(\beta c\mu - 1)}.$$

Therefore, the first term of ν in (2.20) asymptotically dominates.

We should always try to choose the initial step size as big as possible, e.g. $\alpha_1 = \frac{\mu}{LM_G}$. We obtain the best asymptotic behaviour, if the first value of ν is as small as possible, e.g. $\beta = \frac{2}{c\mu}$ (since only the first term is asymptotically relevant). In this case we get

$$\nu = \frac{\beta^2 LM}{2(\beta c\mu - 1)} = \frac{\left(\frac{2}{c\mu}\right)^2 LM}{2\left(\frac{2}{c\mu}c\mu - 1\right)} = \frac{2}{\mu^2}\frac{L}{c}\frac{M}{c}.$$

The last two ratios are essential, as they set the Lipschitz constant and the lower noise bound in proportion to the modulus of convexity.

2.4. Outlook: SGD for general objective functions

In this section we want to expand our analysis of SGD to non-convex objective functions. As we see in most practical application fields (e.g. the majority of machine learning models), the objective function is in general non-convex and therefore, further analysis of the SGD algorithm is needed. The analysis for non-convex objective functions is more challenging, as such functions can possess multiple local minima and other stationary points. The main result of this section will be the generalization of both of the previous theorems, but at the expense of the convergence factor in the case of a diminishing step size sequence. We want to make the same assumptions as before, except for the strong convexity of the objective function.

First, we want to take a look at the sequence of gradients of F when running SGD with fixed step sizes and hence try to generalize Lemma 2.2.3 for the non-convex case.

Theorem 2.4.1 (Non-convex objective function, fixed step size)**.** *Suppose that Assumptions 2.2.1 and 2.2.4 hold. Further, assume running SGD with a fixed step size, $\alpha_k = \bar{\alpha}$, for all $k \in \mathbb{N}$, satisfying the usual condition*

$$0 < \bar{\alpha} \leq \frac{\mu}{LM_G}. \tag{2.24}$$

Then, the expected sum-of-squares and average-squared gradients of the objective function F generated by the SGD iterates, satisfy the following inequalities for all $K \in \mathbb{N}$:

$$\mathbb{E}\left[\sum_{k=1}^{K} \|\nabla F(w_k)\|_2^2\right] \leq \frac{K\bar{\alpha}LM}{\mu} + \frac{2 \cdot (F(w_1) - F_{\text{inf}})}{\mu\bar{\alpha}} \tag{2.25a}$$

$$\text{and therefore } \mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K} \|\nabla F(w_k)\|_2^2\right] \leq \frac{\bar{\alpha}LM}{\mu} + \frac{2 \cdot (F(w_1) - F_{\text{inf}})}{K\mu\bar{\alpha}} \tag{2.25b}$$

$$\xrightarrow{K\to\infty} \frac{\bar{\alpha}LM}{\mu}.$$

Proof. See [BCN18, Theorem 4.8]. □

As in the result of Theorem 2.3.4, the right-hand side of the inequality contains the fixed step size $\bar{\alpha}$, a Lipschitz constant of the objective functions gradient, as well as the constants obtained by the assumptions regarding the first and second moments of the stochastic directions $g(w_k, \xi_k)$. The case of $M = 0$, meaning there is no noise or the noise reduces proportionally to $\|\nabla F(w_k)\|_2^2$, is also similar to Theorem 2.3.4.

Here, the sum of the squared gradients remains finite, implying $\{\|\nabla F(w_k)\|_2\} \to 0$ for $k \to \infty$.

In the situation of $M > 0$, Theorem 2.4.1 illustrates the interplay between the step size $\bar{\alpha}$ and the variance of the stochastic directions. While we lose the upper bound for the expected optimality gap due to the non-convexity of the objective function, inequality (2.25b) bounds the average of the objective functions gradient observed on $\{w_k\}_k$ during the first K iteration steps. The quantity on the right-hand side gets smaller when K increases, indicating that SGD spends increasingly more time in regions where the objective function has a (relatively) small gradient. This problem arises in most practical machine learning projects (a so-called *learning plateau*).

As we have already seen in the convex case, noise in the gradients delays further progress. One can decrease the upper bound for the average norm of the gradients to an arbitrarily small value by reducing the step size. However, Remark 2.3.5 illustrates that this tremendously reduces the speed of approaching this limit.

Now, we want to expand our result to the case with a diminishing step size sequence. We first state the next theorem and will later prove it as a corollary.

Theorem 2.4.2 (Non-convex objective function, diminishing step sizes)**.** *Under Assumptions 2.2.1 and 2.2.4, suppose running the SGD method with a step size sequence* $\{\alpha_k\}_{k\in\mathbb{N}}$*, satisfying*

$$\sum_{k=1}^{\infty} \alpha_k = \infty \text{ and } \sum_{k=1}^{\infty} \alpha_k^2 < \infty. \tag{2.26}$$

Then we obtain

$$\liminf_{k\to\infty} \mathbb{E}\left[\|\nabla F(w_k)\|_2^2\right] = 0 \tag{2.27}$$

for the sequence of gradients generated by SGD.

The proof of this theorem is a direct consequence of the next stated theorem. A "lim inf" result of this type is typical in the context of stochastic.

Theorem 2.4.3 (Non-convex objective function, diminishing step sizes)**.** *We again assume running the SGD method under Assumptions 2.2.1 and 2.2.4 with a diminishing step size sequence satisfying the conditions in* (2.26)*. Then, with* $A_K := \sum_{k=1}^{K} \alpha_k$*,*

for $K \in \mathbb{N}$, we obtain

$$\lim_{K\to\infty} \mathbb{E}\left[\sum_{k=1}^{K} \alpha_k \left\|\nabla F(w_k)\right\|_2^2\right] < \infty \tag{2.28a}$$

$$\textit{and therefore } \mathbb{E}\left[\frac{1}{A_k}\sum_{k=1}^{K} \alpha_k \left\|\nabla F(w_k)\right\|_2^2\right] \overset{K\to\infty}{\longrightarrow} 0. \tag{2.28b}$$

Proof. See [BCN18, Theorem 4.10]. □

In contrast to Theorem 2.4.1, the result of Theorem 2.4.3 states that the weighted average norm of the square gradients converges to zero – even under the presence of noise.

We can see now, why Theorem 2.4.2 is a direct consequence of Theorem 2.4.3. If Eq. (2.26) would not hold, it would contradict Theorem 2.4.3.

This also negates the fact that (2.28b) only specifies a property of a weighted average, as we can still conclude that the expected gradient norms cannot asymptotically stay far away from zero (cf. Theorem 2.4.2).

We can improve the "lim inf" convergence to a stronger convergence in probability, at the expense of only showing this property at randomly selected iterates of the objective functions gradient.

Corollary 2.4.4 (Convergence in probability for randomly selected iterates)**.** *We assume the conditions of Theorem 2.4.3 hold. For any $K \in \mathbb{N}$, let $k(K) \in \{1, \dots, K\}$ represent a randomly chosen index with probabilities proportional to $\{\alpha_k\}_{k=1}^K$. Then $\left\|\nabla F(w_{k(K)})\right\|_2^2 \to 0$ converges in probability as $K \to \infty$.*

Proof. The proof is a direct consequence of the Markov inequality (cf. Lemma A.2.2) and the result given in Eq. (2.28a). For arbitrary $\varepsilon > 0$, we can write

$$\begin{aligned}\mathbb{P}\left(\left\|\nabla F(w_{k(K)})\right\|_2 \geq \varepsilon\right) &= \mathbb{P}\left(\left\|\nabla F(w_{k(K)})\right\|_2^2 \geq \varepsilon^2\right) \\ &\overset{\text{Markov}}{\leq} \frac{\mathbb{E}\left[\mathbb{E}_k\left[\left\|\nabla F(w_{k(K)})\right\|_2^2\right]\right]}{\varepsilon^2} \overset{K\to\infty}{\longrightarrow} 0.\end{aligned}$$

This is exactly the definition of convergence in probability (cf. Definition A.2.1). □

Assuming additional regularity conditions, we can show a stronger convergence result.

Corollary 2.4.5. *We assume the conditions of Theorem 2.4.3 hold. Furthermore, let the objective function F be twice differentiable and we assume that the mapping $w \mapsto \|\nabla F(w)\|_2^2$ has a Lipschitz continuous derivative. Then,*

$$\lim_{k\to\infty} \mathbb{E}\left[\|\nabla F(w_k)\|_2^2\right] = 0.$$

Proof. See [BCN18, Appendix B]. □

In the next section we will analyse concrete implementations of various SGD variations and compare their behaviour using different objective functions.

Chapter 3.

Implementation and numerical analysis

3.1. Sensitivity to step sizes

As already mentioned in Remark 2.3.3, SGD, running with fixed step size, is very sensitive to the choice of $\bar{\alpha}$. We will illustrate this behaviour by considering the optimization problem $\min_{x\in\mathbb{R}^2} F(x)$, where

$$F : \mathbb{R}^2 \to \mathbb{R}_0^+, x \mapsto \frac{1}{2} \|Ax - b\|_2^2 = \frac{1}{2} \sum_{i=1}^{2} (A_{i\cdot}^{\mathsf{T}} x - b_i)^2 \tag{3.1}$$

and A is the symmetric, positive definite matrix

$$A := \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}.$$

With $A_{i\cdot}$, we denote the i-th row of the matrix A and choose $b := (0,0)^{\mathsf{T}}$ for simplicity. So we optimize for the kernel, $\ker(A)$, of the matrix A. We will use different step sizes but keep, for the sake of comparability, the maximum number of iterations fixed to $\text{iter}_{\max} = 100$ and the initial guess is always $x_0 := (1,1)^{\mathsf{T}}$.

Clearly, because A is a positive definite matrix, this problem has the unique minimizer $x_* = (0,0)^{\mathsf{T}}$ which will be marked with a red cross in the contour plots. To compare SGD with the exact gradient descent, we will minimize the function F also using the standard gradient descent method with $\text{tol} = 10^{-1}$, $\text{iter}_{\max} = 100$ and the Powell-Wolfe conditions for the line search algorithm. The behaviour of SGD will be plotted in orange and the behaviour of the exact gradient descent in green.

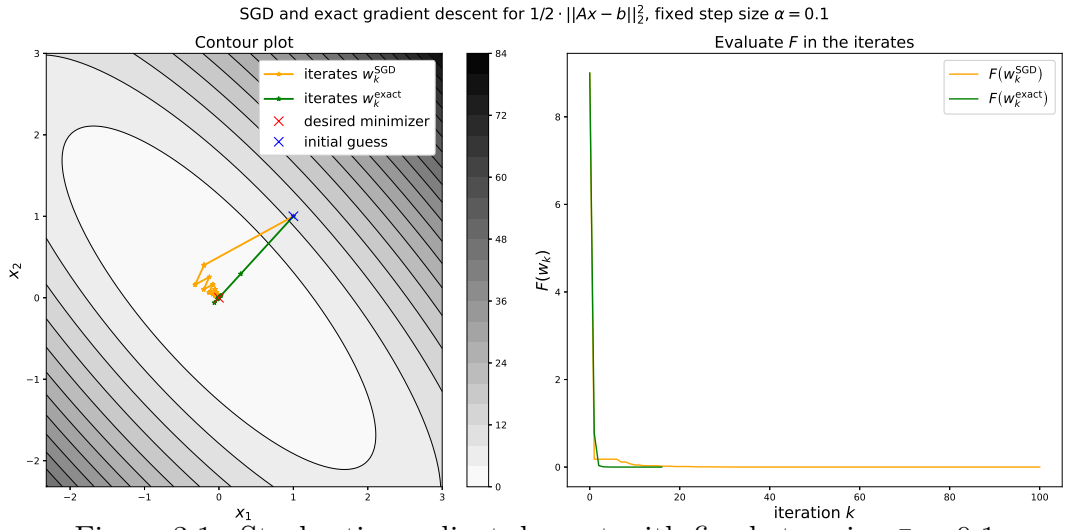


Figure 3.1.: Stochastic gradient descent with fixed step size $\bar{\alpha} = 0.1$.

Similar to Section 3.2.1, one can differentiate the system (3.1), compute the eigenvalues of the matrix A and get $\bar{\alpha}_{\max} = \frac{1}{9} = 0.\overline{1}$ as an upper bound for the step size of SGD. In the following four plots, we will first take a look at the behaviour of SGD running with a step size smaller than the upper bound $\bar{\alpha}_{\max}$, before analysing the algorithm's performance with an overestimated step size.

In the first plot, Fig. 3.1, we run SGD with the fixed step size $\bar{\alpha} = 0.1$ – a step size smaller than the calculated upper bound. The convergence plot is clear and we obtain good results, even after less than the maximum number of iterations. We also see, that the iterates tend in a clear way towards the (global) minimizer. Compared to the (exact) gradient descent, the directions calculated by SGD are not (necessarily) orthogonal to the contour lines. This property is lost by not using the full gradient and can be observed in all graphs Fig. 3.1 to 3.4. In those four plots Fig. 3.1 to 3.4, we can also observe that the exact gradient descent method always converges and is faster than SGD, due to the higher computational effort by using the exact gradient.

Multiplying the initial $\bar{\alpha}$ with the factor 1.5, the curve contains more noise and the behaviour in a local neighbourhood of x_* is a bit more chaotic. On the right-hand side of Fig. 3.2, we can now also see, that not all the directions used by the algorithm reduce the value of the loss function, as the plot has small peaks, where the value of the function rises again. But asymptotically, we obtain convergence. So the step size is still small enough to guarantee convergence in expectation, although we overestimated $\bar{\alpha}$ compared to the analytically calculated upper bound.

If we enlarge $\bar{\alpha}$ again, now to $\bar{\alpha} = 0.19$, both plots get more chaotic. Not only

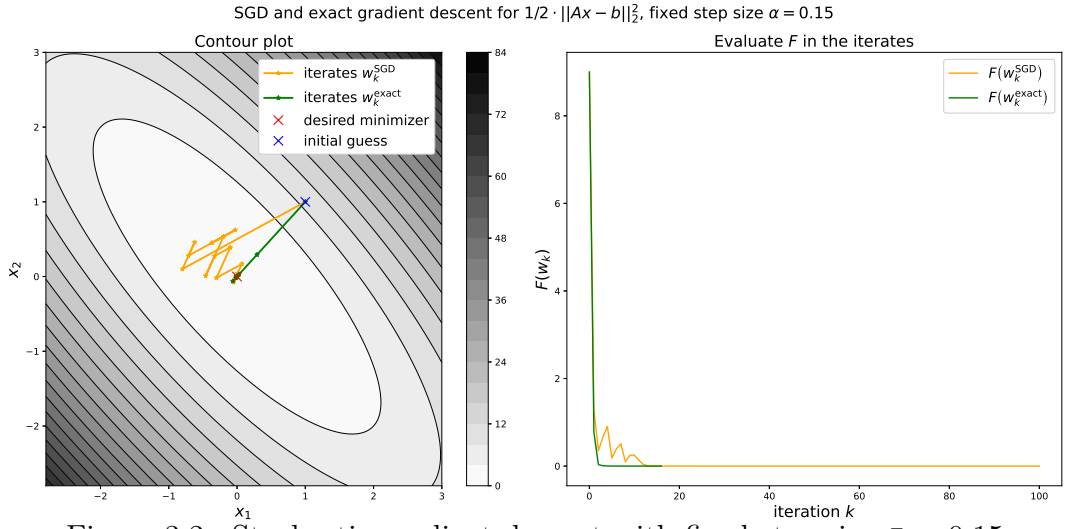


Figure 3.2.: Stochastic gradient descent with fixed step size $\bar{\alpha} = 0.15$.

the peaks in the plot of the evaluated objective function are bigger, but also the algorithm seems to be very unstable in the neighbourhood of x_*. Overall, we still obtain asymptotic convergence.

