

University of Konstanz

Department of Mathematics and Statistics

Master Thesis

Efficient Stochastic Descent Methods for PDE-Constrained Optimization with Uncertain Coefficients

submitted by

Calvin Feineis

Konstanz, April 2021

Supervisor and Reviewer: Prof. Dr. Stefan Volkwein

Abstract

In this thesis, we consider a convex, elliptic PDE-constrained optimal control problem that is subject to uncertainty. To solve this problem numerically we use three stochastic descent methods, namely the Stochastic Gradient method, the Stochastic Variance Reduced Gradient method and the Stochastic Adaptive Sampling method. We state theoretical convergence results for the three stochastic descent methods and present a setting in which we conduct numerical tests to compare the convergence behaviour and the CPU time. The numerical experiments show that a modification of the Stochastic Adaptive Sampling method in combination with the Barzilai-Borwein step size rule is the superior choice for the specific problem.

Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 1 |
| 2 | Definitions and Notations | 4 |
| 3 | Problem Formulation | 15 |
| 3.1 | Finite Element Method | 21 |
| 3.2 | Generated Probability Space | 28 |
| 4 | Descent Methods and Convergence Analysis | 30 |
| 4.1 | Projected Stochastic Gradient | 31 |
| 4.2 | Projected Stochastic Variance Reduced Gradient | 35 |
| 4.3 | Projected Stochastic Adaptive Sampling | 42 |
| 5 | Numerical Experiments and Results | 50 |
| 5.1 | Projected Stochastic Gradient | 52 |
| 5.2 | Projected Stochastic Variance Reduced Gradient | 54 |
| 5.3 | Projected Stochastic Adaptive Sampling | 56 |
| 5.4 | Comparison of the Descent Methods | 60 |
| 6 | Conclusion and Outlook | 63 |
| | Appendix | 65 |
| | Bibliography | 67 |

Chapter 1

Introduction

This thesis considers three stochastic descent methods, which are applied on an optimal control problem. As a part of numerical optimization, optimal control problems normally occur when there exists a desired state in a system, but the state can only be influenced indirectly by a control that is linked to the state through a partial differential equation (PDE). It is then desirable to choose the control in such a way that the resulting state is as close to the desired state as possible, while also trying to avoid a large control, because in most applications a large control induces high costs. This task is formulated in an optimization setting as the minimization of a cost function depending on the control variable.

The optimal control problem considered in this thesis additionally takes randomness into account by allowing the PDE as well as the state variable to have an input of which only the distribution is known. Thus, the cost function depends on a random variable. Minimizing the cost function for every realization of the random variable, if even feasible, would lead to the optimal control depending on the random variable, which is not desirable for real applications as the realization is not known a priori. Therefore the expected value of the underlying costs function is minimized, such that a deterministic optimal control can be obtained. The resulting optimization problem is computationally challenging, mainly because the PDE has to be solved numerous times for different realizations of the random input.

A popular way to solve problems of this kind are the so-called stochastic approximation methods, which led back to Robbins and Monro [RM51] in 1951. Since then, many extensions to the first basic Stochastic Gradient method were developed, which can be sorted into two main categories. Firstly, there are variance reduction

methods, such as the Stochastic Variance Reduced Gradient method [Tan+16], the Stochastic Adaptive Sampling method [BBN17] or the Stochastic Averaged Gradient method [SRB16]. Secondly, as in the deterministic optimization, there are methods that incorporate second-order information such as Stochastic Newton [KMR19] or Quasi-Newton methods [Byr+15]. There are also different step size algorithms, which go hand in hand with the descent methods. Examples can be found [Tan+16] or in [BBN17]. A general overview of the past and present of stochastic descent methods can be found in [BCN18], for instance.

The basis for this thesis is [GP19] in which Geiersbach and Pflug have shown that there exists a solution to an optimal control problem, which includes randomness, for a convex cost function on a closed and convex subspace of a Hilbert space. They additionally have proven convergence results for the Stochastic Gradient method. A similar problem was investigated in [Sin20], where additionally a Reduced Basis method was implemented. This thesis adds to the work of Geiersbach and Pflug by stating convergence analysis and numerical results for two more advanced stochastic descent methods on a similar problem as well as giving a comparison of the two methods to the Stochastic Gradient method and the deterministic Gradient Descent method.

This thesis is structured in the following way. The *second chapter* covers relevant notations, definitions and theorems of function spaces, linear algebra, optimization and probability theory.

The *third chapter* is used to define a box- and PDE-constrained optimal control problem which is subject to uncertainty. Further, it is shown that this problem is convex and that a globally optimal solution exists. Moreover, the Finite Element Method for this problem is introduced and a short excursus on the Reduced Basis method is given.

In the *fourth chapter*, general stochastic descent as well as three specific stochastic descent methods, namely the Projected Stochastic Gradient method, the Projected Stochastic Variance Reduced Gradient method and the Projected Stochastic Adaptive Sampling method are introduced including their algorithms and convergence theory. The *fifth chapter* starts with the description of a specific optimization problem, used for the numerical experiments, which is of the form described in the third chapter. Then numerical results are presented including convergence behaviour for the stochastic descent methods of the fourth chapter for different choices of step size, batch size and other parameters as well as a comparison of the required CPU times.

Further, a comparison against the deterministic projected Gradient method with projected Armijo line search is given.

Lastly, a summary of the results as well as an outlook on further extensions to the considered problem and the descent methods is given.

Chapter 2

Definitions and Notations

In this chapter, the relevant fundamental definitions, notations and theorems which are used in this thesis are presented.

Function spaces

As function spaces play a central role in probability theory as well as in the theory of partial differential equations (PDEs), the basic definitions and statements are summarized in this section. All of these can also be found in [Dzi10] or [DR12], for instance.

Definition 2.1 (σ -Algebra). Let \mathcal{D} be a set, $2^{\mathcal{D}}$ be the power set of \mathcal{D} and $\mathcal{F} \subset 2^{\mathcal{D}}$. Then we call \mathcal{F} a σ -algebra on \mathcal{D} if:

- (i) $\emptyset \in \mathcal{F}$.
- (ii) For every $A \in \mathcal{F}$ it holds that $A^C := \{x \in \mathcal{D} : x \notin A\} \in \mathcal{F}$.
- (iii) For $(A_n)_{n \in \mathbb{N}} \subset \mathcal{F}$ it holds that $\bigcup_{n \in \mathbb{N}} A_n \in \mathcal{F}$.

We then call $(\mathcal{D}, \mathcal{F})$ a *measurable space*.

For $\mathcal{G} \subset 2^{\mathcal{D}}$ we will further denote

$$\sigma(\mathcal{G}) := \bigcap \{ \mathcal{A} \supset \mathcal{G} : \mathcal{A} \text{ is } \sigma\text{-algebra on } \mathcal{D} \}$$

as the *smallest σ -algebra containing \mathcal{G}* .

Definition 2.2 (Measure). Let $(\mathcal{D}, \mathcal{F})$ be a measurable space. We then call $\mu : \mathcal{F} \rightarrow [0, \infty]$ a *measure* on \mathcal{F} if:

- (i) $\mu(\emptyset) = 0$.

(ii) For $(A_n)_{n \in \mathbb{N}} \subset \mathcal{F}$ with $A_n \cap A_m = \emptyset$ ($n \neq m$) it holds that

$$\mu\left(\bigcup_{n \in \mathbb{N}} A_n\right) = \sum_{n \in \mathbb{N}} \mu(A_n).$$

If additionally $\mu(\mathcal{D}) = 1$ we call μ a *probability measure*.

Definition 2.3 (Measure space). Let \mathcal{D} be a set, \mathcal{F} be a σ -algebra on \mathcal{D} and μ be a measure on \mathcal{F} . Then we call $(\mathcal{D}, \mathcal{F}, \mu)$ a *measure space*. If μ is a probability measure we call $(\mathcal{D}, \mathcal{F}, \mu)$ a *probability space*.

Definition 2.4 (Borel- σ -algebra). We call the σ -algebra which is generated by all half-open sets in \mathbb{R}^m

$$\mathcal{B}(\mathbb{R}^m) := \sigma(\{(a, b] : a, b \in \mathbb{R}^m, a < b\})$$

the *Borel- σ -algebra*.

Definition 2.5 (Measurable function). Let $(\mathcal{D}, \mathcal{F}, \mathbb{P})$ be a measure space. We then call a function $f : \mathcal{D} \rightarrow \mathbb{R}^m$ *measurable* if $f^{-1}(B) \in \mathcal{F}$ for all $B \in \mathcal{B}(\mathbb{R}^m)$.

Definition 2.6 (Almost sure). Let \mathcal{F} be a σ -algebra and μ be a measure. An event $A \in \mathcal{F}$ is called to occur *μ -almost sure* (μ -a.s.) if $\mu(A^C) = 0$. If μ is a probability measure this is equivalent to $\mu(A) = 1 - \mu(A^C) = 1$. We will say almost sure and omit the μ if the measure used can be easily derived from the context.

Definition 2.7 (L^p -spaces). Let $(\mathcal{D}, \mathcal{F}, \mu)$ be a measure space. Then for $1 \leq p < \infty$ we define $\mathcal{L}^p(\mathcal{D})$ as the set of measurable functions $f : \mathcal{D} \rightarrow \mathbb{R}$ with

$$\|f\|_{L^p} := \left(\int_{\mathcal{D}} |f(x)|^p d\mu(x) \right)^{\frac{1}{p}} < \infty.$$

For $f, g \in \mathcal{L}^p(\mathcal{D})$ we define the relation

$$f \sim g \Leftrightarrow f = g \text{ } \mu\text{-a.s.} \Leftrightarrow \mu(\{x \in \mathcal{D} \mid f(x) \neq g(x)\}) = 0.$$

Then the L^p -space is defined by

$$L^p(\mathcal{D}) := \{[f] \mid f \in \mathcal{L}^p(\mathcal{D})\}$$

where $[f]$ is the equivalence class of f under the relation \sim . To simplify the notation we still denote f for the equivalence class of f .

Definition 2.8 (Local integrable). Let \mathcal{D} be an open set in \mathbb{R}^n . Then the *set of local integrable functions* is defined by

$$L^1_{\text{loc}}(\mathcal{D}) := \{u : \mathcal{D} \rightarrow \mathbb{R} \mid \forall E \subset \mathcal{D}, E \text{ compact} : u|_E \in L^1(E)\}$$

Definition 2.9 (C_0^∞). Let $\mathcal{D} \subset \mathbb{R}^n$ be an open set. We define the *space of test functions* on \mathcal{D} as

$$C_0^\infty(\mathcal{D}) = \{\varphi \in C^\infty(\mathcal{D}) \mid \text{there exists } K \subset \mathcal{D} \text{ compact with } \text{supp } \varphi \subset K\}$$

where $\text{supp } \varphi = \overline{\{x \in \mathcal{D} : \varphi(x) \neq 0\}}$ is the closure of $\{x \in \mathcal{D} : \varphi(x) \neq 0\}$ in \mathcal{D} .