Now, we take a look at the last plot, Fig. 3.4, where the step size is doubled to the initial step size and nearly twice as big as the upper bound $\bar{\alpha}_{\max}$. We see, why the last choice of $\bar{\alpha}$ is the value 0.19. This time the step size $\bar{\alpha} = 0.2$ is too large. The algorithm cannot handle the amount of chaos, generated by the highly overestimated step size. We do not obtain convergence in expectation anymore.

To clarify some notation, we need one additional remark.

Remark 3.1.1. We denote the minimal eigenvalue of a symmetric matrix $A \in \mathbb{R}^{d\times d}$, $d \in \mathbb{N}$, as usual with $\mu_{\min}(A)$ and the maximal eigenvalue with $\mu_{\max}(A)$. So we have $\mu_{\min}(A) \leq \mu_i(A) \leq \mu_{\max}(A)$ for all $i = 1, \ldots, d$ (ignoring multiplicities).

3.2. Two parametrized boundary value problems under uncertainties

These first plots should only illustrate the general behaviour of SGD running with a fixed step size. In a further analysis, we want to take a closer look at SGD running with a diminishing step size sequence for strongly convex objective functions.

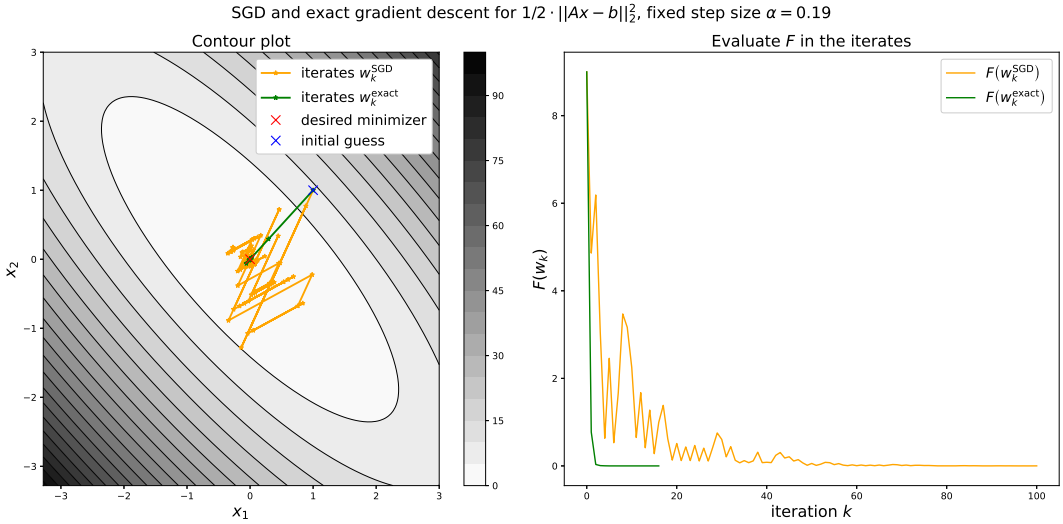


Figure 3.3.: Stochastic gradient descent with fixed step size $\bar{\alpha} = 0.19$.

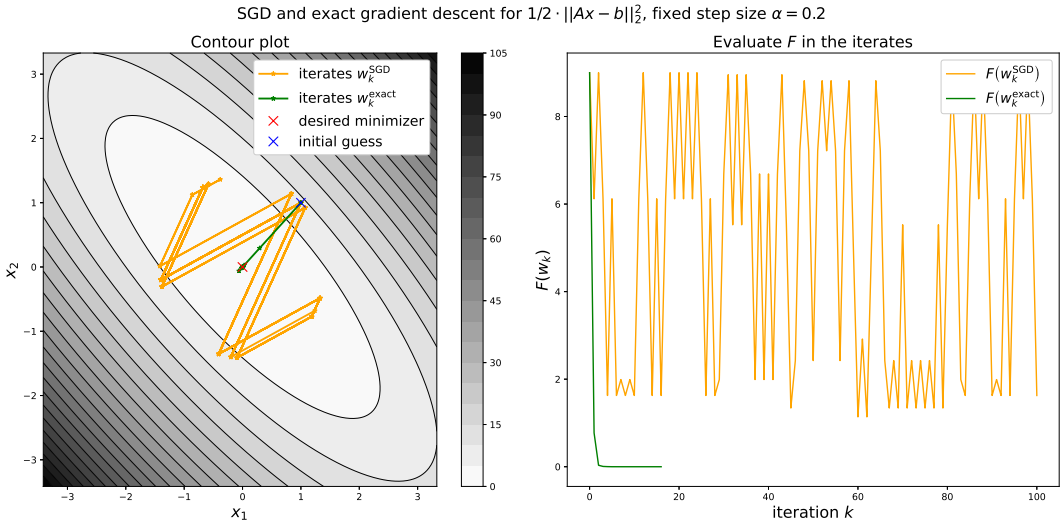


Figure 3.4.: Stochastic gradient descent with fixed step size $\bar{\alpha} = 0.2$.

3.2.1. ODE Dirichlet problem

First, let us consider the convection–diffusion equation for a parameter $\lambda \in \mathbb{R}^+$ and $\Omega := (0,1) \subset \mathbb{R}$. Using Dirichlet boundary conditions, we consider the boundary value problem

$$\left.\begin{aligned} -u''(x) + \lambda u(x) &= f(x), x \in \Omega \\ u(0) = u(1) &= 0 \end{aligned}\right\}, \tag{3.2}$$

for some $f \in C^2(\Omega, \mathbb{R})$. In this chapter, we aim for a (strong) solution $u \in C^2(\overline{\Omega})$ and therefore use finite differences (FD) to approximately solve this ordinary differential equation (ODE).

Following [DR08, p. 69], we discretize the ODE (3.2) using $n+2$, $n \in \mathbb{N}$, grid points $x_j = j \cdot h$ for $j = 0, \ldots, n+1$ and mesh size $h = \frac{1}{n+1}$. For $j = 0, \ldots, n+1$ we define discrete approximations[1] of (the unknown) u and f

$$\begin{aligned} u_j &\simeq u(x_j) \\ \text{and set } f_j &:= h^2 \cdot f(x_j). \end{aligned}$$

Using the boundary conditions, we have $u_0 = u_{n+1} = 0$. Now we can define vectorized versions of u and f

$$\begin{aligned} u &:= (u_j)^{\mathsf{T}}_{j=1,\ldots,n} \in \mathbb{R}^n \\ f &:= (f_j)^{\mathsf{T}}_{j=1,\ldots,n} \in \mathbb{R}^n, \end{aligned}$$

and the symmetric matrix

$$A(\lambda) := \begin{pmatrix} 2+\lambda h^2 & -1 & 0 & \cdots & 0 \\ -1 & 2+\lambda h^2 & -1 & \ddots & \vdots \\ 0 & -1 & 2+\lambda h^2 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & 0 & -1 & 2+\lambda h^2 \end{pmatrix} \in \mathbb{R}^{n \times n},$$

to obtain the linear system

$$A(\lambda) u = f. \tag{3.3}$$

[1] The goal is to get convergence of the form $\max_{i=0,\ldots,n+1} |u_j - u(x_j)| \to 0$ as $h \to 0$.

Solving Eq. (3.3) for $u \in \mathbb{R}^n$ is equivalent to solving the minimization problem

$$\min_{u \in \mathbb{R}^n} F(u; \lambda), \tag{3.4}$$

$$\text{with } F(\cdot; \lambda) : \mathbb{R}^n \to \mathbb{R}, u \mapsto \frac{1}{2} \|A(\lambda)u - f\|_2^2 . \tag{3.5}$$

Differentiating the function $F(\cdot; \lambda)$ (twice) with respect to u, we get

$$\nabla_u F(u; \lambda) = A(\lambda)^\mathsf{T} (A(\lambda)u - f) = A(\lambda) (A(\lambda)u - f) , \tag{3.6}$$

$$\nabla_u^2 F(u; \lambda) = A(\lambda)^\mathsf{T} A(\lambda) \overset{A(\lambda) \text{ symm.}}{=} A(\lambda)^2. \tag{3.7}$$

Recalling [CG20, Chapter 2, Problem 10], we can exploit the tridiagonal Toeplitz-structure of the matrix $A(\lambda)$, to calculate the eigenvalues

$$\mu_k = 2 + \lambda h^2 + 2 \cos \left(\frac{\pi k}{n+1} \right), k = 1, \ldots, n. \tag{3.8}$$

We get the following upper and lower bound for the eigenvalues:

$$\mu_k \leq 4 + \lambda h^2, k = 1, \ldots, n \tag{3.9}$$

$$\mu_k \geq \lambda h^2, k = 1, \ldots, n. \tag{3.10}$$

Now it is clear that $A(\lambda)$ is positive definite for every $\lambda > 0$ and, according to Corollary A.1.3, $A(\lambda)$ is therefore even uniform positive definite.

Furthermore, we want to simulate uncertainties (or noise) in the calculation of the gradient. Hence, we assume $0 < \lambda_{\min} < \lambda_{\max}$ and that $\lambda_{\text{exact}} = \frac{\lambda_{\min} + \lambda_{\max}}{2}$ holds, but λ has a perturbation in the range of $[\lambda_{\min}, \lambda_{\max}]$. For $N \in \mathbb{N}$ and normally distributed[2] $\{\lambda_i\}_{i=1}^N \subset [\lambda_{\min}, \lambda_{\max}]$, we define the function

$$\begin{aligned} &R_N (F(\cdot; \lambda)) : \mathbb{R}^n \to \mathbb{R} \\ &\qquad u \mapsto \frac{1}{N} \sum_{i=1}^N F(u; \lambda_i) = \frac{1}{2N} \sum_{i=1}^N \|A(\lambda_i)u - f\|_2^2 . \end{aligned} \tag{3.11}$$

[2]To get normally distributed values in this interval, we simulate a normal distribution with mean λ_{exact} and a standard deviation of $\sigma := 1$ and afterwards perform a hard cut off. So values smaller than $\lambda_{\min}$ are set to the lower bound and values greater than $\lambda_{\max}$ are set to the upper bound.

Our goal is now to solve the following minimization problem

$$\min_{u\in\mathbb{R}^n} R_N\left(F(u;\lambda)\right), \tag{3.12}$$

using SGD and its mini batch variations. Before we analyse our objective function R_N with respect to Theorem 2.3.6, we want to understand the intuition behind the definition of this function better.

Remark 3.2.1 (Interpretation of R_N as a discrete expected value)**.** An intuitive way to interpret the objective function R_N is to regard it as a discretized expected value. Instead of minimizing the expected value of the original cost-function in Eq. (3.5), we discretize the space of uncertainty by picking normally distributed perturbation parameters $\{\lambda_i\}_{i=1}^N$. Hence, we switch from minimizing a continuous integral equation with respect to a probability measure, to minimizing a (discrete) sum with respect to a (discrete) probability measure.

This procedure is a practical example of the transition from the expected risk (cf. Eq. (1.2)) to the empirical risk (cf. Eq. (1.3)), as discussed in the introduction of this thesis.

Now we can prove that R_N satisfies all of our assumptions and calculate the parameters used in Theorem 2.3.6.

Lipschitz continuity of the gradient We differentiate the function R_N and use the linearity of the gradient operator, to obtain for all $u \in \mathbb{R}^n$

$$\nabla R_N(u) = \frac{1}{N}\sum_{i=1}^N \nabla_u F(u;\lambda_i) = \frac{1}{N}\sum_{i=1}^N A(\lambda_i)\left(A(\lambda_i)u - f\right).$$

Now the calculation of the Lipschitz constant is straight forward, since for $u, w \in \mathbb{R}^n$ it holds

$$\begin{aligned}
\|\nabla R_N(u) - \nabla R_N(w)\|_2 = \left\|\frac{1}{N}\sum_{i=1}^N A(\lambda_i)^2\,(u-w)\right\|_2 &\le \frac{1}{N}\sum_{i=1}^N \left\|A(\lambda_i)^2\,(u-w)\right\|_2 \\
&\le \frac{1}{N}\sum_{i=1}^N \|A(\lambda_i)\|_2^2\,\|u-w\|_2 \\
&\le \max_{i=1,\dots,n} \|A(\lambda_i)\|_2^2\,\|u-w\|_2 \\
&\le \max_{\lambda\in[\lambda_{\min},\lambda_{\max}]} \|A(\lambda)\|_2^2\,\|u-w\|_2
\end{aligned}$$

$$\leq L \left\| u - w \right\|_2 ,$$

where $L := (4 + \lambda_{\max} h^2)^2$. The last inequality follows since $A(\lambda_i)$ is symmetric and hence $\|A(\lambda_i)\|_2 = \mu_{\max}(A(\lambda_i))$ for all $i = 1, \ldots, N$. Using inequality (3.9) and the fact that $\lambda_i \leq \lambda_{\max}$ for all $i = 1, \ldots, N$, we get the desired inequality.

Unbiased estimate Because we are using normally distributed uncertainties with mean λ_{exact}, we have an unbiased estimate of the gradient. Therefore, invoking the section before Remark 2.3.5, we can assume $M_G = 1$ and $\mu = 1$.

Strongly convex Recalling the lecture *Optimization 1*, we know that a twice continuously differentiable mapping $g : \mathbb{R}^n \to \mathbb{R}$ is strongly convex if, and only if, the Hessian matrix $\nabla^2 g(x)$ is uniform positive definite with a constant ν for all $x \in \mathbb{R}^n$. We have also proven, that the modulus of the strong convexity equals $c = \frac{\nu}{2}$. By differentiating ∇R_N again, using Eq. (3.7) and the linearity of the differential operator, we get for $u \in \mathbb{R}^n$

$$\nabla^2 R_N(u) = \frac{1}{N} \sum_{i=1}^{N} \nabla_u^2 F(u; \lambda_i) = \frac{1}{N} \sum_{i=1}^{N} A(\lambda_i)^2.$$

Because the right-hand side does not depend on u, we just write $\nabla^2 R_N$ in the following. Now, we can estimate the minimal eigenvalue of $\nabla^2 R_N$ via

$$\begin{aligned}
\mu_{\min}\left(\nabla^2 R_N\right) &= \mu_{\min}\left(\frac{1}{N} \sum_{i=1}^{N} A(\lambda_i)^2\right) \geq \frac{1}{N} \sum_{i=1}^{N} \mu_{\min}(A(\lambda_i)^2) \\
&\geq \min_{i=1,\ldots,N} \mu_{\min}(A(\lambda_i)^2) \geq \min_{\lambda \in [\lambda_{\min}, \lambda_{\max}]} \mu_{\min}(A(\lambda)^2) \\
&\overset{(3.8)}{=} \mu_{\min}(A(\lambda_{\min})^2) \overset{(3.10)}{\geq} \left(h^2 \cdot \lambda_{\min}\right)^2 > 0.
\end{aligned}$$

The first inequality follows by applying Corollary A.1.2 to a sum of symmetric matrices and using a minimum estimate. Therefore, invoking Corollary A.1.3, $\nabla^2 R_N$ is uniform positive definite with the factor $\nu := \left(h^2 \lambda_{\min}\right)^2$. As a consequence, we obtain that F is strongly convex with modulus $c = \frac{\nu}{2} = \frac{h^4 \lambda_{\min}^2}{2}$. However, $c \searrow 0$ for $h \searrow 0$.

As mentioned in the introduction of this section, we want to analyse the performance of SGD and its mini batch variations. For this reason, we optimize Eq. (3.12) using the stochastic gradient descent and three mini batch versions, with a batch size of

10%, 25% and 50% of N. For the right-hand side of the differential equation, we use $f : [0,1] \to \mathbb{R}, x \mapsto \sin(\pi x)$ and register $f(0) = f(1) = 0$. Furthermore, we set $\lambda_{\text{exact}} = 2$ and $\lambda_{\max} = 3$, as well as $\lambda_{\min} = 1$. Our grid fineness will be fixed to $n = 100$. We set $\max_{\text{iter}} = 5000$ as the maximum number of iterations and $N = 200$, as the number of realizations of the uncertain parameter.

We use the same random vector $w_0 \in \mathbb{R}^n$ for all of the batch variations. As an initial step size, the upper bound provided by Theorem 2.3.6 is used, so the first step size is as big as possible.

In this benchmark test, we set the standard deviation to $\sigma = 1$. The results are shown in Fig. 3.5 and do not look really promising. First, we can see that the value of the loss function decreases in a very slow manner. In fact, the decrease per iteration is so slow that the maximum number of iteration had to be increased to $\max_{\text{iter}} = 10000$, to get a value of approximately 10^{-4} in the loss function.

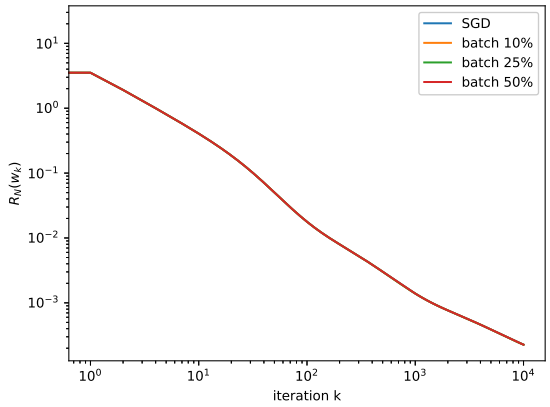


Figure 3.5.: Results of the first benchmark test with a standard deviation of $\sigma = 1$ and $\max_{\text{iter}} = 10000$. The visualization is a log-log plot.

All in all, we can see that all the mini batch variations behave in a similar way, compared to the SGD version. This is atypical, but can be explained by the following two reasons:

1. The gradient descent method itself is not well conditioned for a least-square-problem like (3.4). We discussed this fact in the lecture *Optimization 1* and used it to motivate the conjugated gradient (CG) method. Therefore, the stochastic version of gradient descent is also ill-conditioned for our problem, especially as we blew up the sum in Eq. (3.11).

2. The similar behaviour of all mini-batch variations compared to SGD itself can be explained in the same way. By checking the iterations of the code manually, one can see that the results of the batch variations only differ in a really small manner, as the decrease in the loss function is marginal per iteration.

A good aspect of this benchmark test is the sublinear decrease in the loss function, as we theoretically derived this convergence result. We can also see that the SGD algorithm is really robust against uncertainties, as we still get a pretty good precision, considering the amount of uncertainty (or noise) in the objective function.

As we want to solve the ODE (3.2), we are especially interested in solving the system (3.3) for small h and therefore in minimizing the function (3.12) for $h \to 0$. By analysing the matrix in Eq. (3.3) and the norm used in Eq. (3.11) more closely, two major problems arise:

1. For $h \to 0$ the eigenvalues of the matrix $A(\lambda)$ vanish, cf. Eq. (3.10) as the lower bound[3] converges to zero. Consequently, solving the system (3.3) is ill-conditioned for small values of h.

2. Secondly, the Euclidian norm used in Eq. (3.5) and (3.11) is not weighted and therefore not mesh size independent. To get a good convergence result, we should try to use a weighted norm.

In the following two sections, we want to address the first problem with a preconditioned system and the second one by substituting the Euclidean norm with a discretized L^2-norm. The goal is to combine both of these ideas with the robustness of the SGD algorithm under uncertainties, in order to properly solve the ODE for mesh sizes $h \to 0$.

3.2.1.1. Improvements using a preconditioner

As the optimization problem arose by transferring a system of linear equations into a least square problem, using a preconditioner is a natural way to stabilize the eigenvalues of the matrix $A(\lambda)$ and therefore improve the conditioning of the system in Eq. (3.3), respectively Eq. (3.12).

[3]Indeed, for small values of h (respectively large values of n), the lower bound is approached by some of the eigenvalues. We can see this, if we consider μ_n in equation (3.10). For large n we have $\frac{n}{n+1} \approx 1$ and hence $\mu_n \approx \lambda h^2$. Thus, for small values of h the lower bound is a good indicator for vanishing eigenvalues.

To motivate the term *preconditioner*, we take a look at a system of linear equations of the form

$$Bx = y,$$

for a regular matrix $B \in \mathbb{R}^{m\times m}$ and a vector $y \in \mathbb{R}^m$, $m \in \mathbb{N}$. Furthermore, we assume having a so-called *splitting* of the matrix B into $B = M - N$, where M is invertible. This splitting induces a *stationary iteration* of the form

$$Mx_{k+1} = Nx_k + y, \quad k = 0, 1, 2, \ldots$$

with an initial guess $x_0 \in \mathbb{R}^m$ and iterates $\{x_k\}_{k\in\mathbb{N}_0} \subset \mathbb{R}^m$. If we rewrite the stationary iteration process, we obtain for $k = 0, 1, 2, \ldots$ the following equivalent forms

$$\begin{aligned} Mx_{k+1} = Nx_k + y &\iff x_{k+1} = M^{-1}Nx_k + M^{-1}y \\ &\iff x_{k+1} = (I_m - M^{-1}B)x_k + M^{-1}y. \end{aligned}$$

At convergence $x_k \overset{k\to\infty}{\longrightarrow} x$ of the stationary iterative method, we obtain the system

$$x = (I_m - M^{-1}B)x + M^{-1}y \iff M^{-1}Bx = M^{-1}y,$$

the so-called *preconditioned (or transformed) system*[4].

If we consider our system matrix $A(\lambda)$ again, we observe that for every $\lambda \in \mathbb{R}^+$ the matrix $A(\lambda)$ is not only uniform positive definite, but also strictly diagonally dominant. Hence, the splitting $A(\lambda) = M(\lambda) - N(\lambda)$, with $M(\lambda) = \operatorname{diag}(A(\lambda))$ and $N(\lambda) = \operatorname{diag}(A(\lambda)) - A(\lambda)$ leads to a convergent stationary method[5] (cf. Corollary A.1.7) and is therefore a good choice for preconditioning.

Notice that the matrix $N(\lambda)$ consists of the upper and lower diagonal and is consequently independent of the parameter λ. Another really important advantage of this splitting is the structure of the matrix $M(\lambda)$, because the matrix is not only symmetric and positive definite but also easy to invert. As $M(\lambda)$ is a diagonal matrix[6] with the same value in each diagonal entry, a matrix multiplication with the inverse

[4] Since we multiply the system with M^{-1} from the left-hand side, this is a so-called *left preconditioned system*. One can also easily generalize the preconditioned system to a *right preconditioned system*, cf. [CG20, Sec. 4.2].

[5] This is the so-called *Jacobi method*.

[6] Thus, the matrices $M(\lambda)$ and $M(\lambda)^{-1}$ also commutate with all other occurring matrices.

matrix $M(\lambda)^{-1}$ is equivalent to a scalar multiplication with the factor $\frac{1}{2+\lambda h^2}$. From a numerical perspective, this is a major improvement.

For $\lambda \in \mathbb{R}^+$, we can now define the preconditioned system

$$M(\lambda)^{-1}A(\lambda)u = M(\lambda)^{-1}f \tag{3.13}$$

and consequently, the preconditioned optimization problem is of the form

$$\min_{u\in\mathbb{R}^n} F^{\mathsf{PC}}(u;\lambda), \tag{3.14}$$

$$\text{with } F^{\mathsf{PC}}(\cdot;\lambda) : \mathbb{R}^n \to \mathbb{R}, u \mapsto \frac{1}{2}\left\|M(\lambda)^{-1}\left(A(\lambda)u - f\right)\right\|_2^2. \tag{3.15}$$

Similar to Section 3.2.1, we can differentiate the objective function (twice) with respect to u and obtain

$$\nabla_u F^{\mathsf{PC}}(u;\lambda) = A(\lambda)M(\lambda)^{-\mathsf{T}}M(\lambda)^{-1}(A(\lambda)u - f) \tag{3.16}$$

$$\overset{M^{-1}\text{ symm.}}{=} A(\lambda)\left(M(\lambda)^{-1}\right)^2\left(A(\lambda)u - f\right) \tag{3.17}$$

$$\overset{M^{-1}\text{ comm.}}{=} \left(M(\lambda)^{-1}\right)^2 A(\lambda)(A(\lambda)u - f), \tag{3.18}$$

$$\nabla_u^2 F^{\mathsf{PC}}(u;\lambda) = \left(M(\lambda)^{-1}\right)^2 A(\lambda)^2. \tag{3.19}$$

We can now compute the eigenvalues of the matrix product

$$\mu_k\left(M(\lambda)^{-1}A(\lambda)\right) = \frac{1}{2+\lambda h^2}\cdot\mu_k(A(\lambda)), \quad k = 1,\dots,n \tag{3.20}$$

and, using the inequalities (3.9) and (3.10), we obtain the following upper and lower bounds

$$\mu_k\left(M(\lambda)^{-1}A(\lambda)\right) \leq \frac{4+\lambda h^2}{2+\lambda h^2} \tag{3.21}$$

$$\mu_k\left(M(\lambda)^{-1}A(\lambda)\right) \geq \frac{\lambda h^2}{2+\lambda h^2}, \tag{3.22}$$

for $k = 1,\dots,n$. Compared to the previous lower bound in Eq. (3.10) of the eigenvalues, the lower bound in Eq. (3.22) of the preconditioned system decays with a factor of approximately $\frac{1}{2}$ slower than the initial lower bound as $h \to 0$. This is not a major improvement, but we will see the strengths of the preconditioned system, if we analyse the upper bound for the step size sequence of SGD.

Assuming the same uncertainties in the calculation of the parameter λ as in Section 3.2.1, we can define the preconditioned objective function

$$\begin{aligned} &R_N^{\mathsf{PC}}\left(F^{\mathsf{PC}}(\cdot;\lambda)\right) : \mathbb{R}^n \to \mathbb{R} \\ &\quad u \mapsto \frac{1}{N}\sum_{i=1}^{N} F^{\mathsf{PC}}(u;\lambda_i) = \frac{1}{2N}\sum_{i=1}^{N} \left\| M(\lambda_i)^{-1}\left(A(\lambda_i)u - f\right)\right\|_2^2, \end{aligned} \tag{3.23}$$

again for normally distributed values $\{\lambda_i\}_{i=1}^N \subset [\lambda_{\min}, \lambda_{\max}]$ and $N \in \mathbb{N}$. Our goal is now to solve the preconditioned minimization problem

$$\min_{u\in\mathbb{R}^n} R_N^{\mathsf{PC}}\left(F(u;\lambda)\right). \tag{3.24}$$

We need to verify that our preconditioned objective function R_N^{PC} still satisfies all assumptions of Theorem 2.3.6. As all the calculations are similar to the calculations for the original system, we present the results in a shortened manner.

Lipschitz continuity of the gradient The derivative of the function R_N^{PC} is a direct consequence of the linearity of the gradient operator and our first calculation, thus

$$\nabla R_N^{\mathsf{PC}}(u) = \frac{1}{N}\sum_{i=1}^{N} \nabla_u F^{\mathsf{PC}}(u;\lambda_i) = \frac{1}{N}\sum_{i=1}^{N} \left(M(\lambda)^{-1}\right)^2 A(\lambda)(A(\lambda)u - f).$$

Hence, using Eq. (3.20) and basic properties of the spectral norm, for $u, w \in \mathbb{R}^n$ we have

$$\begin{aligned} \left\|\nabla R_N^{\mathsf{PC}}(u) - \nabla R_N^{\mathsf{PC}}(w)\right\|_2 &= \left\| \frac{1}{N}\sum_{i=1}^{N} \left(M(\lambda_i)^{-1}\right)^2 A(\lambda_i)^2 (u-w)\right\|_2 \\ &\le \frac{1}{N}\sum_{i=1}^{N} \|A(\lambda_i)\|_2^2 \left\|M(\lambda_i)^{-1}\right\|_2^2 \|u-w\|_2 \\ &\le \left(\max_{\lambda\in[\lambda_{\min},\lambda_{\max}]} \|A(\lambda)\|_2^2 \left\|M(\lambda)^{-1}\right\|_2^2\right) \|u-w\|_2 \\ &\le L^{\mathsf{PC}} \|u-w\|_2, \end{aligned}$$

with a Lipschitz constant $L^{\mathsf{PC}} := \frac{\left(4+\lambda_{\max}h^2\right)^2}{\left(2+\lambda_{\min}h^2\right)^2}$.

Unbiased estimate Because we are still using normally distributed uncertainties with mean λ_{exact}, we have again an unbiased estimate of the gradient. Therefore

$M_G = 1$ and $\mu = 1$ stay the same.