Definition 2.10 (Weak derivative). Let \mathcal{D} be an open set in \mathbb{R}^n and $f \in L^1_{\text{loc}}(\mathcal{D})$. Then f is called α -*weak differentiable* if a function $\partial^\alpha f \in L^1_{\text{loc}}(\mathcal{D})$ exists which satisfies

$$\int_{\mathcal{D}} f(x) \partial^\alpha \varphi(x) \, dx = (-1)^{|\alpha|} \int_{\mathcal{D}} \partial^\alpha f(x) \varphi(x) \, dx$$

for all $\varphi \in C_0^\infty(\mathcal{D})$, a multi-index $\alpha \in \mathbb{N}_0^n$ and $|\alpha| = \sum_{i=1}^n \alpha_i$. Then $\partial^\alpha f$ is called the α -*weak derivative of f* .

Definition 2.11 (Sobolev space). Let \mathcal{D} be an open set in \mathbb{R}^n and $k \in \mathbb{N}$. We then define the *Sobolev space* by

$$H^k(\mathcal{D}) = \{u \in L^2(\mathcal{D}) \mid \partial^\alpha u \in L^2(\mathcal{D}) \text{ exists for all } \alpha \in \mathbb{N}_0^n \text{ with } |\alpha| \leq k\}.$$

As a norm on $H^k(\mathcal{D})$ we use

$$\|u\|_{H^k(\mathcal{D})} = \left(\sum_{|\alpha| \leq k} \int_{\mathcal{D}} |\partial^\alpha u(x)|^2 \, dx \right)^{\frac{1}{2}}.$$

Further we define $H_0^k(\mathcal{D})$ as the closure of $C_0^\infty(\mathcal{D})$ with respect to the $\|\cdot\|_{H^k(\mathcal{D})}$ -norm.

Definition 2.12 (Weak Solution to a linear elliptic PDE). Let $\mathcal{D} \subset \mathbb{R}^d$ be an open, bounded set with Lipschitz continuous boundary $\partial\mathcal{D}$ and $a, f : \mathcal{D} \rightarrow \mathbb{R}$. Consider the partial differential equation

$$\begin{aligned} -\nabla \cdot (a(x) \nabla y(x)) &= f(x), & x \in \mathcal{D}, \\ y(x) &= 0, & x \in \partial\mathcal{D}. \end{aligned} \tag{2.1}$$

We define $y \in H_0^1(\mathcal{D})$ to be a *weak solution* to (2.1) if

$$\int_{\mathcal{D}} a(x) \nabla y(x) \cdot \nabla \varphi(x) \, dx = \int_{\mathcal{D}} f(x) \varphi(x) \, dx$$

holds true for all $\varphi \in H_0^1(\mathcal{D})$.

Optimization

This section covers the relevant definitions and theorems to define optimality and convergence rates in the setting used in this thesis. Further information on this can be found in [Vol19], for instance.

Definition 2.13 (Minimum point). We call a point $\bar{u} \in \mathcal{U} \subset \mathbb{R}^n$ a *global minimum point* of a function $\hat{J} : \mathcal{U} \rightarrow \mathbb{R}$ if $\hat{J}(\bar{u}) \leq \hat{J}(u)$ for all $u \in \mathcal{U}$.

A point $\bar{u} \in \mathcal{U}$ is called *local minimum point* of $\mathcal{U} \subset \mathbb{R}^n$ if there exists a neighbourhood $U(\bar{u}) \subset \mathbb{R}^n$ of \bar{u} such that $\hat{J}(\bar{u}) \leq \hat{J}(u)$ for all $u \in U(\bar{u}) \cap \mathcal{U}$.

Definition 2.14 (Convex set). A set $\mathcal{U} \subset \mathbb{R}^n$ is said to be *convex* if for all $x, y \in \mathcal{U}$ and $\lambda \in (0, 1)$ it holds that

$$(\lambda x + (1 - \lambda)y) \in \mathcal{U}.$$

Definition 2.15 (Euclidean norm). For $x, y \in \mathbb{R}^n$ we define the standard *Euclidean scalar product* as

$$\langle x, y \rangle_2 := x^T y$$

and the *Euclidean norm* as

$$\|x\|_2 := \sqrt{\langle x, x \rangle_2}.$$

Definition 2.16 (Convex function). Let $\mathcal{U} \subset \mathbb{R}^n$ be a convex set and $\hat{J} : \mathcal{U} \rightarrow \mathbb{R}$.

(i) \hat{J} is called a *convex function* if for all $\lambda \in (0, 1)$ and $x, y \in \mathcal{U}$ it holds that

$$\hat{J}(\lambda x + (1 - \lambda)y) \leq \lambda \hat{J}(x) + (1 - \lambda) \hat{J}(y). \quad (2.2)$$

(ii) If in (2.2) instead of " \leq " the strict inequality holds for all $x, y \in \mathcal{U}$ with $x \neq y$ we call \hat{J} *strictly convex*.

(iii) If there even exists $\mu > 0$ such that for all $\lambda \in (0, 1)$ and $x, y \in \mathcal{U}$ it holds that

$$\hat{J}(\lambda x + (1 - \lambda)y) + \mu \lambda (1 - \lambda) \|y - x\|_2^2 \leq \lambda \hat{J}(x) + (1 - \lambda) \hat{J}(y)$$

we call \hat{J} μ -strongly convex.

Theorem 2.17. *Let $\mathcal{U} \subset \mathbb{R}^n$ be a convex set and $\hat{J} : \mathcal{U} \rightarrow \mathbb{R}$ be a differentiable function. Then it holds:*

(i) \hat{J} is convex if and only if for all $x, y \in \mathcal{U}$ it holds that

$$\nabla \hat{J}(x)^T (y - x) \leq \hat{J}(y) - \hat{J}(x).$$

(ii) \hat{J} is strictly convex if and only if for all $x, y \in \mathcal{U}$ with $x \neq y$ it holds that

$$\nabla \hat{J}(x)^T (y - x) < \hat{J}(y) - \hat{J}(x).$$

(iii) \hat{J} is μ -strongly convex if and only if for all $x, y \in \mathcal{U}$ it holds that

$$\nabla \hat{J}(x)^T (y - x) + \mu \|y - x\|_2^2 \leq \hat{J}(y) - \hat{J}(x). \quad (2.3)$$

Proof. See [UU11, Satz 6.3]. □

Lemma 2.18. *Let $f, g : \mathbb{R}^n \rightarrow \mathbb{R}$ be two continuous functions with $f(x) \leq g(x)$ for all $x \in \mathbb{R}^n$. Then it holds that*

$$\inf_{x \in \mathbb{R}^n} f(x) \leq \inf_{x \in \mathbb{R}^n} g(x).$$

Proof. Let $(x_k)_{k \in \mathbb{N}}$ be a sequence with $g(x_k) \rightarrow \inf_{x \in \mathbb{R}^n} g(x)$. Then

$$\inf_{x \in \mathbb{R}^n} f(x) \leq f(x_k) \leq g(x_k)$$

holds for all $k \in \mathbb{N}$. Letting k tend to infinity yields the claim. □

Theorem 2.19. *Let $\mathcal{U} \subset \mathbb{R}^n$ be a convex set, $\hat{J} : \mathcal{U} \rightarrow \mathbb{R}$ a differentiable, μ -strongly convex function with the minimizer $\bar{u} \in \mathcal{U}$. Then it holds that*

$$\|\nabla \hat{J}(x)\|_2^2 \geq 4\mu(\hat{J}(x) - \hat{J}(\bar{u}))$$

for all $x \in \mathcal{U}$.

Proof. Let $x, y \in \mathcal{U}$ and consider (2.3). We can minimize both sides of the inequality independently with respect to y . The inequality is maintained due to Lemma 2.18. Minimizing the right-hand side yields

$$\hat{J}(\bar{u}) - \hat{J}(x), \quad (2.4)$$

because \bar{u} is the unique minimizer of \hat{J} .

As the left hand side of (2.3) is a convex, differentiable function in y the first order optimality condition is sufficient, see [UU11, Satz 6.5] for instance. Thus, differentiating the left-hand side and setting it equal to zero gives us the minimizer

$$\bar{y} = x - \frac{1}{2\mu} \nabla \hat{J}(x).$$

Substituting this into y on the left-hand side of (2.3) and using (2.4) on the right-hand side leads to

$$\begin{aligned} \nabla \hat{J}(x)^T (\bar{y} - x) + \mu \|\bar{y} - x\|_2^2 &= -\frac{1}{2\mu} \nabla \hat{J}(x)^T \nabla \hat{J}(x) + \frac{\mu}{4\mu^2} \|\nabla \hat{J}(x)\|_2^2 \\ &= -\frac{1}{2\mu} \|\nabla \hat{J}(x)\|_2^2 + \frac{1}{4\mu} \|\nabla \hat{J}(x)\|_2^2 \\ &= -\frac{1}{4\mu} \|\nabla \hat{J}(x)\|_2^2 \\ &\leq \hat{J}(\bar{u}) - \hat{J}(x), \end{aligned}$$

which after rearranging proves the statement. \square

Definition 2.20 (Lipschitz continuous). Let $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a function. We say that f is *Lipschitz continuous* if there exists $L > 0$ such that

$$\|f(y) - f(x)\|_2 \leq L \|y - x\|_2$$

holds true for all $x, y \in \mathbb{R}^n$. We also call L the *Lipschitz constant*.

Lemma 2.21. Let $\mathcal{U} \subset \mathbb{R}^n$ be a convex set, $\hat{J} : \mathcal{U} \rightarrow \mathbb{R}$ be a convex and differentiable function and $\nabla \hat{J}$ be Lipschitz continuous with Lipschitz constant L . Then it holds for all $x, y \in \mathcal{U}$ that

$$\|\nabla \hat{J}(x) - \nabla \hat{J}(y)\|_2^2 \leq L(x - y)^T (\nabla \hat{J}(x) - \nabla \hat{J}(y)).$$

Proof. See [Zho18, Lemma 4]. \square

Definition 2.22 (Box-constrained optimization). Let $\mathcal{U} \subset \mathbb{R}^n$ be a convex set, $\hat{J} : \mathcal{U} \rightarrow \mathbb{R}$ be a continuous function and $u_{\min}, u_{\max} \in \mathbb{R}^n$ with $u_{\min} < u_{\max}$, where " $<$ " is understood component-wise. We define the bounded, closed and convex set

$$\mathcal{U}_{\text{ad}} := \{u \in \mathcal{U} \mid u_{\min} \leq u \leq u_{\max}\}. \quad (2.5)$$

Then we consider the so-called *box-constrained optimization problem*

$$\min \hat{J}(u) \quad \text{s.t.} \quad u \in \mathcal{U}_{\text{ad}}. \quad (\mathbf{P})$$

Note that if \hat{J} is continuous and \mathcal{U}_{ad} is compact, there exists a global solution to (\mathbf{P}) , see [DR11, Folgerung 10.9] for instance.