Strongly convex We again calculate a lower bound for the eigenvalues of the Hessian matrix $\nabla^2 R_N^{\mathsf{PC}}(u) = \nabla^2 R_N^{\mathsf{PC}}$ (the Hessian matrix does again not depend on u). Since the following inequalities for the minimal eigenvalue hold

$$\begin{aligned}\mu_{\min}\left(\nabla^2 R_N^{\mathsf{PC}}\right) &= \mu_{\min}\left(\frac{1}{N}\sum_{i=1}^{N}\left(M(\lambda_i)^{-1}\right)^2 A(\lambda_i)^2\right)\\ &\geq \min_{\lambda\in[\lambda_{\min},\lambda_{\max}]} \mu_{\min}\left(\left(M(\lambda_i)^{-1}\right)^2 A(\lambda_i)^2\right)\\ &\geq \left(\frac{\lambda_{\min}h^2}{2+\lambda_{\max}h^2}\right)^2.\end{aligned}$$

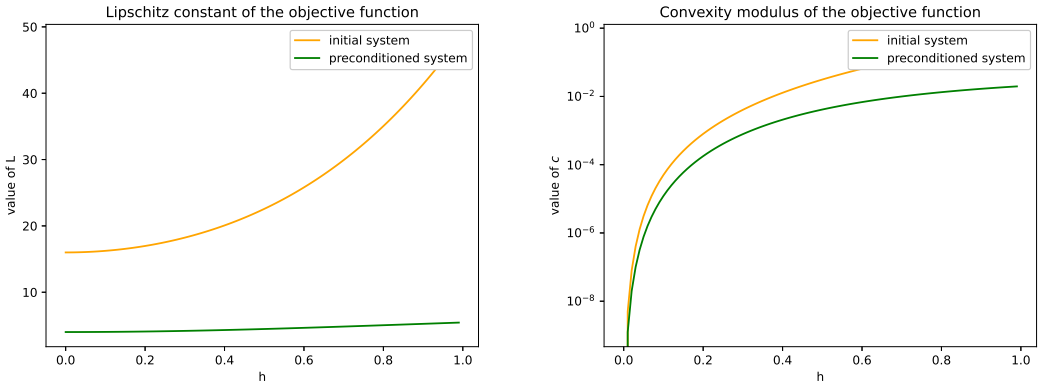
We obtain $c^{\mathsf{PC}} = \frac{(\lambda_{\min}h^2)^2}{2\cdot(2+\lambda_{\max}h^2)^2}$ as the modulus for the preconditioned optimization problem.

After we have shown that the preconditioned system still satisfies all important properties of Theorem 2.3.6, we can compare the Lipschitz constant and the convexity modulus with respect to their behaviour for $h \to 0$. The results are shown in Fig. 3.6. We can clearly see that the Lipschitz constant of the preconditioned objective function is smaller (for $h \to 0$ by a factor of approximately $\frac{1}{4}$). Furthermore, we observe that also the convexity modulus of the preconditioned system is always smaller than the modulus of the initial system, although they both converge to zero as $h \to 0$.

This is a good result, since both of these terms reciprocally occur in Theorem 2.3.6. Consequently, we should expect being able to run the SGD algorithm for the preconditioned system with a bigger initial step size and hence get faster convergence results for small values of the mesh size h.

The behaviour of the upper bound for the initial step size (cf. Theorem 2.3.6) is shown in Fig. 3.7a, the ratio between both of the upper bounds in Fig. 3.7b. Here we can clearly see, why the stabilization of the eigenvalues is worth it. Not only is the upper bound for the maximum initial step size of the preconditioned system always higher than the upper bound for the initial system, but they differ for $h \to 0$ by a factor of four. This is exactly the squared reciprocal of the value $\frac{1}{2}$ – the ratio by which the lower bound of the preconditioned system decayed slower for $h \to 0$.

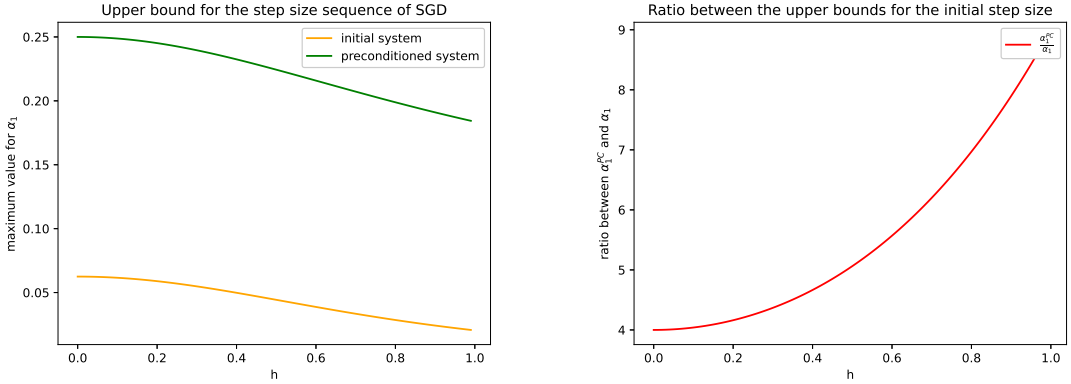
Reconsidering the intuition behind Remark 2.3.5, an increased initial step size by a factor of four is a huge improvement, especially if we are interested in running SGD for high-precision results and hence for numerous iterations.



(a) Lipschitz constant

(b) Convexity modulus

Figure 3.6.: Comparison between the initial and the preconditioned system regarding the Lipschitz constant and the convexity modulus.



(a) Comparison between the upper step size bound of the initial system and the preconditioned system (cf. Theorem 2.3.6) for different values of h.

(b) Ratio between the maximum initial step size α_1^{PC} for the preconditioned system and the maximum initial step size α_1 for the initial system.

Figure 3.7.: Comparison of the maximum value for the first step size and the ratio between the first step sizes of the preconditioned and the initial optimization problem.

We want to verify our theoretical results with a second benchmark test for different values of h as $h \to 0$. For the sake of comparability, we keep the maximum number of iterations with $\text{iter}_{\max} = 2000$ fixed, as well as the standard deviation $\sigma = 1$ and the upper summation index $N = 50$ for the number of realizations of the uncertainty parameter λ. The batch size is for both of the systems fixed with value of $\text{batch}_{\text{size}} = 25$, so 50% of the summands are represented in a batch.

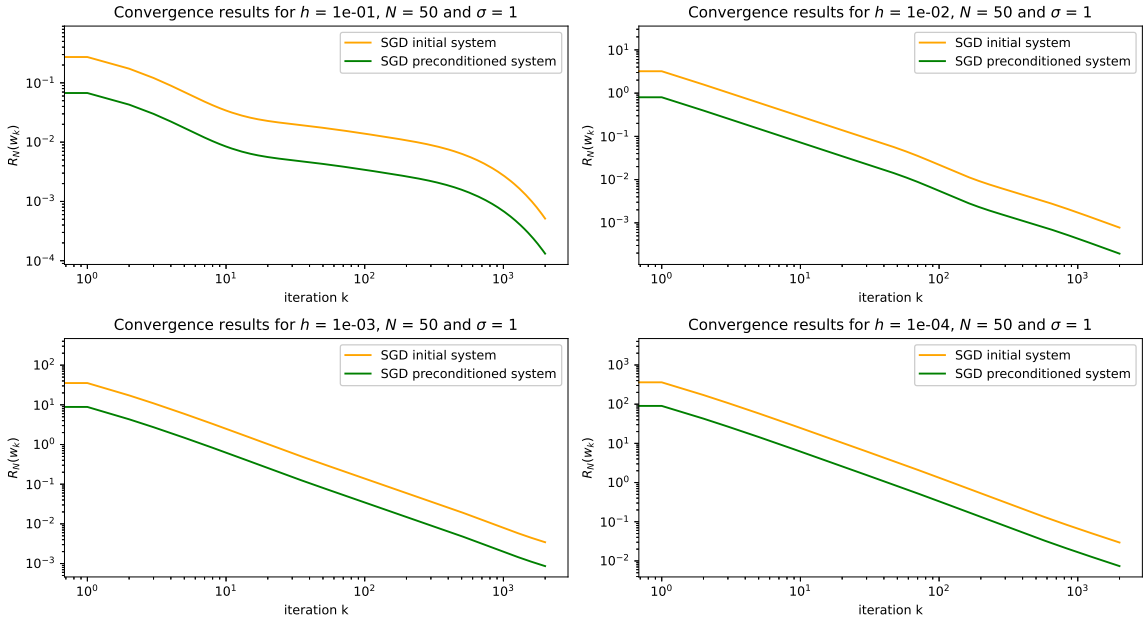


Figure 3.8.: Convergence results for different mesh sizes h as $h \to 0$. The visualization is a log-log-plot. The batch size is for both systems set to 25 and therefore represents half of the sum in each batch.

The results are shown in Fig. 3.8 and Table 3.1 shows the needed CPU time, as well as the value of the loss function after the last iteration. The graphs show in a clear manner, what we expected. Due to the higher initial step size, the SGD algorithm converges faster for the preconditioned system and also yields a better result in all the four cases. The precision of SGD still degenerates for both systems as $h \to 0$, since we are not using a weighted norm yet. This will be the next step in our analysis.

3.2.1.2. Improvements using a weighted norm

To improve the unstable convergence behaviour of both systems in the case of $h \to 0$, we want to derive a weighted norm by discretizing the L^2–norm.

From a more functional analytic perspective, solving the ODE (3.2) is equivalent to minimizing the squared L^2–norm

$$\int_0^1 |-u''(x) + \lambda u(x) - f(x)|^2 \, \mathrm{d}x \tag{3.25}$$

Mesh size	System	Loss	CPU time [s]
$h = 1.0 \cdot 10^{-01}$	Initial	$5.1388 \cdot 10^{-04}$	$\mathbf{8.9444 \cdot 10^{-01}}$
	Precond.	$\mathbf{1.3173 \cdot 10^{-04}}$	$1.0511 \cdot 10^{+00}$
$h = 1.0 \cdot 10^{-02}$	Initial	$7.7720 \cdot 10^{-04}$	$1.7214 \cdot 10^{+00}$
	Precond.	$1.9428 \cdot 10^{-04}$	$1.8633 \cdot 10^{+00}$
$h = 1.0 \cdot 10^{-03}$	Initial	$3.4479 \cdot 10^{-03}$	$1.4313 \cdot 10^{+01}$
	Precond.	$8.6198 \cdot 10^{-04}$	$1.4686 \cdot 10^{+01}$
$h = 1.0 \cdot 10^{-04}$	Initial	$2.9516 \cdot 10^{-02}$	$1.4499 \cdot 10^{+02}$
	Precond.	$7.3791 \cdot 10^{-03}$	$1.5318 \cdot 10^{+02}$

Table 3.1.: CPU time and loss after the last iteration for the initial and preconditioned (Precond. in the table) system.

over the set $u \in \{g \in C^2(\overline{\Omega}) \ : \ g|_{\partial\Omega} = 0\}$.[7]

By discretizing the ODE using finite differences, we again obtain the linear system $A(\lambda)u = f$ for vectorized versions u and f, of the unknown u and the right-hand side f evaluated on the discrete mesh (for detailed steps, see Section 3.2.1). Now we can approximate the integral (3.25) using the trapezoidal rule [DR08, p. 349] and obtain

$$\begin{aligned}\int_0^1 \left|-u''(x) + \lambda u(x) - f(x)\right|^2 \, \mathrm{d}x &\approx \sum_{i=1}^{n} h \cdot (A(\lambda)u - f)_i \\ &= h \cdot \|A(\lambda)u - f\|_2^2 = \|A(\lambda)u - f\|_{2,h}^2 \,,\end{aligned}$$

where $\|v\|_{2,h} := \|v\|_D := \sqrt{v^\mathsf{T} D v} = \sqrt{h} \cdot \|v\|_2$ for a vector $v \in \mathbb{R}^n$ and $D := h \cdot \mathrm{I}_n \in \mathbb{R}^{n \times n}$. This is the so-called *weighted residual*. Consequently, the weighted objective function is defined as

$$F_\mathsf{h}(\cdot;\lambda) : \mathbb{R}^n \to \mathbb{R}, u \mapsto \|A(\lambda)u - f\|_{2,h}^2 = h \cdot \|A(\lambda)u - f\|_2^2 \,. \tag{3.26}$$

By using normally distributed realizations $\{\lambda_i\}_{i=1}^N \subset [\lambda_{\min}, \lambda_{\max}]$ of the parameter λ,

[7] To be here formally correct, one would need $u \in H^2(\Omega) \cap H_0^1(\Omega)$ with the Sobolev spaces $H^2(\Omega)$ and $H_0^1(\Omega)$ (cf. Section 3.2.2). Since we still aim for a strong solution, we assume $u \in C^2(\overline{\Omega})$ with $u|_{\partial\Omega} = 0$ for simplicity. Hence, we can discretize the integral and obtain a finite dimensional optimization problem. One could also consider the integral (3.25) over the Banach space $C_0(\overline{\Omega}) \cap C^2(\Omega)$. This is more complicated, since we do not have an inner product anymore.

we get

$$R_{N,\mathsf{h}}(F_{\mathsf{h}}(\cdot;\lambda)) : \mathbb{R}^n \to \mathbb{R}, u \mapsto \frac{1}{N}\sum_{i=1}^{N} F_{\mathsf{h}}(u;\lambda_i) = \frac{h}{2N}\sum_{i=1}^{N} \|A(\lambda_i)u - f\|_2^2\,, \tag{3.27}$$

for the corresponding objective function under uncertainty. By analogously calculating the gradient and using the bounds (3.9) and (3.10) for the eigenvalues of the matrix, we observe the constants

$$L_{\mathsf{h}} = h \cdot L = h \cdot (4 + \lambda_{\max} h^2)^2$$
$$c_{\mathsf{h}} = h \cdot c = h \cdot \frac{h^4 \lambda_{\min}^2}{2}$$

for the weighted system. Since the matrix $A(\lambda)$ is unchanged and therefore still strictly diagonally dominant, we will also use the Jacobi method as a preconditioner for the weighted system. According to the definition of the initial preconditioned objective function in Eq. (3.24), we can directly derive the preconditioned weighted objective function

$$R_{N,\mathsf{h}}^{\mathsf{PC}}(F_{\mathsf{h}}(\cdot;\lambda)) : \mathbb{R}^n \to \mathbb{R}, u \mapsto \frac{h}{2N}\sum_{i=1}^{N} \left\|M(\lambda_i)^{-1}\left(A(\lambda_i)u - f\right)\right\|_2^2. \tag{3.28}$$

In a similar calculation, we obtain the Lipschitz constant and the convexity modulus for the preconditioned weighted system, by multiplying the constants of the initial preconditioned system with a factor of h. As a result, we have

$$L_{\mathsf{h}}^{\mathsf{PC}} = h \cdot L^{\mathsf{PC}} = h \cdot \frac{(4 + \lambda_{\max} h^2)^2}{(2 + \lambda_{\min} h^2)^2}$$
$$c_{\mathsf{h}}^{\mathsf{PC}} = h \cdot c^{\mathsf{PC}} = h \cdot \frac{(\lambda_{\min} h^2)^2}{2 \cdot (2 + \lambda_{\max} h^2)^2}.$$

Our goal is now to compare the behaviour of the two minimization problems

$$\min_{u \in \mathbb{R}^n} R_{N,\mathsf{h}}(u) \text{ and } \min_{u \in \mathbb{R}^n} R_{N,\mathsf{h}}^{\mathsf{PC}}(u), \tag{3.29}$$

with respect to decreasing mesh sizes $h \to 0$.