Theorem 2.23 (First order optimality). *Consider (\mathbf{P}) with a continuous function $\hat{J} : \mathbb{R}^n \rightarrow \mathbb{R}$. Let \bar{u} be a local minimizer of \hat{J} and \hat{J} be differentiable in \bar{u} . Then the inequality*

$$\langle \nabla \hat{J}(\bar{u}), u - \bar{u} \rangle_2 = \nabla \hat{J}(\bar{u})^T (u - \bar{u}) \geq 0 \quad (2.6)$$

holds for all $u \in \mathcal{U}_{\text{ad}}$.

Proof. See [Vol19, Satz 15.2]. □

Definition 2.24 (Projection on closed and convex subsets). Let \mathcal{U}_{ad} be a closed and convex subset of \mathbb{R}^n . Then we define the *projection* $\mathcal{P} : \mathbb{R}^n \rightarrow \mathcal{U}_{\text{ad}}$ by

$$\mathcal{P}(x) = \arg \min_{u \in \mathcal{U}_{\text{ad}}} \|u - x\|_2. \quad (2.7)$$

In case that \mathcal{U}_{ad} is of the form (2.5) the projection \mathcal{P} is given as

$$\mathcal{P}(x)_i = \begin{cases} x_i, & \text{for } x_i \in ((u_{\min})_i, (u_{\max})_i), \\ u_{\min}, & \text{for } x_i \leq u_{\min}, \\ u_{\max}, & \text{for } x_i \geq u_{\max}, \end{cases}$$

for $i \in \{1, \dots, n\}$.

Theorem 2.25 (Non-expansive projection). *Let \mathcal{U}_{ad} be a closed and convex subset of \mathbb{R}^n and \mathcal{P} the projection given by (2.7). Then for all $x, y \in \mathbb{R}^n$ it holds that*

$$\|\mathcal{P}(x) - \mathcal{P}(y)\|_2 \leq \|x - y\|_2.$$

Proof. See [Sin20, Proposition 7.1]. □

Definition 2.26 (\mathcal{O} -notation). Let $(d_n)_{n \in \mathbb{N}}$ and $(e_n)_{n \in \mathbb{N}}$ be sequences in \mathbb{R} . We say that $d_n = \mathcal{O}(e_n)$ for $n \rightarrow \infty$ if there exists a constant $C > 0$ and $K \in \mathbb{N}$ such that

$$|d_k| \leq C|e_k| \quad \text{for all } k \geq K.$$

Definition 2.27 (Convergence Rate). Let $(u_n)_{n \in \mathbb{N}} \subset \mathbb{R}^n$ and $(e_n)_{n \in \mathbb{N}} \subset \mathbb{R}$. We say that $(u_n)_{n \in \mathbb{N}}$ converges to $\bar{u} \in \mathbb{R}^n$ with respect to a norm $\|\cdot\|$ with *convergence rate* $\mathcal{O}(e_n)$ if

$$\|u_n - \bar{u}\| = \mathcal{O}(e_n) \quad (n \rightarrow \infty).$$

Probability theory

In this section, the relevant definitions and notations of probability theory for this thesis are covered. More details can be found in any introductory book on probability theory such as [Rüs16], for instance.

Definition 2.28 (Random variable). Let (Ω, \mathcal{F}) be a measure space. A function $X : \Omega \mapsto \mathbb{R} \cup \{-\infty, \infty\}$ is called *random variable* if X is measurable.

Analogous we call $X = (X_1, \dots, X_n)$ a *random vector* if X_1, \dots, X_n are random variables.

Definition 2.29 (Generated σ -algebra). Let Ω be a set, $X : \Omega \rightarrow \mathbb{R}^m$ be a random vector. We then call the smallest σ -algebra for which X is measurable, defined by

$$\sigma(X) := \{X^{-1}(B) : B \in \mathcal{B}(\mathbb{R}^m)\},$$

the of X *generated σ -algebra*.

Analogous for random vectors X^1, \dots, X^k we define the smallest σ such that all of the random vectors are measurable by

$$\sigma(X^1, \dots, X^k) := \sigma\left(\bigcup_{i=1}^k \sigma(X^i)\right).$$

Definition 2.30 (Independence). Let \mathcal{F} be a σ -algebra, \mathbb{P} a probability measure, I an index set and $\mathcal{E}_i \subset \mathcal{F}$ for all $i \in I$. Then we say that $(\mathcal{E}_i)_{i \in I}$ are *independent* if for every finite subset $J \subset I$ and every choice of $E_j \in \mathcal{E}_j$ ($j \in J$) it holds that

$$\mathbb{P}\left(\bigcap_{j \in J} E_j\right) = \prod_{j \in J} \mathbb{P}(E_j).$$

We say that X^1, \dots, X^m are *independent random vectors* if the generated σ -algebras $\sigma(X^1), \dots, \sigma(X^m)$ are independent.

Definition 2.31 (Expected Value). Let $X \in L^1(\Omega)$ be a random variable on a

probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Then

$$\mathbb{E}[X] := \int_{\Omega} X(\omega) d\mathbb{P}(\omega)$$

is called the *expected value* of X .

For a random vector $X = (X_1, \dots, X_n)$ we define the expected value component-wise, i.e. $\mathbb{E}[X] = (\mathbb{E}[X_1], \dots, \mathbb{E}[X_n])$.

Theorem 2.32 (Jensen's inequality). *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a convex function and $X \in L^1(\Omega)$. Then*

$$f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$$

holds true.

Proof. See [Kle13, Theorem 7.9]. □

Definition 2.33 (Conditional Expected Value). Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, $\mathcal{G} \subset \mathcal{F}$ be a σ -algebra and $X \in L^1(\Omega)$. Then we call $Y \in L^1(\Omega)$ the *conditional expectation* or *conditional expected value* of X given \mathcal{G} if Y is measurable with respect to \mathcal{G} and for every $A \in \mathcal{G}$ it holds that

$$\int_A X d\mathbb{P} = \int_A Y d\mathbb{P}.$$

We will then write $\mathbb{E}[X|\mathcal{G}] = Y$. Analogously for $Z \in L^1(\Omega)$, we will denote $\mathbb{E}[X|Z] := \mathbb{E}[X|\sigma(Z)]$.

Theorem 2.34. *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and $X \in L^1(\Omega)$.*

(i) *Let $\mathcal{G} \subset \mathcal{F}$ be a σ -algebra and X be independent of \mathcal{G} . Then it holds that*

$$\mathbb{E}[X|\mathcal{G}] = \mathbb{E}[X].$$

(ii) *Let $\mathcal{F}_1, \mathcal{F}_2 \subset \mathcal{F}$ be two sub- σ -algebras with $\mathcal{F}_1 \subset \mathcal{F}_2$. Then it holds that*

$$\mathbb{E}[\mathbb{E}[X|\mathcal{F}_1]|\mathcal{F}_2] = \mathbb{E}[\mathbb{E}[X|\mathcal{F}_2]|\mathcal{F}_1] = \mathbb{E}[X|\mathcal{F}_1].$$

(iii) *Let $\mathcal{G} \subset \mathcal{F}$ be a σ -algebra, $X : \Omega \rightarrow \mathbb{R}^m$ be \mathcal{G} measurable and $f : \mathbb{R}^m \rightarrow \mathbb{R}^m$ be a measurable function. Then it holds that*

$$\mathbb{E}[f(X)|\mathcal{G}] = f(X).$$

Proof. See [Bor17]. □

Definition 2.35 (Variance). Let $X \in L^2(\Omega)$ be a random variable on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. We define the *variance* of X as

$$\text{Var}(X) := \mathbb{E}[(X - \mathbb{E}[X])^2].$$

We will also call $\sigma(X) := \sqrt{\text{Var}(X)}$ the *volatility* or *standard deviation* of X and $\sigma(X)/\mathbb{E}[X]$ the *coefficient of variation* of X .

Definition 2.36 (Covariance). Let $X, Y \in L^2(\Omega)$ be random variables on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Then

$$\text{Cov}(X, Y) := \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$$

is called the *covariance* of X and Y .

Definition 2.37 (Correlation). Let $X, Y \in L^2(\Omega)$ be two random variables on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. We call

$$\text{Cor}(X, Y) := \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)}\sqrt{\text{Var}(Y)}}$$

correlation of X and Y . If $\text{Cor}(X, Y) = 0$ we say that X and Y are *uncorrelated*.

Definition 2.38 (Normal Distributed Random Variable). Let $\mu \in \mathbb{R}$ and $\sigma^2 > 0$. Then we will call a random variable X *normally distributed* with parameters μ and σ^2 if

$$\mathbb{P}(X \leq z) = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^z \exp\left(-\frac{(t - \mu)^2}{2\sigma^2}\right) dt$$

holds true for all $z \in \mathbb{R}$.

Theorem 2.39. *Let X be normally distributed with parameters μ and σ^2 . Then it holds that*

$$\begin{aligned} \mathbb{E}[X] &= \mu, \\ \text{Var}(X) &= \sigma^2. \end{aligned}$$

Proof. See [Kle13]. □

Theorem 2.40. *Let X and Y be normal distributed random variables. Then X and Y are independent if and only if they are uncorrelated.*

Proof. See [Bor17, Proposition 8.1]. □

Linear Algebra

This section covers some Linear Algebra basics, which in more detail can be found in [Sai17], for instance. In the following let $A \in \mathbb{R}^{n \times n}$ be a matrix.

Definition 2.41 (Eigenvalue). Let $\lambda \in \mathbb{C}$ and $v \in \mathbb{R}^n \setminus \{0\}$ be such that

$$Av = \lambda v.$$

Then we call λ an *eigenvalue* and v an *eigenvector* of A .

Definition 2.42 (Definiteness). We say that A is

- *positive semi-definite* if all eigenvalues of A are non-negative,
- *positive definite* if all eigenvalues of A are positive.

Theorem 2.43. A is positive definite if and only if it holds for all $x \in \mathbb{R}^n \setminus \{0\}$ that

$$x^T Ax > 0.$$

Moreover A is positive semi-definite if and only if

$$x^T Ax \geq 0$$

holds true for all $x \in \mathbb{R}^n$.

Proof. See [Sai17, Theorem 8.2.1]. □

Theorem 2.44. Let $I \in \mathbb{R}^{n \times n}$ be the identity matrix, $\mu \in \mathbb{R}$ and λ an eigenvalue of A . Then $\lambda - \mu$ is an eigenvalue of $A - \mu I$.

Proof. Let v be the corresponding eigenvector to λ . Then it holds that

$$(A - \mu I)v = Av - \mu v = (\lambda - \mu)v,$$

such that $\lambda - \mu$ is an eigenvalue to $A - \mu I$ with the eigenvector v . □

Chapter 3

Problem Formulation

Let $\mathcal{D} \subset \mathbb{R}^d$ be an open, bounded set with Lipschitz continuous boundary $\partial\mathcal{D}$.

The control space is given by $\mathcal{U} = \mathbb{R}^{n_u}$ and admissible control vectors belong to the closed, convex and bounded set

$$\mathcal{U}_{\text{ad}} = \{u = (u_i)_{1 \leq i \leq n_u} \in \mathcal{U} \mid u_{\mathbf{a}} \leq u \leq u_{\mathbf{b}} \in \mathbb{R}^{n_u}\},$$

where " \leq " is understood component-wise in \mathbb{R}^{n_u} .

We suppose that $(\Omega, \mathcal{F}, \mathbb{P})$ is a probability space, where Ω represents the sample space, $\mathcal{F} \subset 2^\Omega$ is a σ -algebra on Ω , $\mathbb{P} : \Omega \rightarrow [0, 1]$ is a probability measure and $\xi : \Omega \rightarrow \Xi \subset \mathbb{R}^m$ is a random vector.

Let us set $V = H_0^1(\mathcal{D})$. For a control vector $u \in \mathcal{U}_{\text{ad}}$ and a given $\xi \in \Xi$ let $y = y_u(\cdot; \xi) \in V$ be the weak solution to the linear elliptic partial differential equation (PDE)

$$-\nabla \cdot (a(\mathbf{x}; \xi(\omega)) \nabla y_u(\mathbf{x}; \xi(\omega))) = f(\mathbf{x}; \xi(\omega)) + \sum_{i=1}^{n_u} u_i b_i(\mathbf{x}), (\mathbf{x}, \omega) \in \mathcal{D} \times \Omega, \quad (3.1a)$$

$$y_u(\mathbf{x}; \xi(\omega)) = 0, \quad (\mathbf{x}, \omega) \in \partial\mathcal{D} \times \Omega. \quad (3.1b)$$

The functions a and f are assumed to have the following affine representation

$$a(\mathbf{x}; \xi(\omega)) = \sum_{i=1}^{N_a} \vartheta_i^a(\xi(\omega)) a_i(\mathbf{x}) \quad (\mathbf{x}, \omega) \in \mathcal{D} \times \Omega, \quad (3.2a)$$

$$f(\mathbf{x}; \xi(\omega)) = \sum_{i=1}^{N_f} \vartheta_i^f(\xi(\omega)) f_i(\mathbf{x}) \quad (\mathbf{x}, \omega) \in \mathcal{D} \times \Omega, \quad (3.2b)$$

where a_i, f_i are functions in \mathbf{x} and $\vartheta_i^a, \vartheta_i^f$ are functions in ω respectively.

In (3.2) we assume that the coefficient functions $\vartheta_1^a, \dots, \vartheta_{N_a}^a$ are Lipschitz continuous and that there exist positive constants $0 < a_{\min} \leq a_{\max}$ such that

$$a_{\min} \leq a(\mathbf{x}; \xi(\omega)) \leq a_{\max} \quad \text{for all } \mathbf{x} \in \mathcal{D} \text{ and } \omega \in \Omega \text{ almost sure (a.s.).} \quad (3.3)$$

Furthermore, the function f satisfies

$$\int_{\Omega} \int_{\mathcal{D}} |f(\mathbf{x}; \xi(\omega))|^2 d\mathbf{x} d\mathbb{P}(\omega) < \infty.$$

The control shape functions b_1, \dots, b_{n_u} belong to $H = L^2(\mathcal{D})$ endowed by the standard inner product and its induced norm. We equip V by the inner product

$$\langle \varphi, \psi \rangle_V = \int_{\mathcal{D}} \nabla \varphi(\mathbf{x}) \cdot \nabla \psi(\mathbf{x}) d\mathbf{x} \quad \text{for } \varphi, \psi \in V$$

and the induced norm $\|\cdot\|_V = \sqrt{\langle \cdot, \cdot \rangle_V}$.

It follows directly from the Lax-Milgram lemma [Dzi10, Lemma 4.3] that for every $u \in \mathcal{U}_{\text{ad}}$ and for $\omega \in \Omega$ a.s. there exists a unique weak solution $y = y_u(\cdot; \xi(\cdot)) \in V$ to (3.1). For a more detailed derivation of this, see [Sin20, Chapter 4] for instance. More precisely, y satisfies

$$\int_{\mathcal{D}} a(\mathbf{x}; \xi(\omega)) \nabla y(\mathbf{x}; \xi(\omega)) \cdot \nabla \varphi(\mathbf{x}) d\mathbf{x} = \int_{\mathcal{D}} \left(f(\mathbf{x}; \xi(\omega)) + \sum_{i=1}^{n_u} u_i b_i(\mathbf{x}) \right) \varphi(\mathbf{x}) d\mathbf{x}$$

for all $\varphi \in V$ and $\omega \in \Omega$ a.s. Using the linearity of the integral and (3.2) we obtain the equivalent formulation

$$\begin{aligned} & \sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) \int_{\mathcal{D}} a_k(\mathbf{x}) \nabla y(\mathbf{x}; \xi(\omega)) \cdot \nabla \varphi(\mathbf{x}) d\mathbf{x} \\ &= \sum_{k=1}^{N_f} \vartheta_k^f(\xi(\omega)) \int_{\mathcal{D}} f_k(\mathbf{x}) \varphi(\mathbf{x}) d\mathbf{x} + \sum_{i=1}^{n_u} u_i \int_{\mathcal{D}} b_i(\mathbf{x}) \varphi(\mathbf{x}) d\mathbf{x} \end{aligned} \quad (3.4)$$

for all $\varphi \in V$ and all $\omega \in \Omega$ a.s. or

$$\sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) \langle a_k \nabla y(\cdot; \xi(\omega)), \nabla \varphi \rangle_{H^d} = \sum_{k=1}^{N_f} \vartheta_k^f(\xi(\omega)) \langle f_k, \varphi \rangle_H + \sum_{i=1}^{n_u} u_i \langle b_i, \varphi \rangle_H$$

for all $\varphi \in V$ and all $\omega \in \Omega$ a.s.

Next we introduce the objective function. For given desired state $y^d \in H = L^2(\mathcal{D})$

and regularization parameter $\lambda \geq 0$ we define

$$J(u, \xi(\omega)) := \frac{1}{2} \left(\|y_u(\cdot; \xi(\cdot)) - y^d\|_H^2 + \lambda \left\| \sum_{i=1}^{n_u} u_i b_i \right\|_H^2 \right) \quad \text{for } u \in \mathcal{U}_{\text{ad}}$$

and

$$\hat{J}(u) := \mathbb{E}[J(u; \xi)] = \frac{1}{2} \mathbb{E}[\|y_u(\cdot; \xi(\cdot)) - y^d\|_H^2] + \frac{\lambda}{2} \left\| \sum_{i=1}^{n_u} u_i b_i \right\|_H^2 \quad \text{for } u \in \mathcal{U}_{\text{ad}}$$

where

$$\mathbb{E}[\|y_u(\cdot; \xi(\cdot)) - y^d\|_H^2] = \int_{\Omega} \int_{\mathcal{D}} |y_u(\mathbf{x}; \xi(\omega)) - y^d(\mathbf{x})|^2 d\mathbf{x} d\mathbb{P}(\omega)$$

and

$$\left\| \sum_{i=1}^{n_u} u_i b_i \right\|_H^2 = \sum_{i=1}^{n_u} \sum_{j=1}^{n_u} u_i u_j \underbrace{\int_{\mathcal{D}} b_i(\mathbf{x}) b_j(\mathbf{x}) d\mathbf{x}}_{=: W_{ij}} = u^T W u \quad (3.5)$$

with $W = (W_{ij})_{1 \leq i, j \leq n_u}$.

Now we can formulate the PDE constrained optimization problem

$$\min \hat{J}(u) \quad \text{subject to } u \in \mathcal{U}_{\text{ad}}. \quad (\hat{\mathbf{P}})$$

In $(\hat{\mathbf{P}})$ the control variable u is deterministic, while the state variable $y_u(\cdot; \xi(\omega))$ is a random variable depending on the realization $\omega \in \Omega$.

Theorem 3.1. *There exists an optimal solution $\bar{u} \in \mathcal{U}_{\text{ad}}$ to $(\hat{\mathbf{P}})$. Uniqueness follows in the case $\lambda > 0$.*

Proof. The existence is proven in [GP19, Lemma 4.2]. For $\lambda > 0$ it holds that \hat{J} is λ -strongly convex and therefore also strictly convex, see [Sin20, Corollary 4.1]. Thus the uniqueness of \bar{u} follows. \square

Now we show that J and \hat{J} are almost surely differentiable and give an explicit formula for the gradients.

Theorem 3.2. *The function \hat{J} is differentiable with*

$$\begin{aligned} \nabla_u J(u, \xi(\omega))h &= -\langle y_h(\cdot; \xi(\omega)), y^d - y_u(\cdot; \xi(\omega)) \rangle_H + \lambda h^T W u, \\ \nabla \hat{J}(u)h &= \mathbb{E}[\nabla_u J(u, \xi(\cdot))h] \end{aligned}$$

for $h \in \mathcal{U}$ where $y_h(\cdot; \xi(\omega))$ is the weak solution to the linearized state equation (3.1)

$$-\nabla \cdot (a(\mathbf{x}; \xi(\omega)) \nabla y_h(\mathbf{x}; \xi(\omega))) = \sum_{i=1}^{n_u} h_i b_i(\mathbf{x}), \quad (\mathbf{x}, \omega) \in \mathcal{D} \times \Omega, \quad (3.6a)$$

$$y_h(\mathbf{x}; \xi(\omega)) = 0, \quad (\mathbf{x}, \omega) \in \partial \mathcal{D} \times \Omega, \quad (3.6b)$$

i.e. $y_h(\cdot; \xi(\omega))$ solves

$$\langle a(\cdot; \xi(\omega)) \nabla y_h(\cdot; \xi(\omega)), \nabla \varphi \rangle_{L^2(\mathcal{D})^d} = \sum_{i=1}^{n_u} h_i \langle b_i, \varphi \rangle_H \quad \forall \varphi \in V, \omega \in \Omega. \quad (3.7)$$

Proof. See [Sin20, Proposition 4.1]. □

To obtain a more suitable representation of the gradient of \hat{J} we have a look at the corresponding adjoint equation

$$-\nabla(a(\mathbf{x}; \xi(\omega)) \nabla p(\mathbf{x}; \xi(\omega))) = y^d(\mathbf{x}) - y_u(\mathbf{x}; \xi(\omega)), \quad (x, \omega) \in \mathcal{D} \times \Omega, \quad (3.8a)$$

$$p(\mathbf{x}; \xi(\omega)) = 0, \quad (x, \omega) \in \partial \mathcal{D} \times \Omega, \quad (3.8b)$$

which can be found in [GP19].