As before, we set the maximum number of iterations fixed to $\text{iter}_{\max} = 2000$, the standard deviation to $\sigma = 1$ and the number of realizations of the uncertainty parameter λ to $N = 50$. The batch size is, for the sake of comparability, set to 50% of the

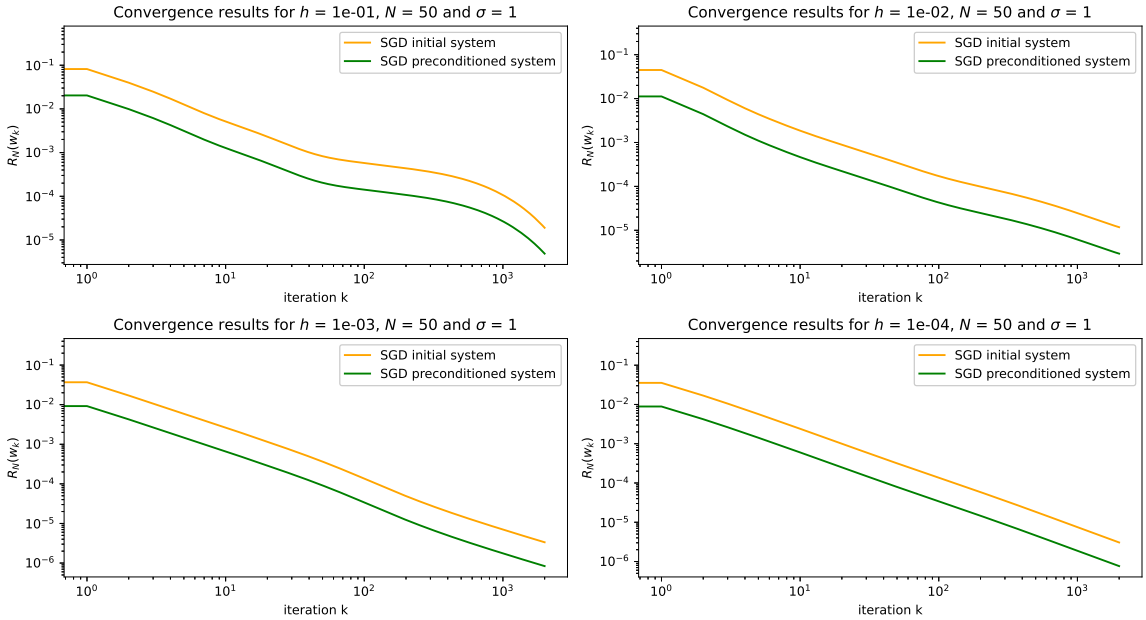


Figure 3.9.: Convergence results of the weighted systems for different mesh sizes h as $h \to 0$. The visualization is a log-log-plot. The batch size is for both systems set to 25 and therefore represents half of the sum in each batch.

summands, meaning $\text{batch}_{\text{size}} = 25$. The results are shown in Fig. 3.9 and additionally Table 3.2 shows the loss of the objective function evaluated in the last iterate, as well as the needed computational time for each of the two optimizations problems.

In contrast to the convergence plots in Fig. 3.8, the results of the plots in Fig. 3.9 really improved and show, what we expected. In all the cases, the loss decreases in a very stable manner, especially as $h \to 0$. We do not observe a degeneration of the loss for very small mesh sizes anymore and Table 3.2 clearly shows that we can also expect a high-precision solution, even for very small values of h. Similar to Fig. 3.8, the preconditioned system performs better in all the four scenarios, due to a higher initial step size.

If we compare the results of this last benchmark test to our first test (cf. Fig. 3.5), we can observe major improvements. Applying the SGD algorithm in combination with a preconditioner to the weighted objective function, provides a really stable and convincing method to obtain a strong solution for the parametrized ODE (3.2) under uncertainties.

In the next chapter, we want to expand this approach to aim for weak solutions of a parametrized partial differential equation (PDE) under uncertainties.

Mesh size	System	Loss	CPU time [s]
$h = 1.0 \cdot 10^{-01}$	Initial	$1.9023 \cdot 10^{-05}$	$\mathbf{1.0555 \cdot 10^{+00}}$
	Precond.	$4.8708 \cdot 10^{-06}$	$1.1859 \cdot 10^{+00}$
$h = 1.0 \cdot 10^{-02}$	Initial	$1.1756 \cdot 10^{-05}$	$1.8091 \cdot 10^{+00}$
	Precond.	$2.9387 \cdot 10^{-06}$	$1.9382 \cdot 10^{+00}$
$h = 1.0 \cdot 10^{-03}$	Initial	$3.3485 \cdot 10^{-06}$	$1.5797 \cdot 10^{+01}$
	Precond.	$8.3712 \cdot 10^{-07}$	$1.6006 \cdot 10^{+01}$
$h = 1.0 \cdot 10^{-04}$	Initial	$3.0561 \cdot 10^{-06}$	$1.4185 \cdot 10^{+02}$
	Precond.	$\mathbf{7.6403 \cdot 10^{-07}}$	$1.6685 \cdot 10^{+02}$

Table 3.2.: CPU time and loss after the last iteration for the weighted initial and preconditioned (Precond. in the table) weighted system.

3.2.2. PDE Dirichlet problem

In this section, we assume having a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, with a parameter domain Ω. Now we consider the following parametrized elliptic partial differential equation (PDE) for a domain $G = (0,1)^2$ and Dirichlet boundary conditions

$$\left.\begin{aligned} -\operatorname{div}(a(x;\omega)\nabla y(x)) = f(x),\ x \in G,\ \text{a.s. } \omega \in \Omega, \\ y(x) = 0,\ x \in \partial G \end{aligned}\right\}, \tag{3.30}$$

for $f \in L^2(G)$ and a function a satisfying

$$a_{\min} < a(x;\omega) < a_{\max} \quad \forall x \in G,\ \text{a.s. } \omega \in \Omega, \tag{3.31}$$

with $0 < a_{\min} < a_{\max} < \infty$. We want to solve Eq. (3.30) for $\omega \in \Omega$ almost surely (a.s.). In contrast to Section 3.2.1, we now aim for a weak solution and interpret problem (3.30) not in the strong form. We define the Sobolev space $V := H_0^1(G) = \overline{C_0^\infty(G)}^{\|\cdot\|_{H^1}}$ (see [Den21, Sec. 6 a)]) with its dual space $H^{-1}(G) := V'$. We equip the space V with the inner product $\langle \cdot, \cdot \rangle_V := \langle \nabla \cdot, \nabla \cdot \rangle_{L^2}$. Instead of considering (3.30) directly, we regard the following problem: Find $y \in V$, which solves

$$-\operatorname{div}\left(a(\cdot;\omega)\nabla y\right) = f \in V' \text{ a.s. } \omega \in \Omega. \tag{3.32}$$

Using the Gauss divergence theorem [Den21, p. 50], we can interpret Eq. (3.32) as

$$\int_G a(\cdot;\omega)\nabla y \cdot \nabla \varphi \,\mathrm{d}x = \langle f, \varphi \rangle_{V' \times V} = \int_G f\varphi \,\mathrm{d}x,\ \forall \varphi \in V \text{ a.s. } \omega \in \Omega, \tag{3.33}$$

with the canonical dual pairing $V' \times V$. The next Lemma shows the existence of a (unique) weak solution for almost every (fixed) $\omega \in \Omega$.

Lemma 3.2.2 (Existence of a weak solution for almost every $\omega \in \Omega$)**.** *For $f \in L^2(G)$ and almost every $\omega \in \Omega$ Eq.* (3.33) *has a (unique) weak solution $y = y(\cdot;\omega) \in V$.*

Proof. Let $\omega \in \Omega$ be fixed and $f \in L^2(G)$. We want to apply the lemma of Lax-Milgram (Lemma A.3.2) and define the functions

$$\begin{aligned} b_\omega : V \times V \to \mathbb{R}, (v,w) &\mapsto \langle a(\cdot;\omega)\nabla v, \nabla u\rangle_{L^2(G)} \\ \text{and } F_f : V \to \mathbb{R}, v &\mapsto \langle v, f\rangle_{L^2(G)}. \end{aligned}$$

We have to show that the assumptions

(i) F_f is linear and continuous, so $F_f \in V'$ holds and

(ii) b_ω is a continuous and coercive bilinear map

of the Lax-Milgram lemma are satisfied. The inner product in $L^2(G)$ is linear in the first argument, so F_f is linear. For $v \in V$ it holds

$$|F_f(v)| = \left|\langle v, f\rangle_{L^2(G)}\right| \leq \|f\|_{L^2(G)} \|v\|_{H_0^1(G)},$$

using the Cauchy–Schwarz inequality and the trivial inequality $\|v\|_{L^2(G)} \leq \|v\|_{H_0^1(G)}$. Hence, F_f is continuous and (i) holds. The bilinearity of b_ω is clear, since the inner product in $L^2(G)$ is linear in both arguments. Next, we show that b_ω is continuous. Let $v, w \in V$, using the Cauchy–Schwarz inequality yields

$$|b_\omega(v,w)| = \left|\langle a(\cdot;\omega)\nabla v, \nabla w\rangle_{L^2(G)}\right| \leq a_{\max} \|v\|_{H_0^1(G)} \|w\|_{H_0^1(G)}.$$

We use Friedrich's inequality (Lemma A.3.3) and for $u \in V$ obtain

$$\begin{aligned} b_\omega(u,u) = \langle a(\cdot;\omega)\nabla u, \nabla u\rangle_{L^2(G)} &\geq \frac{a_{\min}}{2}\left(\langle \nabla u, \nabla u\rangle_{L^2(G)} + \mathrm{d}(G)\langle u, u\rangle_{L^2(G)}\right) \\ &\geq \frac{a_{\min}}{2} \min\left\{1, \mathrm{d}(G)\right\} \|u\|^2_{H_0^1(G)}, \end{aligned}$$

where $\mathrm{d}(G) = \sup_{x,y\in G} \|x - y\|_{\mathbb{R}^2}$ is the *diameter* of the domain G. In our case, where $G = (0,1)^2$ is the unit square, we have $\mathrm{d}(G) = \sqrt{2}$. The lemma of Lax-Milgram yields the desired unique weak solution for Eq. (3.33). □

We gain a solution for Eq. (3.32) and therefore also for Eq. (3.33), by solving the minimization problem

$$\min_{y \in V} \| - \operatorname{div}(a(\cdot;\omega)\nabla y) - f \|_{V'}, \tag{3.34}$$

a.s. for $\omega \in \Omega$. As before, we switch from minimizing over the whole parameter domain, to minimizing a discrete expected value. Hence, we choose $n \in \mathbb{N}$ discrete, distinct samples $\{\omega_i\}_{i=1}^n$ and aim for a (deterministic) function $y = y(x) \in V$, which minimizes the discrete expected value

$$R_n : V \to \mathbb{R}, y \mapsto \frac{1}{2n} \sum_{i=1}^{n} \| - \operatorname{div}(a(\cdot;\omega_i)\nabla y) - f \|_{V'}. \tag{3.35}$$

3.2.2.1. Discretization using Finite Elements and numerical results

In the next step, we want to discretize the expected value in Eq. (3.35). Hence, we choose a Finite Element (FE) discretization[8]. We assume having a finite-dimensional subspace $V_N \subset V$ spanned by $V_N = \operatorname{span}\{\varphi_1, \ldots, \varphi_N\}$ for $N \in \mathbb{N}$ linearly independent functions $\varphi_1, \ldots, \varphi_N \in V$, and equip the space V_N with the topology induced by V. In both spaces, we consider inner product $\langle \cdot, \cdot \rangle_V := \langle \nabla \cdot, \nabla \cdot \rangle_{L^2}$. For $\omega \in \Omega$ we define the system matrix via

$$A(\omega) := \left(\left(\int_G a(\cdot;\omega) \nabla \varphi_j \cdot \nabla \varphi_i \, \mathrm{d}x \right) \right)_{1 \le i,j \le N} \in \mathbb{R}^{N \times N} \tag{3.36}$$

and the vector corresponding to the right-hand side as

$$F := \left(\int_G f \varphi_i \, \mathrm{d}x \right)_{1 \le i \le N} \in \mathbb{R}^N. \tag{3.37}$$

For a fixed $\omega \in \Omega$ Eq. (3.34) is now discretized as

$$A(\omega) y = F,$$

with $y = (y_i)_{1 \le i \le N} \in \mathbb{R}^N$. The function $y_h := \sum_{i=1}^N y_i \varphi_i$ solves the variational problem

$$\int_G a(\cdot;\omega) \nabla y_h \cdot \nabla \varphi_h \, \mathrm{d}x = \langle f, \varphi_h \rangle_{V' \times V} = \int_G f \varphi_h \, \mathrm{d}x, \ \forall \varphi_h \in V_N.$$

[8] We use the *FEniCS* library [Aln+15] in our program for Finite Element discretizations.

Next, we have to realize an evaluation of the norm in the dual space V'. For this purpose, we consider the corresponding Riesz-representative, provided by the lemma of Fréchet-Riesz (Lemma A.3.1). Let us define the *stiffness matrix*

$$S = ((\langle \varphi_j, \varphi_i \rangle_V))_{1 \leq i,j \leq N} = \left(\left(\int_G \nabla \varphi_j \cdot \nabla \varphi_i \, \mathrm{d}x \right) \right)_{1 \leq i,j \leq N}.$$

For a fixed $\omega \in \Omega$ we consider the (loss) functional[9] $\Psi_h = \Psi_h(\omega) \in V_N'$ given by

$$\Psi_h : V_N \to \mathbb{R}, \varphi_h \mapsto \langle \Psi_h, \varphi_h \rangle_{V_N' \times V_N} := \int_G a(\cdot;\omega) \nabla y_h \cdot \nabla \varphi_h \, \mathrm{d}x - \langle f, \varphi_h \rangle_{V' \times V}.$$

Using the lemma of Fréchet-Riesz, we obtain the Riesz-representative $\psi_h \in V_N$ using the equivalent forms

$$\begin{aligned} &\langle \psi_h, \varphi_h \rangle_V = \langle \Psi_h, \varphi_h \rangle_{V_N' \times V_N} \ \forall \varphi_h \in V_N \\ \iff \quad &S\psi = \Psi \in \mathbb{R}^N, \end{aligned}$$

with vectors

$$\begin{aligned} \psi &= (\psi_1, \ldots, \psi_N)^\mathsf{T} \in \mathbb{R}^N \\ \Psi &= \left(\langle \Psi_h, \varphi_1 \rangle_{V_N' \times V_N}, \ldots, \langle \Psi_h, \varphi_N \rangle_{V_N' \times V_N} \right)^\mathsf{T} \in \mathbb{R}^N. \end{aligned}$$

We obtain the desired representative $\psi_h = \sum_{i=1}^N \psi_i \varphi_i \in V_N$ as a linear combination of the basis vectors. Note, that

$$\begin{aligned} \Psi = \Psi(\omega) &= A(\omega) y - F \\ \implies \psi &= S^{-1} \left(A(\omega) y - F \right) \end{aligned} \tag{3.38}$$

holds. Notice that for the corresponding values of the norms, we obtain

$$\|\psi_h\|_{V_N}^2 = \|\psi\|_S^2 = \left\| S^{-1} \Psi \right\|_S^2 = \|\Psi\|_{S^{-1}}^2 = \|\Psi_h\|_{V_N'}^2.$$

[9] We will later use this result for every summand in Eq. (3.35).