As with the state equation the existence of a unique weak solution $p_u(\cdot; \xi(\cdot)) \in V$ to (3.8) for every $u \in \mathcal{U}_{\text{ad}}$ and $\omega \in \Omega$ a.s. is ensured by the Lax-Milgram lemma [Dzi10, Lemma 4.3]. Precisely $p = p_u(\cdot; \xi(\cdot))$ solves

$$\begin{aligned} & \sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) \int_{\mathcal{D}} a_k(\mathbf{x}) \nabla p(\mathbf{x}; \xi(\omega)) \cdot \nabla \varphi(\mathbf{x}) d\mathbf{x} \\ &= \int_{\mathcal{D}} (y^d(\mathbf{x}) - y_u(\mathbf{x}; \xi(\omega))) \varphi(\mathbf{x}) d\mathbf{x} \end{aligned} \quad (3.9)$$

or equivalently

$$\sum_{k=1}^{N_a} \vartheta_k^a(\omega) \langle a_k \nabla p(\cdot; \xi(\omega)), \nabla \varphi \rangle_{L^2(\mathcal{D})^d} = \langle y^d - y_u(\cdot; \xi(\omega)), \varphi \rangle_H \quad (3.10)$$

for all $\varphi \in V$ and $\omega \in \Omega$ a.s.

We can now formulate another representation of the gradient of \hat{J} .

Theorem 3.3. *Let $p_u(\cdot; \xi(\omega)) \in V$ be the weak solution for $\omega \in \Omega$ to the adjoint*

equation (3.8). Then the gradient of \hat{J} can be represented as

$$\nabla \hat{J}(u)h = \lambda h^T W u - \sum_{i=1}^{n_u} h_i \mathbb{E}[\langle b_i, p_u(\cdot; \xi(\cdot)) \rangle_H]$$

or without the direction h as

$$\nabla \hat{J}(u) = \lambda W u - \left(\mathbb{E}[\langle b_i, p_u(\cdot; \xi(\cdot)) \rangle_H] \right)_{1 \leq i \leq n_u} \quad (3.11)$$

for $u \in \mathcal{U}_{\text{ad}}$.

Proof. Let $\omega \in \Omega$ and $y_h(\cdot; \xi(\omega))$ be the weak solution to the linearized state equation (3.6). We further denote $a_\omega := a(\cdot; \xi(\omega))$.

As $y_h(\cdot, \xi(\omega)), p_u(\cdot, \xi(\omega)) \in V$ it holds, by using (3.7) and (3.10), that

$$\begin{aligned} \langle y_h(\cdot; \xi(\omega)), y^{\text{d}} - y_u(\cdot; \xi(\omega)) \rangle_H &= \langle a_\omega \nabla p_u(\cdot; \xi(\omega)), \nabla y_h(\cdot; \xi(\omega)) \rangle_{L^2(\mathcal{D})^d} \\ &= \langle a_\omega \nabla y_h(\cdot; \xi(\omega)), \nabla p_u(\cdot; \xi(\omega)) \rangle_{L^2(\mathcal{D})^d} \\ &= \sum_{i=1}^{n_u} h_i \langle b_i, p_u(\cdot; \xi(\omega)) \rangle_H. \end{aligned}$$

Therefore the claimed representation follows with Theorem 3.2. \square

We further state properties of \hat{J} which will be necessary for convergence analysis later on.

Theorem 3.4. *The gradient $\nabla \hat{J}$ is Lipschitz continuous.*

Proof. See [Sin20, Proposition 4.4]. \square

Theorem 3.5. *\hat{J} is two times continuously differentiable. Further, for $\lambda > 0$, there exist constants $L > 0$ and $\mu > 0$ such that with I being the identity matrix it holds that*

$$LI - \nabla^2 \hat{J}(u) \quad \text{and} \quad \nabla^2 \hat{J}(u) - \mu I \quad (3.12)$$

are symmetric and positive semi-definite for all $u \in \mathcal{U}_{\text{ad}}$.

Proof. Let $h_1, h_2 \in \mathcal{U}$ and $\omega \in \Omega$ be fixed. For the rest of the proof we will omit ω and denote $J(u) = J(u, \xi(\omega))$ and analogously $y_u(x) = y_u(x; \xi(\omega))$. With Theorem 3.2 and the linearity of the solution of the linearized state equation y_h in h , see

[Sin20, Theorem 4.1] for instance, we get that

$$\nabla J(u + h_2)h_1 - \nabla J(u)h_1 = \lambda h_1^T W h_2 + \langle y_{h_1}, y_{h_2} \rangle_H.$$

By defining the bilinear form

$$\nabla^2 J(u)(h_1, h_2) := \nabla(\nabla J(u)h_1)h_2 := \lambda h_1^T W h_2 + \langle y_{h_1}, y_{h_2} \rangle_H \quad (3.13)$$

it holds that

$$\lim_{\|h_2\|_2 \rightarrow 0} \frac{|\nabla J(u + h_2)h_1 - \nabla J(u)h_1 - \nabla^2 J(u)(h_1, h_2)|}{\|h_2\|_2} = 0,$$

such that J is two times differentiable and its second derivative is given by (3.13).

As $\nabla^2 J$ does not depend on u it is constant and therefore also continuous.

By definition, W (see page 17) is a symmetric matrix. As therefore W and the scalar product are symmetric it follows that $\nabla^2 J$ is symmetric. Thus the symmetries of (3.12) are easy to see.

Further we get by (3.5) that W is positive semidefinite. Now let $s \in \mathcal{U} \setminus \{0\}$. Then it holds that $\|y_s\|_H > 0$ as the right side of the PDE is unequal to zero and therefore y_s cannot be zero almost everywhere in \mathcal{D} .

Thus

$$\nabla^2 J(u)(s, s) = \lambda s^T W s + \langle y_s, y_s \rangle_H > 0$$

holds true so that $\nabla^2 J(u)$ is positive definite.

Now let $\pi_{min} > 0$ be the minimal eigenvalue of $\nabla^2 J(u)$. Choose $\mu = \pi_{min}/2$ then the smallest eigenvalue of $\nabla^2 \hat{J}(u) - \mu I$ is $\pi_{min}/2$ such that $\nabla^2 \hat{J}(u) - \mu I$ is positive definite.

Analogously, let $\pi_{max} > 0$ be the maximal eigenvalue of $\nabla^2 J(u)$ and choose $L = 2\pi_{max}$ then the smallest eigenvalue of $LI - \nabla^2 \hat{J}(u)$ is π_{max} such that $LI - \nabla^2 \hat{J}(u)$ is positive definite. As $\nabla^2 \hat{J}(u)(h_1, h_2) = \mathbb{E}[\nabla^2 J(u)(h_1, h_2)]$ these properties also hold for \hat{J} . \square

Notice that (3.12) implies that \hat{J} is μ -strongly convex.

Since the cost functional \hat{J} is continuously differentiable and $(\hat{\mathbf{P}})$ is a convex optimization problem, we can characterize an optimal solution \bar{u} to $(\hat{\mathbf{P}})$ by first-order sufficient optimality conditions.

3.1 Finite Element Method

In this section, we will describe how the Finite Element (FE) Method and the Galerkin approach can be applied on the weak formulation (3.4). A full description of the FE Method and the Galerkin approach can be found in [Dzi10], for instance. Let therefore $\omega \in \Omega$ be fixed and, for brevity, omitted as an argument of y and p . The function space V is replaced by a finite-dimensional subspace

$$V^N = \text{span}\{\varphi_1, \dots, \varphi_N\} \subset V$$

endowed by the V -topology. First, we discretize (3.4) by choosing the test functions $\varphi = \varphi_i$ for $i \in \{1, \dots, N\}$:

$$\begin{aligned} & \sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) \sum_{j=1}^N \int_{\mathcal{D}} a_k(\mathbf{x}) \nabla y(\mathbf{x}) \cdot \nabla \varphi_i(\mathbf{x}) d\mathbf{x} \\ &= \sum_{k=1}^{N_f} \vartheta_k^f(\xi(\omega)) \int_{\mathcal{D}} f_k(\mathbf{x}) \varphi_i(\mathbf{x}) d\mathbf{x} + \sum_{j=1}^{n_u} u_j \int_{\mathcal{D}} b_j(\mathbf{x}) \varphi_i(\mathbf{x}) d\mathbf{x}. \end{aligned} \quad (3.14)$$

Further, we suppose that the discrete FE solution belongs to the space V^N , i.e.

$$y(\mathbf{x}) \approx y^N(\mathbf{x}) = \sum_{j=1}^N y_j \varphi_j(\mathbf{x}) \in V^N. \quad (3.15)$$

Inserting (3.15) into (3.14) we find

$$\begin{aligned} & \sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) \sum_{j=1}^N \underbrace{\left[\int_{\mathcal{D}} a_k(\mathbf{x}) \nabla \varphi_j(\mathbf{x}) \cdot \nabla \varphi_i(\mathbf{x}) d\mathbf{x} \right]}_{=: A_{ij}^k} y_j \\ &= \sum_{k=1}^{N_f} \vartheta_k^f(\xi(\omega)) \underbrace{\int_{\mathcal{D}} f_k(\mathbf{x}) \varphi_i(\mathbf{x}) d\mathbf{x}}_{=: f_i^k} + \sum_{j=1}^{n_u} u_j \underbrace{\int_{\mathcal{D}} b_j(\mathbf{x}) \varphi_i(\mathbf{x}) d\mathbf{x}}_{=: B_{ij}} \end{aligned} \quad (3.16)$$

for $i \in \{1, \dots, N\}$. Thus, (3.16) can be written as a linear system using $A^k = (A_{ij}^k)_{1 \leq i, j \leq N}$, $f^k = (f_i^k)_{1 \leq i \leq N}$, $B^k = (B_{ij}^k)_{1 \leq i, j \leq N}$, $y = (y_1, \dots, y_N)^T$ and $u = (u_1, \dots, u_{n_u})^T$:

$$\left[\sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) A^k \right] y = \sum_{k=1}^{N_f} \vartheta_k^f(\xi(\omega)) f^k + Bu. \quad (3.17)$$

Notice that neither A^k nor f^k depend on ω and can therefore be precomputed before the optimization. This reduces computational effort as the optimization involves solving (3.17) for various ω .