Finally, we can determine a realization of the norm in the dual space V_N'. We gain for $y \in V_N$ the simplification

$$\begin{aligned} R_n(y) &:= \frac{1}{2n} \sum_{i=1}^{n} \|\Psi_h(\omega_i)\|_{V_N'}^2 \\ &\overset{\text{Fréchet-Riesz}}{=} \frac{1}{2n} \sum_{i=1}^{n} \|\psi_h(\omega_i)\|_{V_N}^2 \\ &\overset{(*)}{=} \frac{1}{2n} \sum_{i=1}^{n} \psi(\omega_i)^\mathsf{T} S \psi(\omega_i) \\ &\overset{(3.38)}{=} \frac{1}{2n} \sum_{i=1}^{n} \psi(\omega_i)^\mathsf{T} S S^{-1} \left(A(\omega_i) y - F\right) \\ &= \frac{1}{2n} \sum_{i=1}^{n} \psi(\omega_i)^\mathsf{T} \left(A(\omega_i) y - F\right) \\ &\overset{(3.38)}{=} \frac{1}{2n} \sum_{i=1}^{n} \|A(\omega_i) y - F\|_{S^{-1}}^2 . \end{aligned}$$

The equality $(*)$ holds since for $v_h, w_h \in V_N$ with $v_h = \sum_{i=1}^N \varphi_i v_i$ and $w_h = \sum_{i=1}^N \varphi_i w_i$ we have

$$\begin{aligned} \langle v_h, w_h \rangle_{V_N} &= v^\mathsf{T} S w \\ \Longrightarrow \|v_h\|_{V_N} &= \|v\|_S , \end{aligned}$$

with vectors $v = (v_i)_{1 \le i \le N}, w = (w_i)_{1 \le i \le N} \in \mathbb{R}^N$.

Since we only deal with a fixed and finite number of realizations of the parameter $\omega \in \Omega$, it is from a numerical perspective really useful to assemble the matrices $A(\omega_i)$ and $S^{-1}A(\omega_i)$ (here one should solve $SB = A(\omega_i)$) for $i = 1, \dots, N$ once and save them for further needs. If one needs a lot of calls with different values of ω, e.g. in a PDE constrained inverse problem under uncertainties, one can decompose the matrices $A(\omega_i)$ using the Discrete Empirical Interpolation Method (DEIM), as discussed in Remark 3.2.3.

Similarly, we can compute the gradient of the objective function. Instead of considering V_N, we equip $\mathbb{R}^N$ with the inner product $\langle \cdot, \cdot \rangle_S$ and compute the Gradient with respect to this inner product. Hence, we later have to calculate the Lipschitz constant with respect to the induced norm $\|\cdot\|_S$ on $\mathbb{R}^N$.

For an arbitrary direction $h \in \mathbb{R}^N$ and a fixed parameter $\omega \in \Omega$, we have

$$\begin{aligned}
\frac{1}{2}\left(\nabla_y\left(\langle A(\omega)y - F, A(\omega)y - F\rangle_{S^{-1}}\right)\right)^{\mathsf{T}} h &= \langle A(\omega)y - F, A(\omega)h\rangle_{S^{-1}} \\
&= \left(A(\omega)y - F\right)^{\mathsf{T}} S^{-1} A(\omega)h \\
&= \left[\left(A(\omega)y - F\right)^{\mathsf{T}} S^{-1} A(\omega)h\right]^{\mathsf{T}} \\
&= h^{\mathsf{T}} \underbrace{A(\omega)^{\mathsf{T}}}_{=A(\omega)} \underbrace{S^{-\mathsf{T}}}_{=S^{-1}} \left(A(\omega)y - F\right) \\
&= h^{\mathsf{T}} A(\omega) S^{-1} \left(A(\omega)y - F\right) \\
&= h^{\mathsf{T}} A(\omega)\psi(\omega) \\
&= h^{\mathsf{T}} S S^{-1} A(\omega)\psi(\omega) \\
&= \langle S^{-1}A(\omega)\psi, h\rangle_S.
\end{aligned}$$

Therefore, we have

$$S^{-1}A(\omega)\psi = \frac{1}{2}\nabla_y\left(\|A(\omega)y - F\|^2_{S^{-1}}\right)$$

and as a direct consequence, we obtain

$$\nabla R_n(y) = \frac{1}{n}\sum_{i=1}^{n} S^{-1}A(\omega_i)\psi(\omega_i).$$

Remark 3.2.3 (Matrix decompositions for parametrized PDE constrained optimization problems)**.** If one regards a PDE similar to Eq. (3.30), with an additional control parameter as a constraint in an optimization problem, e.g. in an inverse problem setting, one needs to assemble the matrices $A(\omega)$ very often. This is really expensive and can be improved.

We use the *Discrete Empirical Interpolation Method* (DEIM) [CS10], to obtain an affine decomposition of the form

$$a(x;\omega) \approx \sum_{k=1}^{\Theta_a} \vartheta_k^a(\omega) a_k(x)$$

for the function a. The scalar-valued coefficient functions $\vartheta_k^a : \Omega \to \mathbb{R}$ can be quickly evaluated for every $\omega \in \Omega$ and the functions a_k for $k = 1, \ldots, \Theta_a$ do not depend on the random parameter ω anymore. Hence, we can apply this decomposition to the

definition of the system matrix, obtaining

$$A(\omega) \approx \sum_{k=1}^{\Theta_a} \vartheta_k^a(\omega) A_k,$$

for matrices $A_k \in \mathbb{R}^{N\times N}$, $k = 1, \ldots, \Theta_a$, which do not depend on the random parameter ω either. Now, one can save the matrices $\{A_k\}_{k=1}^{\Theta_a}$ and the functions $\{\vartheta_k^a\}_{k=1}^{\Theta_a}$, to provide a fast evaluation of the matrix $A(\omega)$ for arbitrary $\omega \in \Omega$. In this case, the computation of the norm in the dual space V_N' can be accelerated too. In a first step, we compute for the vector

$$b(\omega_i) := A(\omega_i)y - F \approx \sum_{k=1}^{\Theta_a} \vartheta_k^a(\omega) A_k y - F,\ i = 1, \ldots, n.$$

The most expensive step is the computation of the Riesz-representative, which can also be accelerated. We precompute for $k = 1, \ldots, \Theta_a$ the matrices $C_k := S^{-1}A_k$ (again by solving the systems $SC_k = A_k$) and the vector $c = S^{-1}F$ (all under the assumption, that the value Θ_a is not too large). Now we can easily compute the Riesz-representative via

$$\begin{aligned}\psi(\omega_i) = S^{-1}b(\omega_i) &= \sum_{k=1}^{\Theta_a} \vartheta_k^a(\omega_i) S^{-1}A_k y - S^{-1}F \\ &\approx \sum_{k=1}^{\Theta_a} \vartheta_k(\omega_i) C_k y - c,\end{aligned}$$

for $i = 1, \ldots, n$. The computation of $\|\cdot\|_{S^{-1}}$ can be simplified to

$$\begin{aligned}\|A(\omega_i)y - F\|_{S^{-1}}^2 &= \psi(\omega_i)^{\mathsf{T}} \left(A(\omega_i)y - F\right) \\ &\approx \sum_{k=1}^{\Theta_a} \vartheta_k^a(\omega_i)\psi(\omega_i)^{\mathsf{T}} A_k y - \psi(\omega_i)^{\mathsf{T}} F.\end{aligned}$$

If in addition, the function f on the right-hand side is also parametrized, in particular $f = f(\cdot;\omega)$, one can also apply the DEIM algorithm to f and proceed analogously.

Error bounds for the approximation using DEIM are provided in [CS10] and [CS12].

Before we consider concrete functions and a concrete parameter domain for our next benchmark problem, we want to calculate the constants used in Theorem 2.3.4 and Theorem 2.3.6. We need the following result.

Lemma 3.2.4 (Condition number of the stiffness matrix)**.** *In the above setting, we assume having a finite element discretization, with a maximum distance of $\rho^{-1}h$ between the triangulation nodes. Then the stiffness matrix S is symmetric and positive definite, with*

$$\mu_{\min}(S) \geq \rho h^2$$
$$\mu_{\max}(S) \leq \rho,$$

as bounds for the eigenvalues.

Proof. We apply [Dzi10, Thm. 3.54] to our domain $G \subset \mathbb{R}^2$. □

We now obtain bounds for the eigenvalues of our system matrix $A(\omega)$ as a direct consequence.

Corollary 3.2.5 (Bounds for the eigenvalues of the system matrix)**.** *Let $\omega \in \Omega$ be a fixed parameter. Then,*

$$\mu_{\min}(A(\omega)) \geq a_{\min}\rho h^2$$
$$\mu_{\max}(A(\omega)) \leq a_{\max}\rho$$

hold for the system matrix $A(\omega)$ in Eq. (3.36), where again the FE discretization has a maximum triangulation distance of $\rho^{-1}h$.

Proof. It is clear, that for every $\omega \in \Omega$ the matrix $A(\omega)$ is symmetric. Thus, Corollary A.1.2 yields

$$\mu_{\max}(A(\omega)) = \max_{x \in \mathbb{R}^N \setminus \{0\}} \frac{x^{\mathsf{T}} A(\omega) x}{\|x\|_2^2}$$
$$\text{and } \mu_{\min}(A(\omega)) = \min_{x \in \mathbb{R}^N \setminus \{0\}} \frac{x^{\mathsf{T}} A(\omega) x}{\|x\|_2^2}.$$

We assume $\omega \in \Omega$, $x \in \mathbb{R}^N$ and $\tilde{x} := \sum_{i=1}^N x_i \varphi_i \in V_N$. Using the bilinear map b_ω, defined in the proof of Lemma 3.2.2, we conclude

$$\begin{aligned} x^{\mathsf{T}} A(\omega) x = d_\omega(\tilde{x}, \tilde{x}) &\geq a_{\min} \langle \nabla \tilde{x}, \nabla \tilde{x} \rangle_{L^2(G)} \\ &= a_{\min} x^{\mathsf{T}} S x = a_{\min} \|x\|_S^2 \\ &\geq a_{\min} \mu_{\min}(S) \|x\|_2^2 . \end{aligned}$$

Since $x \in \mathbb{R}^N$ was arbitrary, we get

$$\mu_{\min}(A(\omega)) \geq a_{\min}\rho h^2 > 0,$$

using the lower bound provided by Lemma 3.2.4. Consequently, the eigenvalues of $A(\omega)$ have a parameter independent lower bound, which is greater than zero. For the upper bound, we proceed analogously. Thus,

$$x^{\mathsf{T}}A(\omega)x \leq a_{\max}x^{\mathsf{T}}Sx = a_{\max}\|x\|_S^2 \leq a_{\max}\mu_{\max}(S)\|x\|_2^2$$

and accordingly we obtain

$$\mu_{\max}(A(\omega)) \leq a_{\max}\rho,$$

as a parameter independent upper bound. □

Since we found parameter independent bounds for the eigenvalues, the calculation of the constants is now straight forward. As we use the unit square for our domain G, the maximum triangulation distance is bounded by $\sqrt{2}h$, where $h = \frac{1}{m}$ using m discretization points. Hence, $\rho = \frac{1}{\sqrt{2}}$ holds.[10]

Lipschitz continuity of the gradient For $v, w \in \mathbb{R}^N$, we have

$$\begin{aligned}
\|\nabla_y R_n(v) - \nabla_y R_n(v)\|_S &\leq \frac{1}{n}\sum_{i=1}^{n}\left\|S^{-1}A(\omega_i)S^{-1}A(\omega_i)(v-w)\right\|_S \\
&= \frac{1}{n}\sum_{i=1}^{n}\left\|S^{-\frac{1}{2}}A(\omega_i)S^{-1}A(\omega_i)(v-w)\right\|_2 \\
&= \frac{1}{n}\sum_{i=1}^{n}\left\|S^{-\frac{1}{2}}A(\omega_i)S^{-\frac{1}{2}}S^{-\frac{1}{2}}A(\omega_i)S^{-\frac{1}{2}}S^{\frac{1}{2}}(v-w)\right\|_2 \\
&\leq \frac{1}{n}\sum_{i=1}^{n}\left\|S^{-\frac{1}{2}}A(\omega_i)S^{-\frac{1}{2}}\right\|_2^2\left\|S^{\frac{1}{2}}(v-w)\right\|_2 \\
&\leq \max_{i=1,\dots,n}\left\|S^{-\frac{1}{2}}A(\omega_i)S^{-\frac{1}{2}}\right\|_2^2\|v-w\|_S \\
&\leq L\|v-w\|_S
\end{aligned}$$

with $L := \frac{a_{\max}^2}{h^4}$ (i.e. $L \to \infty$ as $h \to 0$). We have to prove the last inequality.

[10] This result immediately follows, since *FEniCS* uses a 2D-meshgrid with uniform mesh sizes, combined with the Pythagorean theorem.

For ω_i fixed, $i = 1, \ldots, n$, we obtain using Lemma 3.2.4 and Corollary 3.2.5 the inequalities

$$\left\| S^{-\frac{1}{2}} A(\omega_i) S^{-\frac{1}{2}} \right\|_2 = \mu_{\max} \left(S^{-\frac{1}{2}} A(\omega_i) S^{-\frac{1}{2}} \right) = \mu_{\max} \left(S^{-1} \right) \cdot \mu_{\max}(A(\omega_i))$$
$$= \frac{\mu_{\max}(A(\omega_i))}{\mu_{\min}(S)} \leq \frac{a_{\max}\rho}{\rho h^2} = \frac{a_{\max}}{h^2},$$

as a parameter independent upper bound. Now, the definition of L follows immediately.

Unbiased estimate As we use normally distributed realizations of the parameter ω, we assume $\mu = 1$ and $M_G = 1$.

Strong convexity We calculate a lower bound for the minimal eigenvalue of $\nabla_y^2 R_n(y) = \nabla_y^2 R_n$ (the Hessian matrix again does not depend on $y \in \mathbb{R}^N$). First, the Hessian is given by

$$\nabla_y^2 R_n(y) = \frac{1}{n} \sum_{i=1}^{n} S^{-1} A(\omega_i) S^{-1} A(\omega_i).$$

Now we can proceed similarly, having

$$\mu_{\min}(\nabla_y^2 R_n) \geq \min_{i=1,\ldots,n} \mu_{\min}(S^{-1} A(\omega_i))^2$$
$$\geq a_{\min}^2 h^4 =: \nu.$$

This yields the corresponding modulus of convexity $c = \frac{\nu}{2} = \frac{a_{\min}^2 h^4}{2}$, so $c \to 0$ as $h \to 0$.