Similarly, for the adjoint equation, one uses

$$p(\mathbf{x}) \approx p^N(\mathbf{x}) = \sum_{j=1}^N p_j \varphi_j(\mathbf{x}) \in V^N$$

such that (3.9) can be discretized as

$$\begin{aligned} & \sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) \sum_{j=1}^N \left[\int_{\mathcal{D}} a_k(\mathbf{x}) \nabla \varphi_j(\mathbf{x}) \cdot \nabla \varphi_i(\mathbf{x}) \, d\mathbf{x} \right] p_j \\ &= \int_{\mathcal{D}} (y^d(\mathbf{x}) - y_u(\mathbf{x}; \xi(\omega))) \varphi_i(\mathbf{x}) \, d\mathbf{x} \\ &= \underbrace{\int_{\mathcal{D}} y^d \varphi_i \, d\mathbf{x}}_{=: Y_i^d} - \sum_{j=1}^N y_{u,j} \underbrace{\int_{\mathcal{D}} \varphi_j \varphi_i \, d\mathbf{x}}_{=: M_{ij}} \end{aligned}$$

for $i \in \{1, \dots, N\}$. With $M = (M_{ij})_{1 \leq i, j \leq N}$ and $Y^d = (Y_i^d)_{1 \leq i \leq N}$ this can be written as

$$\left[\sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) A^k \right] p = Y^d - M y_u,$$

where again neither Y^d nor M depend on ω .

Example 3.6. The affine linear representation of (3.2) is often not given but has to be obtained by approximating the underlying not affine linear functions. For simplicity, we denote $\xi(\omega)$ as ξ in this example. We further use $\mathcal{D} = (0, 1)^2$. Let

$$\begin{aligned} a^c(\mathbf{x}; \xi) &= 6 + \sin(\pi x_1 \xi_1) \sin(\pi x_2 \xi_2) \\ f^c(\mathbf{x}; \xi) &= \cos(\pi x_1 \xi_3) \exp(\pi x_2 \xi_4). \end{aligned}$$

be the underlying functions. A simple approach to obtain the form of (3.2) is to use the Taylor-Approximation. In this example, we just use the multidimensional first-

order Taylor-Approximation around $s := (0.5, 0.5)^T$ in x on a^c and f^c respectively:

$$\begin{aligned}
a(\mathbf{x}; \xi) &= a^c(s; \xi) + \nabla_x a^c(s; \xi)^T (\mathbf{x} - s) \\
&= \underbrace{(6 + \sin(0.5\pi\xi_1) \sin(0.5\pi\xi_2))}_{\vartheta_1^a(\xi)} \cdot \underbrace{1}_{a_1(\mathbf{x})} + \underbrace{\pi\xi_1 \cos(0.5\pi\xi_1) \sin(0.5\pi\xi_2)}_{\vartheta_2^a(\xi)} \underbrace{(x_1 - 0.5)}_{a_2(\mathbf{x})} \\
&\quad + \underbrace{\pi\xi_2 \cos(0.5\pi\xi_2) \sin(0.5\pi\xi_1)}_{\vartheta_3^a(\xi)} \underbrace{(x_2 - 0.5)}_{a_3(\mathbf{x})}, \\
f(\mathbf{x}; \xi) &= \underbrace{\cos(0.5\pi\xi_3) \exp(0.5\pi\xi_4)}_{\vartheta_1^f(\xi)} \cdot \underbrace{1}_{f_4(\mathbf{x})} + \underbrace{(-\pi\xi_3 \sin(0.5\pi\xi_3) \exp(0.5\pi\xi_4))}_{\vartheta_2^f(\xi)} \underbrace{(x_1 - 0.5)}_{f_2(\mathbf{x})} \\
&\quad + \underbrace{\pi\xi_4 \cos(0.5\pi\xi_3) \exp(0.5\pi\xi_4)}_{\vartheta_3^f(\xi)} \underbrace{(x_2 - 0.5)}_{f_3(\mathbf{x})}.
\end{aligned}$$

To illustrate the numerical advantage of this functional form, which enables us to preassemble the matrix A^k and the vector f^k of (3.16) and avoid recomputing them for every $\omega \in \Omega$, the state equation (3.17) is solved for 100 different realizations of ξ . Therefore, we use 100 realizations of a normally distributed random vector $\xi : \Omega \rightarrow \mathbb{R}^4$, which is truncated between 0.5 and 3.5 with an expected value of two and a standard deviation of 0.5 in each component. The random variables $\xi_i : \Omega \rightarrow \mathbb{R}$ of each random vector ξ are generated mutually uncorrelated and therefore independent, see Theorem 2.40. The truncation ensures that (3.3) is fulfilled. These bounds are not very restrictive as about 99.7% of the values of a normal distribution lie within three standard deviations around the mean what can be easily calculated with the properties of the normal distribution.

In comparison to that, we solve the state equation with the function a^c and f^c i.e solve

$$\begin{aligned}
&\sum_{j=1}^N \int_{\mathcal{D}} a^c(\mathbf{x}; \xi) \nabla \varphi_j(\mathbf{x}) \nabla \varphi_i(\mathbf{x}) d\mathbf{x} y_j \\
&= \int_{\mathcal{D}} f^c(\mathbf{x}; \xi) \varphi_i(\mathbf{x}) d\mathbf{x} + \sum_{j=1}^{n_u} u_j \int_{\mathcal{D}} b_j(\mathbf{x}) \varphi_i(\mathbf{x}) d\mathbf{x} \quad \text{for } i \in \{1, \dots, N\}
\end{aligned}$$

where all matrices, except of B , have to be reassembled for every realization.

The FE discretization uses piecewise linear elements. This results, when 961 vertices are used, in a triangulation as in Figure 3.1.

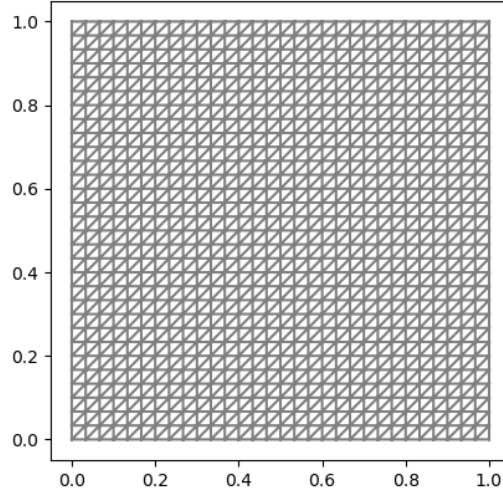


Figure 3.1: FE triangulation with piecewise linear elements and 961 vertices.

Simulations were run on FEniCS [Aln+15] on a laptop with Intel Core i7-8650u Processor with 16 GB RAM.

We start by having a look at the solutions y_{cont}^N and y_{app}^N of both settings for fixed realizations ξ . In Figure 3.2 the computed states of both settings for $\xi_1 \approx (1.15, 2.27, 1.26, 0.70)$ are displayed.

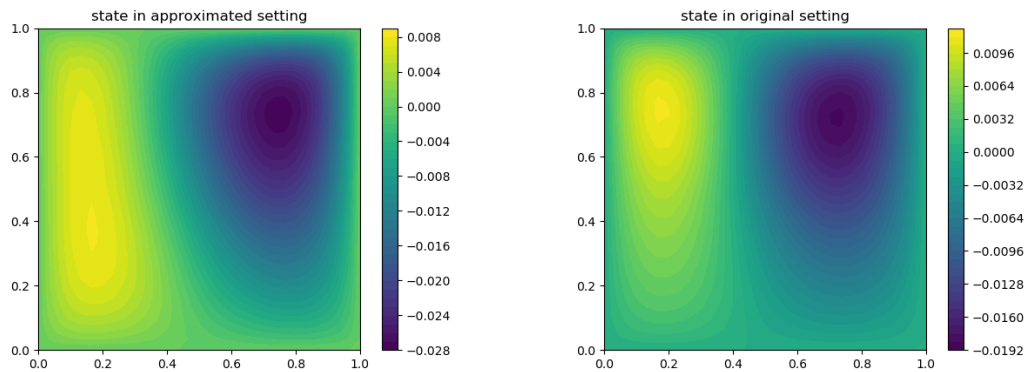


Figure 3.2: Contour plots of the solution y^N for the approximated setting (left) and original setting (right).

We can see that for this particular realization the resulting states look similar in both shape and amplitude. As expected for an approximation, there are also realizations for which the approximation is not as good, for instance as in Figure 3.3, where both, shape and amplitude of the states are quite different for $\xi_2 \approx (2.13, 2.12, 1.47, 3.5)$.

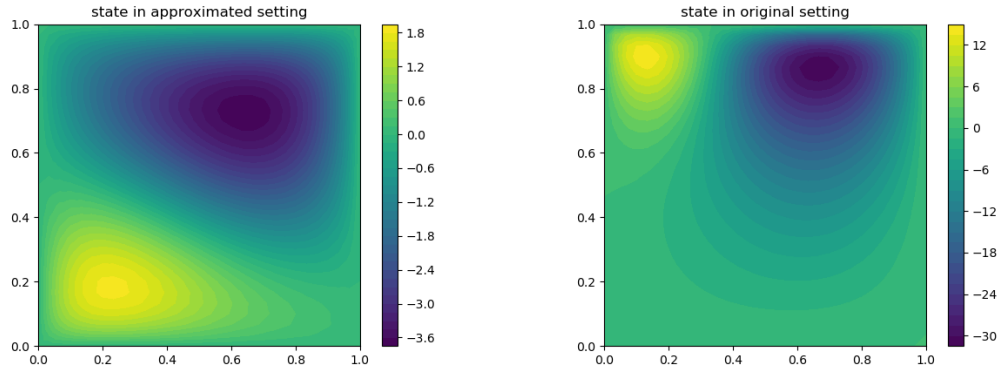


Figure 3.3: Contour plots of the solution y^N for the approximated setting (left) and original setting (right).

The approach used here is of course not very sophisticated. One could use higher-order Taylor approximation or different approaches such as the Empirical Interpolation Method, see [Bar+04] for instance, to obtain a more precise approximation while still keeping a functional form as in (3.2).

The main reason we are interested in the functional form of (3.2) is that solutions of the state and the adjoint equation can be computed faster, due to the preassembling of A^k and f^k . In Table 3.1 the required computational time to solve the state equation 100 times for different amounts of vertices used in the FE discretization is presented.

| | | | | |
|---|------|------|-------|-------|
| Vertices used | 441 | 961 | 1681 | 2601 |
| Required time with original function | 10.0 | 32.4 | 105.7 | 337.1 |
| Required time with Taylor approximation | 0.8 | 3.2 | 10.7 | 27.5 |

Table 3.1: Computational time (in sec.) to solve the state equation 100 times for different amounts of vertices used for the FE discretization.

We observe that the preassembling leads to an about ten times faster computation of the solution independent of the used amount of vertices.