For our benchmark problem, we set $\Omega := (\omega_{\min}, \omega_{\max})^2$, where $\omega_{\min} := 0.5$, $\omega_{\max} = 3.5$ act as bounds for the uncertainty parameter domain. For the functions a and f, we use

$$a(\cdot; \omega) : G \to \mathbb{R}, x \mapsto 6 + \sin(\pi x_1 \omega_1) \cdot \sin(\pi x_2 \omega_2), \omega \in \Omega,$$
$$f : G \to \mathbb{R}, x \mapsto \cos(\pi x_1) \cdot \exp(\pi x_2).$$

The number of realizations of the uncertainty parameter ω is set to $n = 30$, and we use $m = 30$, as the number of grid points for the FE discretization in each dimension. For the realization of the discrete random parameters $\{\omega_i\}_{i=1}^n$, we use a multivariate

normal distribution with mean $(2,2)^\mathsf{T} \in \mathbb{R}^2$ and covariance-matrix $0.5 \cdot \mathrm{I}_2 \in \mathbb{R}^{2\times 2}$. Values outside our desired domain Ω are cut off, using the same rules as in the previous chapter. As before, we use a batch size of $\mathrm{batch}_{\mathrm{size}} = \frac{n}{2}$, so 50% of the summands are represented in each batch. The maximum number of iterations is set to $\mathrm{iter}_{\max} = 2500$ and we use a random initial guess.

Instead of only using the theoretically calculated step size, we try running SGD with an overestimated step size of $\hat{\alpha} = 0.01$, which was calculated by trial and error, as for larger step sizes the algorithm diverged. Compared to the theoretically computed value of $\alpha_1 = \bar{\alpha} \approx 2.52 \cdot 10^{-5}$, we overestimated the maximum step size by a factor of nearly 400.

The convergence results are shown in Fig. 3.10. Table 3.3 provides additional information regarding the optimization process itself. The table includes the needed CPU times, the values of the loss function evaluated at the last iterate and the norm of the differences between the result computed by SGD and the (exact) FE solution for $\omega_{\mathrm{exact}} = (2,2)^\mathsf{T} \in \mathbb{R}^2$.

We clearly see that SGD running with an overestimated fixed step size leads to the best convergence result. After around 500 iterations, the loss curve stagnates. Hence, it would be sufficient to run SGD with $\hat{\alpha}$ using a lower maximum iteration bound. Especially for higher iteration counts, we can see the typical peaks in the loss curve. Using both, a fixed and diminishing step size sequence with the theoretically calculated initial step size, yields an improvement in the value of the loss function, but the results are not truly convincing. Particularly the diminishing step sizes really vanish for larger iteration counts and the improvement per iteration gets marginal. In the case of running SGD with an overestimated diminishing step size sequence, we obtain a clear convergence curve, which even for higher values of k still decreases. Compared to previous results, this test shows a pretty interesting outcome, as so far SGD with a diminishing step size sequence provided better results. Overall, a result of approximately 10^{-4} as the value of the loss function is sufficiently good.

Fig. 3.11, respectively Table 3.4, show, how the results change, if we increase the maximum number of iterations to $\mathrm{iter}_{\max} = 4000$. While SGD running with the theoretically calculated diminishing step size sequence still stagnates, the results obtained by running SGD with the theoretically computed fixed step size tremendously improves. In Fig. 3.11, we can see that after the last iteration the orange curve is even below the blue one. Similar to the behaviour in Fig. 3.10, SGD running with the overestimated fix steps size stagnates and we do not gain any major improvements

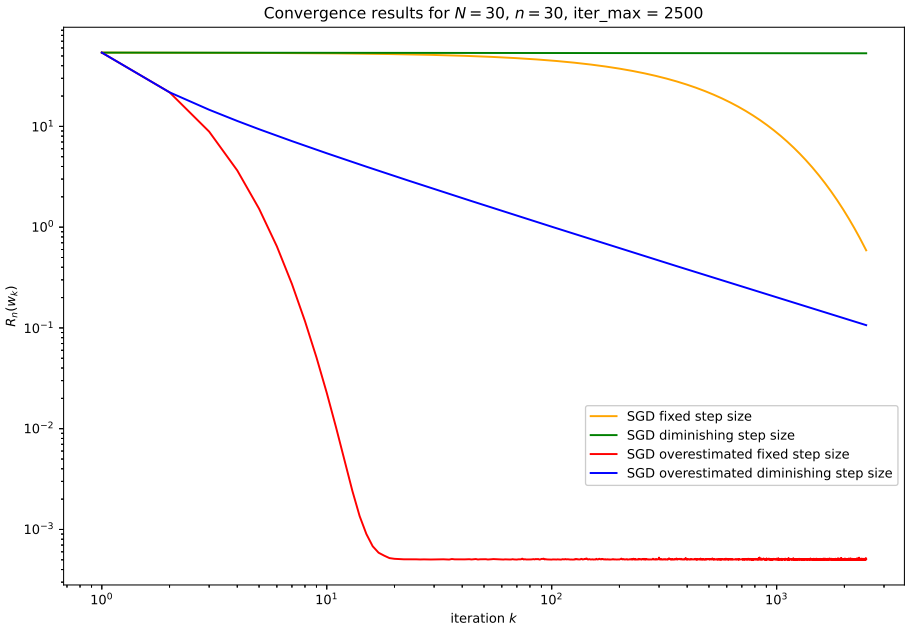


Figure 3.10.: Convergence results for the PDE benchmark test with different step size sequences and $\text{iter}_{\max} = 2500$. The visualization is a log-log-plot. For the tests with an overestimated step size, we used $\hat{\alpha} = 0.01$.

Step size sequence	Loss	CPU time [s]	Error in $\|\cdot\|_\infty$
fixed	$5.8894 \cdot 10^{-01}$	$1.2166 \cdot 10^{+02}$	$1.3846 \cdot 10^{-02}$
fixed (overestimated)	$\mathbf{5.0621 \cdot 10^{-04}}$	$1.1646 \cdot 10^{+02}$	$\mathbf{2.5746 \cdot 10^{-03}}$
diminishing	$5.3141 \cdot 10^{+01}$	$\mathbf{1.0630 \cdot 10^{+02}}$	$1.2190 \cdot 10^{-01}$
diminishing (overestimated)	$5.3141 \cdot 10^{+01}$	$1.1794 \cdot 10^{+02}$	$6.7544 \cdot 10^{-03}$

Table 3.3.: Convergence results and CPU times for the PDE benchmark test with $\text{iter}_{\max} = 2500$. The column "Error in $\|\cdot\|_\infty$" shows the maximum norm of the difference between the result computed by SGD and the (exact) FE solution for $\omega_{\text{exact}} = (2, 2)^\mathsf{T} \in \mathbb{R}^2$.

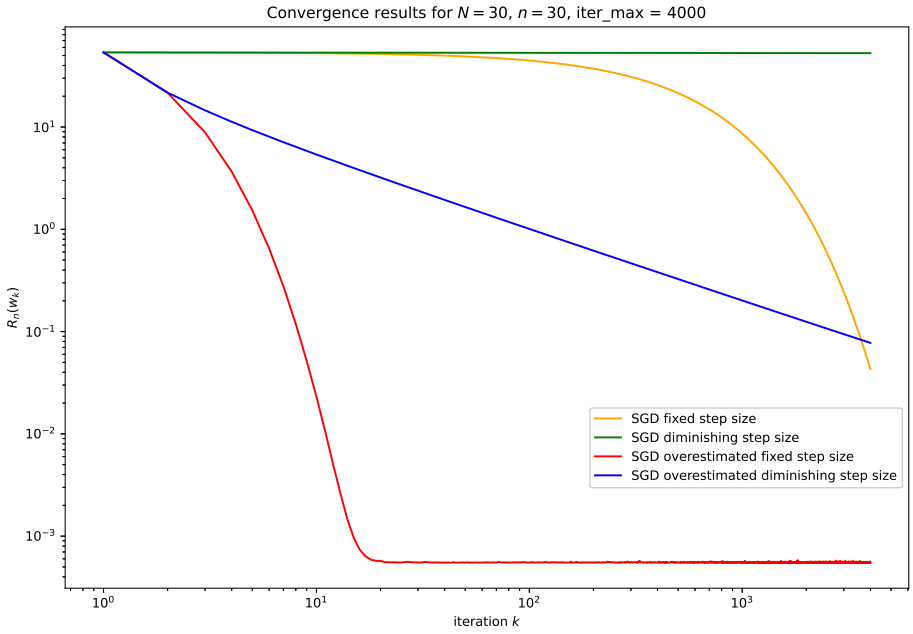


Figure 3.11.: Convergence results for the PDE benchmark test with different step size sequences and $\text{iter}_{\max} = 4000$. The visualization is a log-log-plot. For the tests with an overestimated step size, we used $\hat{\alpha} = 0.01$.

using this step size sequence with more iterations.

In conclusion, we register that not always the diminishing step size sequence should be preferred over a constant step size choice. Furthermore, it can be worth it, to purposely overestimate the theoretically calculated step size bound, as we have seen in our last example. Here, trial and error is (usually) required.

3.3. Outlook: Applications for non-convex objective functions

In Section 2.4 we expand the analysis of the stochastic gradient descent method to non-convex objective functions. In practical applications, calculating the necessary constants, like in the previous section, to get the optimal step size (or step size sequence) is not only really difficult, but rather almost impossible.

As a real-world application, we will take a look at a deep learning approach to solve the visual odometry problem. To be more precise, a colleague and I tried to predict the velocity of a car, only using video footage produced from a dashcam inside the car, see [WH21]. We tried different approaches, including convolutional neural networks

Step size sequence	Loss	CPU time [s]	Error in $\|\cdot\|_\infty$
fixed	$4.2946 \cdot 10^{-02}$	$2.4742 \cdot 10^{+02}$	$3.9807 \cdot 10^{-03}$
fixed (overestimated)	$\mathbf{5.5042 \cdot 10^{-04}}$	$2.0205 \cdot 10^{+02}$	$\mathbf{1.1149 \cdot 10^{-03}}$
diminishing	$5.2797 \cdot 10^{+01}$	$1.7171 \cdot 10^{+02}$	$1.2154 \cdot 10^{-01}$
diminishing (overestimated)	$5.2797 \cdot 10^{+01}$	$\mathbf{1.6672 \cdot 10^{+02}}$	$5.2990 \cdot 10^{-03}$

Table 3.4.: Convergence results and CPU times for the PDE benchmark test with $\text{iter}_{\max} = 4000$. The column "Error in $\|\cdot\|_\infty$" shows the norm of the difference between the result computed by SGD and the (exact) FE solution for $\omega_{\text{exact}} = (2, 2)^\mathsf{T} \in \mathbb{R}^2$.

(CNN) for optical flow fields and siamese CNNs for two consecutive frames[11]. The architecture of the networks are all similar to the one in Fig. 3.12.

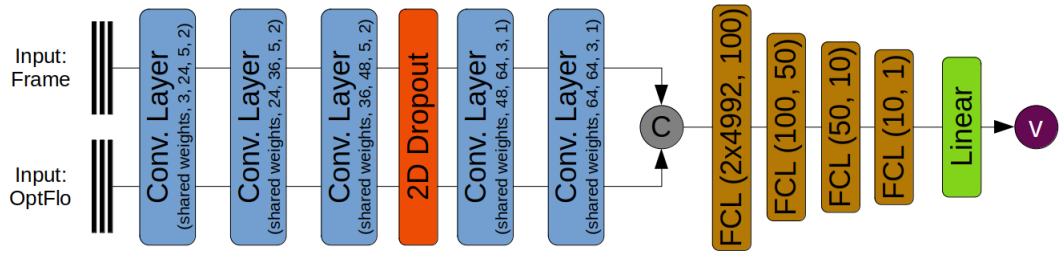


Figure 3.12.: Example structure of one of our networks. Here we used a siamese approach with the flow field and the raw frame, to let the network decide which features of the flow field should be extracted. The abbreviations "conv. layer" and "FCL" represent a *convolutional layer*, respectively a *fully connected layer* or so-called *dense layer*. After passing the siamese part inside the net, the resulting feature vector is concatenated (represented with a "c" in the figure) and mapped into the fully connected layers.

To train the networks, we used SGD (without momentum) as an optimizer. Although this is, compared to e.g. autonomous driving, a simplified task and we sampled down the frames twice, in the end we had a database with a size between 10 and 15 GB. Here one can clearly see that even loading all frames at once into the memory is impossible – so especially a full gradient step. All of our networks, like the one in Fig. 3.12, had a total number of around 640.000 parameters, which we tried to optimize. In practical applications, you split your dataset into small batches, with a fixed batch size. The batch size should be chosen as big as possible, to get, recalling

[11] All networks are implemented in *PyTorch* [Pas+19].

our theory of the first chapters, the best possible convergence result. One full cycle, consisting of all batches passing the model is called an *epoch*. In our case, the training of one epoch took between one and two hours and we had to train between 10 and 150 epochs (depending on the network).

In the two graphs Fig. 3.13 and 3.14 one can see how the stochastic gradient descent performs on our dataset with a batch size of 64 frames. We worked on a test and validation data set. Both plots emphasize the strengths of SGD. Although the loss does not decrease in every epoch, it decreases in expectation.

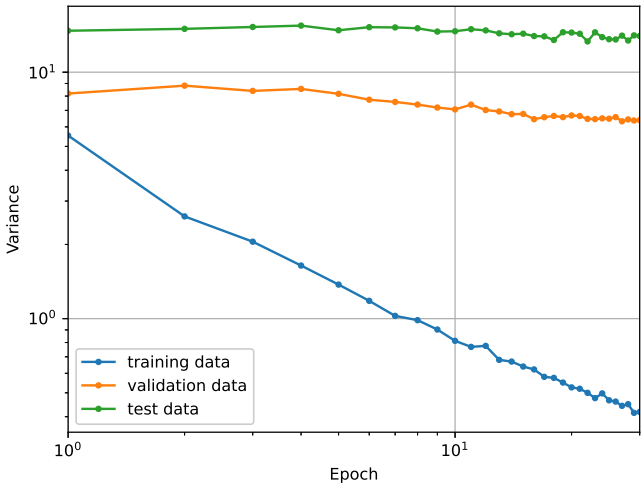


Figure 3.13.: Training with SGD on the initial network. We still had some problems with overfitting. We used a learning rate of 0.001 and a plateau scheduler.

As mentioned in the beginning of this chapter, the choice of the step size is in practical applications really hard. Setting up a step size sequence like in Theorem 2.3.6 is even harder, if not impossible, as the needed constants are unknown. To simulate a step size sequence, most of the commonly used frameworks provide a so called *scheduler*, which adjusts the learning rate according to predefined rules. Possible options are for example a reduction of the learning rate, if the loss does not change after a specified number of epochs in between a given tolerance, or one can multiply the learning rate by a factor after a specified number of iterations.