Excursus - Reduced Basis Method

Another way to reduce the computational effort is the Reduced Basis (RB) method. For general derivation and theoretical background of the RB method as well as a description of how reduced basis spaces can be constructed can be found in [QMN16],

for instance. A more detailed derivation and numerical results of the RB method applied to a similar problem to $(\hat{\mathbf{P}})$ can be found in [Sin20], for instance. We will here just present the basic approach.

The main idea of the RB method is to reduce the N -dimensional system (3.17) to a ℓ -dimensional system with $\ell \ll N$. One starts by choosing

$$\psi_1, \dots, \psi_\ell \in V^N = \text{span}\{\varphi_1, \dots, \varphi_N\}.$$

As $\psi_i \in V^N$ for $i \in \{1, \dots, \ell\}$ it holds that there exist coefficients $\psi_{1i}, \dots, \psi_{Ni}$ such that

$$\psi_i(\mathbf{x}) = \sum_{j=1}^N \psi_{ji} \varphi_j(\mathbf{x}). \quad (3.18)$$

We denote

$$\Psi := (\psi_{ji})_{1 \leq j \leq N, 1 \leq i \leq \ell} \in \mathbb{R}^{N \times \ell},$$

such that

$$(\psi_1(\mathbf{x}), \dots, \psi_\ell(\mathbf{x}))^T = \Psi^T (\varphi_1(\mathbf{x}), \dots, \varphi_N(\mathbf{x}))^T.$$

Similar to (3.15) we use

$$y(\mathbf{x}) \approx y^\ell(\mathbf{x}) = \sum_{j=1}^{\ell} y_j^\ell \psi_j(\mathbf{x}). \quad (3.19)$$

We now want to find a solution $y^\ell \in V^\ell = \text{span}\{\psi_1, \dots, \psi_\ell\}$ which fulfils the reduced variational equation

$$\begin{aligned} & \sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) \int_{\mathcal{D}} a_k(\mathbf{x}) \nabla y^\ell(\mathbf{x}) \cdot \nabla \psi(\mathbf{x}) d\mathbf{x} \\ &= \sum_{k=1}^{N_f} \vartheta_k^f(\xi(\omega)) \int_{\mathcal{D}} f_k(\mathbf{x}) \psi(\mathbf{x}) d\mathbf{x} + \sum_{j=1}^{n_u} u_j \int_{\mathcal{D}} b_j(\mathbf{x}) \psi(\mathbf{x}) d\mathbf{x} \end{aligned} \quad (3.20)$$

for all $\psi \in V^\ell$.

Plugging (3.19) into (3.20) results in the discretized version

$$\begin{aligned}
& \sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) \sum_{j=1}^{\ell} \left[\int_{\mathcal{D}} a_k(\mathbf{x}) \nabla \psi_j(\mathbf{x}) \cdot \nabla \psi_i(\mathbf{x}) d\mathbf{x} \right] y_j^\ell \\
&= \sum_{k=1}^{N_f} \vartheta_k^f(\xi(\omega)) \int_{\mathcal{D}} f_k(\mathbf{x}) \psi_i(\mathbf{x}) d\mathbf{x} + \sum_{j=1}^{n_u} u_j \int_{\mathcal{D}} b_j(\mathbf{x}) \psi_i(\mathbf{x}) d\mathbf{x}
\end{aligned} \tag{3.21}$$

for all $i \in \{1, \dots, \ell\}$.

Due to (3.18) and (3.19) we have

$$\begin{aligned}
y^\ell(\mathbf{x}) &= (y^\ell)^T \Psi^T(\varphi_1(\mathbf{x}), \dots, \varphi_N(\mathbf{x}))^T \\
&= \sum_{j=1}^{\ell} \sum_{i=1}^N y_j^\ell \psi_{ij} \varphi_i(\mathbf{x}).
\end{aligned}$$

By defining

$$\begin{aligned}
A_{ij}^{k,\ell} &:= \int_{\mathcal{D}} a_k(\mathbf{x}) \nabla \psi_i(\mathbf{x}) \cdot \nabla \psi_j(\mathbf{x}) dx \\
&= \sum_{\nu=1}^N \sum_{\mu=1}^N \psi_{\nu i} \psi_{\mu j} \int_{\mathcal{D}} a_k(\mathbf{x}) \nabla \varphi_\mu(\mathbf{x}) \cdot \nabla \varphi_\nu(\mathbf{x}) dx \\
&= (\Psi^T A^k \Psi) \in \mathbb{R}^{\ell \times \ell}
\end{aligned}$$

and analogously

$$f_i^{k,\ell} := \Psi^T f^k \in \mathbb{R}^\ell,$$

$$B^\ell := \Psi^T B \in \mathbb{R}^{\ell \times n_u},$$

where A^k , f^k and B are defined in (3.16), (3.21) can be written as

$$\left[\sum_{k=1}^{N_a} \vartheta_k^a(\xi(\omega)) A^{k,\ell} \right] y^\ell = \sum_{k=1}^{N_f} \vartheta_k^f(\xi(\omega)) f^{k,\ell} + B^\ell u,$$

which is equivalent to multiplying (3.17) with Ψ^T from the left. Notice that this system is ℓ -dimensional.

Information on a posteriori error estimators as well as on extensions to the basic model can be found in [HRS16], for instance.

3.2 Generated Probability Space

The probability space $(\Omega, \mathcal{F}, \mathbb{P})$ used in the previous sections is an abstract space. For the numerical experiments in Chapter 5 we will use a discrete random space of which the underlying set is contained in \mathbb{R}^m . In this section, we will show how such a discrete random space can be obtained from the abstract random space $(\Omega, \mathcal{F}, \mathbb{P})$ using the so-called image measure.

To start, we show that a measure space contained in $(\mathbb{R}^m, \mathcal{B}(\mathbb{R}^m))$ can be generated from an abstract random space and a random vector mapping to \mathbb{R}^m .

Theorem 3.7. *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and ξ be a random vector. If ξ is bounded, then it holds with $\overline{\xi(\Omega)}$ being the closure of $\xi(\Omega) := \{\xi(\omega) : \omega \in \Omega\}$ with respect to $\|\cdot\|_2$ that*

$$\left(\overline{\xi(\Omega)}, \mathcal{B}(\mathbb{R}^m)|_{\overline{\xi(\Omega)}}\right)$$

is a measure space and $X : \overline{\xi(\Omega)} \rightarrow \mathbb{R}^m$ with $X(\hat{\xi}) = \hat{\xi}$ is a random vector.

Proof. See [Sin20, Theorem 7.9]. □

Additionally, it holds that we can generate a probability measure on \mathbb{R}^m with a random vector mapping to \mathbb{R}^m .

Proposition 3.8. *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and $\xi : \Omega \rightarrow \mathbb{R}^m$ be a random vector. Then*

$$\mathbb{P}_\xi := \mathbb{P}(\xi^{-1})$$

is a probability measure on $(\mathbb{R}^m, \mathcal{B}(\mathbb{R}^m))$. We will also call this measure the image measure of ξ .

Proof. See [Rüs16, Proposition 1.3.9]. □

Now we are able to generate a probability space with the underlying set being contained in \mathbb{R}^m from an abstract probability space.

Theorem 3.9. *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and ξ be a random variable with $|\xi(\omega)| < b$ for all $\omega \in \Omega$. Then it holds that*

$$\left(\overline{\xi(\Omega)}, \mathcal{B}(\mathbb{R}^m)|_{\overline{\xi(\Omega)}}, \mathbb{P}_\xi\right)$$

is a probability space.

Proof. Follows from Theorem 3.7 and Proposition 3.8. □

We will use a specific discrete probability space, the so-called Laplace space.

Definition 3.10 (Laplace space). Let Ω be a finite set with $|\Omega| = N \in \mathbb{N}$, 2^Ω be the powerset of Ω and

$$\mathbb{P} : 2^\Omega \rightarrow [0, 1], \quad A \mapsto \frac{1}{N} \sum_{\omega \in \Omega} \mathbb{1}_A(\omega).$$

Then the probability space $(\Omega, \mathcal{F}, \mathbb{P})$ is called *Laplace space*.

Now we can define the random space we will use in Chapter 5. Consider the discrete abstract Laplace space $(\Omega, \mathcal{F}, \mathbb{P})$ with $\Omega = \{\omega_1, \dots, \omega_K\}$ and a bounded random vector $\xi : \Omega \rightarrow \mathbb{R}^m$. Then due to Theorem 3.9, we obtain a discrete probability space contained in \mathbb{R}^m by defining

$$\tilde{\Omega} := \{\xi(\omega_1), \dots, \xi(\omega_K)\} \subset \mathbb{R}^m, \quad \tilde{\mathcal{F}} := 2^{\tilde{\Omega}} \subset \mathcal{B}(\mathbb{R}^m), \quad \tilde{\mathbb{P}} := \mathbb{P}_\xi. \quad (3.22)$$

We will denote $\xi_i := \xi(\omega_i)$. The probability space defined in (3.22) even is a Laplace space as for $A \in \tilde{\mathcal{F}}$ it holds that

$$\tilde{\mathbb{P}}(A) = \frac{1}{K} \sum_{\omega \in \Omega} \mathbb{1}_{\xi^{-1}(A)}(\omega) = \frac{1}{K} \sum_{\omega \in \Omega} \mathbb{1}_A(\xi(\omega)) = \frac{1}{K} \sum_{\tilde{\xi} \in \tilde{\Omega}} \mathbb{1}_A(\tilde{\xi}).$$

Chapter 4

Descent Methods and Convergence Analysis

In this chapter, the basic ideas, the algorithms, the theoretical background and the convergence analysis of three different stochastic descent methods is presented. The name stochastic descent methods may be misleading as there is not a guaranteed descent in every iteration as it is for most of the deterministic descent methods such as the Gradient Descent method with Armijo line search. Nevertheless, this name is chosen as in expectation a descent occurs, which will be shown in the convergence analysis later on.

To start, a setting in which stochastic descent methods can be used is described.

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, $n_u \in \mathbb{N}$, $\mathcal{U}_{\text{ad}} \subset \mathbb{R}^{n_u}$ be a bounded set, $\xi : \Omega \rightarrow \Xi \subset \mathbb{R}^m$ be a random vector and $J : \mathcal{U}_{\text{ad}} \times \mathbb{R}^m \rightarrow \mathbb{R}$ be a function.