In this project, we used two different schedulers. First, we used an algorithm, which reduces the learning rate, if the loss is on a plateau for two epochs, which mostly worked best. Some of our networks needed a bit more fine adjustments, so we predefined a sequence of epochs (a typical choice was here $\{10, 30, 70, 120\}$, cf. Remark 2.3.5) after

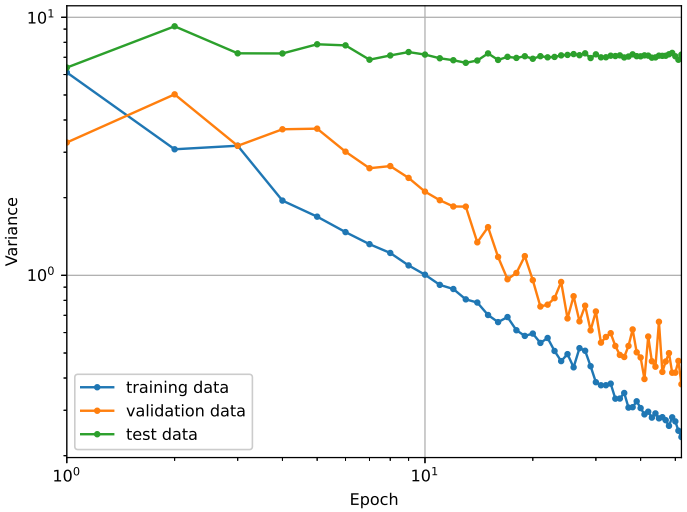


Figure 3.14.: Training with SGD on one of the improved networks. We started again with a learning rate of 0.001 and multiplied the learning rate by a factor 0.3 after the epochs 10, 30, 70 and 120. One can clearly see small peaks in the loss curve, which are typical for a stochastic optimizer, but in expectation the loss is decreasing to zero.

each the learning rate was multiplied by a factor of 0.3.

Chapter 4.

Conclusion

We theoretically established SGD for strongly convex objective functions, using constant and diminishing step size sequences. SGD convinced as a stable optimizer, which is able to ensure convergence in expectation for large-scale problems, even under the presence of noise or uncertainty in the gradient calculation. We then expanded the results to non-convex objective functions and later generalized step size choices for practical applications, when we introduced schedulers.

Several numerical experiments substantiated our theoretical results. In a preliminary study, we showed the importance of the step size sequence and tried to sensitize the reader, when choosing step sizes for SGD in practical applications. SGD convinced as a really robust optimization algorithm in the case of solving the ODE boundary value problem. These results were transferred to the PDE boundary value problem. We could even see, how resilient SGD is with respect to overestimated step size choices, when we gained a major speed up by consciously enlarging the theoretically calculated step size. In the case of the PDE boundary value problem, one could try to introduce more advanced preconditioning techniques, to further improve the results.

In Remark 3.2.3, we showed how the (projected) SGD algorithm could be used for stochastic optimization problems, when dealing with (PDE constrained) inverse problems. Another possibility to build on this work would be to generalize SGD to Stochastic gradient descent with momentum (SGDM), see e.g. [LGY20].

Appendix A.

Additional lemmas

A.1. Matrix properties

A.1.1. Characterization of uniform positive definite matrices

Lemma A.1.1 (Rayleigh quotient)**.** *Let $A \in \mathbb{C}^{n\times n}$ be a hermitian matrix, with ordered eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ (ignoring their multiplicities). Then for all $x \in \mathbb{C}^n\backslash\{0\}$ we get*

$$\lambda_1 \leq \frac{x^\mathsf{T} Ax}{\|x\|_2^2} \leq \lambda_n. \tag{A.1}$$

Proof. The spectral theorem [Fis13, p. 307] yields, that we have an orthonormal basis $\{v_i\}_{i=1}^n$ of eigenvectors $v_i \in \mathbb{C}^n\backslash\{0\}$ with $Av_i = \lambda_i v_i$ for $i = 1, \ldots, n$. Using the Bessel equation [Fis13, p. 300], we obtain for arbitrary $x \in \mathbb{C}^n\backslash\{0\}$ with $x = \sum_{i=1}^n \langle x, v_j\rangle v_j$, that

$$\|x\|_2^2 \overset{\text{Bessel}}{=} \sum_{i=1}^n |\langle x, v_i\rangle|^2$$

holds. Therefore, our claim follows as an easy consequence, since we have

$$\begin{aligned}\frac{x^\mathsf{T} Ax}{\|x\|_2^2} &= \frac{\langle \sum_{i=1}^n \lambda_i \langle x, v_j\rangle v_j, x\rangle}{\|x\|_2^2} \\ &\leq \lambda_n \frac{\sum_{i=1}^n |\langle x, v_i\rangle|^2}{\|x\|_2^2} = \lambda_n.\end{aligned}$$

The other inequality follows analogously. □

Corollary A.1.2 (Courant–Fischer–Weyl min-max principle)**.** *Let $A \in \mathbb{C}^{n\times n}$ be a hermitian matrix, with ordered eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ (ignoring their multiplicities). Then we can estimate the minimal and maximal eigenvalue of A via*

$$\mu_{\min}(A) = \min_{x\in\mathbb{C}^n\setminus\{0\}} \frac{x^\mathsf{T} Ax}{\|x\|_2^2}$$
$$\mu_{\max}(A) = \max_{x\in\mathbb{C}^n\setminus\{0\}} \frac{x^\mathsf{T} Ax}{\|x\|_2^2}.$$

Proof. The result is a direct consequence of the proof of Lemma A.1.1. We just need to consider the eigenpairs (λ_1, v_1) and (λ_n, v_n), follow the proof with $x := v_1$, respectively $x := v_n$, and obtain the desired equalities. □

Corollary A.1.3 (Hermitian positive definite matrices are uniform positive definite)**.** *Let $A \in \mathbb{C}^{n\times n}$ be a hermitian positive definite matrix, then A is uniform positive definite, meaning that there exists a $C \in \mathbb{R}^+$, such that for every $d \in \mathbb{C}^n$ the inequality*

$$d^\mathsf{T} Ad \geq C \cdot \|d\|_2^2$$

holds. In particular, we can choose $C = \lambda_{\min}(A) \in \mathbb{R}^+$.

Proof. Using Lemma A.1.1, we observe for $d \in \mathbb{C}^n$

$$d^\mathsf{T} Ad \geq \lambda_{\min}(A) \cdot \|d\|_2^2 .$$

Since A is positive definite, we have $\lambda_{\min}(A) > 0$. By defining $C := \lambda_{\min}(A)$, we obtain the desired result. □

A.1.2. Convergence analysis of the Jacobi method

Definition A.1.4 (Spectral radius of a matrix)**.** The **spectral radius** of a matrix $A \in \mathbb{R}^{n\times n}$ is defined as

$$\rho(A) := \max_{j=1,\ldots,n} |\lambda_j(A)| ,$$

where $\lambda_j(A) \in \mathbb{C}$ denotes the j-th eigenvalue of the matrix A (ignoring multiplicities).

Lemma A.1.5 (Basic properties of the spectral radius)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a matrix. For all induced matrix norms, $\rho(A) \leq \|A\|$ holds. If A is additionally symmetric and $\|\cdot\| = \|\cdot\|_2$ is the 2-norm, we have $\rho(A) = \|A\|_2$.*

Proof. First, let $A \in \mathbb{R}^{n\times n}$ be an arbitrary matrix and (λ, v) be an eigenpair of A. Using the relation $Av = \lambda v$, we obtain

$$|\lambda| \, \|v\| = \|\lambda v\| = \|Av\| \leq \|A\| \, \|v\| \,,$$

and since $\|v\| \neq 0$, we have $|\lambda| \leq \|A\|$. Since this holds for any eigenvalue, the first claim follows.

For the second claim we assume that A is symmetric. By the definition of the 2-Norm we observe

$$\|A\|_2^2 = \lambda_{\max}(A^\mathsf{T} A) = \lambda_{\max}(A^2) = \max |\lambda(A)|^2 = \rho(A)^2.$$

The claim follows. □

Theorem A.1.6 (Convergence of stationary methods)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a regular matrix, $A = M - N$ with M regular and $f \in \mathbb{R}^n$. The stationary iterative method*

$$Mx_{k+1} = Nx_k + f$$

converges for any initial vector $x_0 \in \mathbb{R}^n$ to the solution x of the linear system $Ax = f$ if and only if $\rho(M^{-1}N) < 1$.

Proof. See [CG20, Theorem 3]. □

Corollary A.1.7 (Convergence of the Jacobi method for strictly diagonally dominant matrices)**.** *If the matrix $A \in \mathbb{R}^{n\times n}$ is strictly diagonally dominant, i.e.*

$$|a_{ii}| > \sum_{j\neq i} |a_{ij}| \,, \qquad \text{for } i = 1, \ldots, n, \tag{A.2}$$

then the Jacobi method, i.e. the iterative method induced by the splitting $M = \operatorname{diag}(A)$ and $N = \operatorname{diag}(A) - A$, converges.

Proof. The condition (A.2) allows us to estimate

$$\left\|M^{-1}N\right\|_\infty = \max_{i=1,\ldots,n} \frac{1}{|a_{ii}|} \sum_{j\neq i} |a_{ij}| < 1.$$

Using Lemma A.1.5, we observe

$$\rho(M^{-1}N) \leq \left\|M^{-1}N\right\|_\infty < 1,$$

which, together with Theorem A.1.6, yields the desired result. □

A.2. Probability theory

A.2.1. Convergence in probability and Markov's inequality

Definition A.2.1 (Convergence in probability)**.** Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space with a sequence of real-valued random variables $\{X_n\}_{n\in\mathbb{N}} : \Omega \to \mathbb{R}$. We say $\{X_n\}_{n\in\mathbb{N}}$ converges **in probability** to a real-valued random variable $X : \Omega \to \mathbb{R}$, if for every $\varepsilon > 0$

$$\lim_{n\to\infty} \mathbb{P}\left(|X_n - X| \geq \varepsilon\right) = 0$$

holds. We write $X_n \xrightarrow{\mathbb{P}} X$.

Lemma A.2.2 (Markov's inequality)**.** *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, $X : \Omega \to \mathbb{R}$ a real-valued random variable and $h : \mathbb{R} \to [0, \infty)$ a monotonically increasing function. Let $\varepsilon \geq 0$ with $h(\varepsilon) > 0$, then*

$$\mathbb{P}(X \geq \varepsilon) \leq \frac{\mathbb{E}\left[h(X)\right]}{h(\varepsilon)}.$$

Proof. As a monotone function, h is $\mathcal{B}(\mathbb{R})$-measurable. Hence, we have

$$\mathbb{P}(X \geq \varepsilon) = \int_\Omega \chi_{\{X\geq\varepsilon\}} \,\mathrm{d}\mathbb{P} \leq \int_\Omega \chi_{\{X\geq\varepsilon\}} \frac{h(X)}{h(\varepsilon)} \,\mathrm{d}\mathbb{P} \leq \frac{\mathbb{E}\left[h(X)\right]}{h(\varepsilon)}.$$

□

A.3. Functional analysis

A.3.1. The lemmas of Fréchet-Riesz, Lax-Milgram and Friedrich

Lemma A.3.1 (Fréchet-Riesz)**.** *Consider a Hilbert space $(H, \langle\cdot,\cdot\rangle_V)$ with its dual space H'. For every $u' \in H'$, there exists a unique $u \in H$ with*

$$\langle u', v\rangle_{H'\times H} = \langle u, v\rangle_H \ \forall v \in H$$

and $\|u'\|_{H'} = \|u\|_H$.

Proof. See [Wer07, Theorem V.3.6, p. 226]. □

Lemma A.3.2 (Lax-Milgaram)**.** *Let H be a Hilbert space over a field $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$ and $B : H \times H \to \mathbb{K}$ a continuous bilinear function. Further, assume having a constant $p > 0$ with*

$$|B(u,u)| \geq p\,\|u\|^2 \ \forall u \in H.$$

Then there exists for every $F \in H'$ a unique $u \in H$, such that

$$F(v) = B(u,v)\ \forall v \in H$$

holds.

Proof. See [Den21, Thm. 6.6]. □

Lemma A.3.3 (Friedrichs inequality)**.** *Let $G \subset \mathbb{R}^n$ be a bounded Lipschitz domain of $\mathbb{R}^n$ and $\mathrm{d}(G) := \sup_{x,y\in G} \|x - y\|_{\mathbb{R}^n}$ the diameter of the domain. Then for every $u \in H_0^1(G)$ the inequality*

$$\int_G u(x)\,\mathrm{d}x \leq \mathrm{d}(G) \int_G |\nabla u(x)|^2\,\mathrm{d}x$$

holds.

Proof. See [Rek12, Thm. 18.1, p. 118]. □

Bibliography

[Aln+15] Martin S. Alnæs et al. "The FEniCS Project Version 1.5". In: *Archive of Numerical Software* 3.100 (2015). DOI: 10.11588/ans.2015.100.20553.

[BCN18] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. "Optimization Methods for Large-Scale Machine Learning". In: *SIAM Review* 60.2 (2018), pp. 223–311. DOI: 10.1137/16M1080173. eprint: https://doi.org/10.1137/16M1080173. URL: https://doi.org/10.1137/16M1080173.

[CG20] Gabriele Ciaramella and Martin J. Gander. *Iterative Methods and Preconditioners for Systems of Linear Equations*. 2020.

[CS10] Saifon Chaturantabut and Danny C. Sorensen. "Nonlinear Model Reduction via Discrete Empirical Interpolation". In: *SIAM Journal on Scientific Computing* 32.5 (2010), pp. 2737–2764.

[CS12] Saifon Chaturantabut and Danny C. Sorensen. "A state space error estimate for POD-DEIM nonlinear model reduction". In: *SIAM Journal on Numerical Analysis* 50.1 (2012), pp. 46–63.

[Den21] Robert Denk. "Theorie partieller Differentialgleichungen – Vorlesungsskript". Konstanz, 2021.

[DR08] Wolfgang Dahmen and Arnold Reusken. *Numerik für Ingenieure und Naturwissenschaftler*. 2nd ed. Berlin Heidelberg New York: Springer Spektrum, 2008. ISBN: 978-3-540-76493-9.

[Dzi10] Gerhard Dziuk. *Theorie und Numerik partieller Differentialgleichungen*. Berlin: Walter de Gruyter, 2010. ISBN: 978-3-110-14843-5.

[Fis13] Gerd Fischer. *Lineare Algebra - Eine Einführung für Studienanfänger*. 18th ed. Berlin Heidelberg New York: Springer Spektrum, 2013. ISBN: 978-3-658-03945-5.

[Jun21] Michael Junk. "Numerik partieller Differentialgleichungen – Vorlesungsskript". Konstanz, 2021.

[Kle13] Achim Klenke. *Wahrscheinlichkeitstheorie*. 3rd ed. Berlin Heidelberg New York: Springer Spektrum, 2013. ISBN: 978-3-642-36017-6.

[LGY20] Yanli Liu, Yuan Gao, and Wotao Yin. *An Improved Analysis of Stochastic Gradient Descent with Momentum*. 2020. arXiv: 2007.07989 [math.OC].

[Pas+19] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[Rek12] Karel Rektorys. *Variational Methods in Mathematics, Science and Engineering*. Berlin Heidelberg: Springer Science & Business Media, 2012. ISBN: 978-9-401-16450-4.

[Wer07] Dirk Werner. *Funktionalanalysis*. 6th ed. Berlin Heidelberg New York: Springer-Verlag, 2007. ISBN: 978-3-540-72536-7.

[WH21] Florian Wolf and Franz Herbst. *A machine learning approach to predict a vehicles velocity using dashcam video footage*. Tech. rep. Konstanz: University Konstanz, Department of Computer and Information Science, 2021.