Consider the problem

$$\min_{u \in \mathcal{U}_{\text{ad}}} \hat{J}(u) := \mathbb{E}[J(u, \xi(\cdot))] = \int_{\Omega} J(u, \xi(\omega)) \, d\mathbb{P}(\omega). \quad (4.1)$$

A common approach to solve this problem numerically is to sample a discrete amount of elements of Ω . Then a discrete probability space is created as described in Section 3.2 and as an approximation of (4.1) we get

$$\min_{u \in \mathcal{U}_{\text{ad}}} \frac{1}{K} \sum_{i=1}^K J(u, \xi_i).$$

Solving this problem with common deterministic optimization methods, such as the Gradient Descent method, involves the calculation of the *full gradient*

$$\frac{1}{K} \sum_{i=1}^K \nabla J(u, \xi_i), \quad (4.2)$$

which can be computationally expensive for large K if the evaluation of the gradient of J is expensive. Stochastic descent methods are mainly used to reduce the computational cost in this setting by avoiding calculating the full gradient (4.2) in every iteration.

In the following, we will use a *stochastic gradient* $G(u, \xi)$, which is an approximation to the full gradient $\nabla \hat{J}(u)$ in the sense that $\mathbb{E}[G(u, \xi)] \approx \nabla \hat{J}(u)$. For example, the stochastic gradient can be

$$\nabla J(u, \xi),$$

where in the end just one realization of ξ will be used in every iteration. It can also be a so-called *batch gradient* with batch size $m \in \mathbb{N}$ defined by

$$\frac{1}{m} \sum_{i=1}^m \nabla J(u, \xi^i)$$

with ξ^1, \dots, ξ^m being *independent and identically distributed* (i.i.d.) random vectors. In general, when given an iterate u_k , the next iterate is computed as in a deterministic box-constrained gradient descent by

$$u_{k+1} = \mathcal{P}(u_k - \alpha_k G(u_k, \xi_k)), \quad (4.3)$$

where \mathcal{P} is again the projection on the admissible set \mathcal{U}_{ad} , α_k is a step size and ξ_k is the random vector used in this iteration. As the iterate u_k is a function of the random variable ξ_k and the history of iterates u_0, \dots, u_{k-1} which are not all known a priori, the iterate u_k is random itself.

4.1 Projected Stochastic Gradient

The Projected Stochastic Gradient (PSG) method, see [GP19] for instance, is the simplest stochastic descent method in which a fixed batch size (normally one) is used to compute an unbiased estimator of the full gradient. When a batch size larger than one is used, one also calls this method stochastic batch gradient descent.

Convergence Analysis

In this section, convergence analysis will be stated based on [GP19]. Consider therefore the setting used in Chapter 3 with the generalized problem (4.1). In the following, we assume that the stochastic gradient is given as an unbiased estimator of $\nabla \hat{J}$ which means that there exists a random variable $\bar{\omega}$ with $\mathbb{E}[\bar{\omega}] = 0$ such that

$$G(u, \xi) = \nabla \hat{J}(u) + \bar{\omega}$$

holds true almost sure for all $u \in \mathcal{U}_{\text{ad}}$.

Further, we assume that there exist positive constants M_1, M_2, M such that

$$\mathbb{E}[\|G(u, \xi)\|_2^2] \leq M_1 + M_2 \|u\|_2^2 \leq M. \quad (4.4)$$

Note that this assumption is fulfilled for our problem ($\hat{\mathbf{P}}$), see [Sin20, Corollary 4.2] for instance.

We start with a corollary that states a bound of the expected difference of the control at iteration $k + 1$ to the optimal control depending on the difference at iteration k , the step size α_k , the convexity parameter μ and the bound of the stochastic gradient M .

Corollary 4.1. *Let \hat{J} be μ -strongly convex, $(u_k)_{k \in \mathbb{N}}$ be the sequence generated by (4.3) and \bar{u} be the optimal solution of (4.1). Define $e_k := \mathbb{E}[\|u_k - \bar{u}\|_2^2]$, then*

$$e_{k+1} \leq e_k(1 - \alpha_k \mu) + \alpha_k^2 M$$

holds true for all $k \in \mathbb{N}$ with M as in (4.4).

Proof. See [Sin20, Corollary 3.2]. □

Now we are able to find a bound at iteration k depending on the initial control instead of the preceding control at iteration $k - 1$.

Theorem 4.2. *Let \hat{J} be μ -strongly convex, $M > 0$ chosen as in (4.4) and \bar{u} the optimal solution. Let $\vartheta > \frac{1}{\mu}$, $\alpha_k := \frac{\vartheta}{k}$ and $C := \max(\vartheta^2 M (\mu \vartheta - 1)^{-1}, \|u_0 - \bar{u}\|_2^2)$. Then*

$$e_k \leq \frac{C}{k}$$

holds true for all $k \geq 1$. Further it holds that

$$\mathbb{E}[\|u_k - \bar{u}\|_2] \leq \sqrt{\frac{C}{k}}.$$

Proof. See [Sin20, Theorem 3.2] and [Sin20, Corollary 3.3]. \square

This theorem shows that, with a specific step size chosen, the expected difference of the control to the optimal control decreases in every iteration.

Lastly, we extend the statement of Theorem 4.2 to obtain a similar statement for the expected difference of the function value of \hat{J} at iteration k to the optimal value $\hat{J}(\bar{u})$.

Corollary 4.3. *Let \hat{J} be μ -strongly convex and $\nabla \hat{J}$ Lipschitz continuous with Lipschitz constant L . Further let the optimal solution \bar{u} be an interior point of \mathcal{U}_{ad} and C as in Theorem 4.2. Then*

$$\mathbb{E}[|\hat{J}(u_k) - \hat{J}(\bar{u})|] \leq \frac{LC}{2k}$$

holds true for all $k \geq 1$.

Proof. This proof is based on [Sin20, Corollary 3.4]. Let u_k be an iterate, \bar{u} the optimal solution and

$$\phi: [0, 1] \rightarrow \mathbb{R}, \quad t \mapsto \hat{J}(\bar{u} + t(u_k - \bar{u})).$$

Then it holds that

$$\begin{aligned} \phi(0) &= \hat{J}(\bar{u}), \\ \phi(1) &= \hat{J}(u_k), \\ \phi'(t) &= \langle \nabla \hat{J}(\bar{u} + t(u_k - \bar{u})), u_k - \bar{u} \rangle_2. \end{aligned}$$

By using the fundamental theorem of calculus, see [DR11, Satz 8.23], the Cauchy-Schwarz inequality, see [DR11, Satz 5.49] and the Lipschitz continuity of $\nabla \hat{J}$ we get that

$$\begin{aligned} \hat{J}(u_k) - \hat{J}(\bar{u}) &= \phi(1) - \phi(0) = \int_0^1 \phi'(t) dt \\ &= \int_0^1 \langle \nabla \hat{J}(\bar{u} + t(u_k - \bar{u})), u_k - \bar{u} \rangle_2 - \langle \nabla \hat{J}(\bar{u}), u_k - \bar{u} \rangle_2 + \langle \nabla \hat{J}(\bar{u}), u_k - \bar{u} \rangle_2 dt \\ &= \int_0^1 \langle \nabla \hat{J}(\bar{u} + t(u_k - \bar{u})) - \nabla \hat{J}(\bar{u}), u_k - \bar{u} \rangle_2 dt + \int_0^1 \langle \nabla \hat{J}(\bar{u}), u_k - \bar{u} \rangle_2 dt \\ &\leq \int_0^1 Lt \|u_k - \bar{u}\|_2^2 dt + \langle \nabla \hat{J}(\bar{u}), u_k - \bar{u} \rangle_2 \\ &= \frac{L}{2} \|u_k - \bar{u}\|_2^2 + \langle \nabla \hat{J}(\bar{u}), u_k - \bar{u} \rangle_2. \end{aligned} \tag{4.5}$$

As \bar{u} is the optimal solution to the box-constraint problem $(\hat{\mathbf{P}})$, the necessary first order optimality condition, see Theorem 2.23,

$$\langle \nabla \hat{J}(\bar{u}), u - \bar{u} \rangle_2 \geq 0 \quad \text{for all } u \in \mathcal{U}_{\text{ad}}$$

holds true. As \bar{u} is an interior point there exists some $\varepsilon > 0$ such that $\bar{u} \pm \varepsilon e_i \in \mathcal{U}_{\text{ad}}$ for all $i \in \{1, \dots, n_u\}$ where $e_i \in \mathcal{U}$ is the canonical unit vector. Thus

$$0 \leq \langle \nabla \hat{J}(\bar{u}), \bar{u} \pm \varepsilon e_i - \bar{u} \rangle_2 = \pm \varepsilon \langle \nabla \hat{J}(\bar{u}), e_i \rangle_2 = \pm \varepsilon \nabla \hat{J}(\bar{u})_i \quad (4.6)$$

and therefore $\nabla \hat{J}(\bar{u})_i = 0$ holds true for all $i \in \{1, \dots, n_u\}$ such that

$$\langle \nabla \hat{J}(\bar{u}), u_k - \bar{u} \rangle_2 = 0.$$

Using this, Theorem 4.2 and (4.5) we obtain

$$\mathbb{E}[\hat{J}(u_k) - \hat{J}(\bar{u})] \leq \frac{L}{2} \mathbb{E}[\|u_k - \bar{u}\|_2^2] \leq \frac{LC}{2k}.$$

□

As stated in Chapter 3, \hat{J} of $(\hat{\mathbf{P}})$ is λ -strongly convex and $\nabla \hat{J}$ is Lipschitz continuous. Thus, the preceding convergence results hold true for $(\hat{\mathbf{P}})$ and we expect that u_k converges to \bar{u} with a rate of $\mathcal{O}(k^{-0.5})$ and that the corresponding function values $\hat{J}(u_k)$ converge with a rate of $\mathcal{O}(k^{-1})$ to $\hat{J}(\bar{u})$.

In Chapter 5, we will investigate the convergence behaviour in practice.

In the following, we will denote

$$G^\xi(u) := \nabla J(u, \xi) \quad (4.7)$$

for a random variable ξ and the batch gradient for a batch size of m as

$$G^m(u) := \frac{1}{m} \sum_{i=1}^m \nabla J(u, \xi^i) \quad (4.8)$$

for i.i.d. random vectors ξ^1, \dots, ξ^m . Note that even though we omit the random vectors as inputs of G , the stochastic gradient is still a random variable until fixed realizations of the random vectors are generated. The basic algorithm is stated in Algorithm 1.

Algorithm 1 Projected Stochastic Gradient (PSG)

Require: Initial control $u_0 \in \mathcal{U}_{\text{ad}}$, batch size m , Ω .

- 1: Set $k = 0$
 - 2: **while** stopping criterion not satisfied **do**
 - 3: Generate i.i.d. ξ_k^1, \dots, ξ_k^m independent of $\xi_0^1, \dots, \xi_0^m, \dots, \xi_{k-1}^1, \dots, \xi_{k-1}^m$
 - 4: Generate $G_k^m(u_k)$
 - 5: Compute step size $\alpha_k > 0$
 - 6: $u_{k+1} = \mathcal{P}(u_k - \alpha_k G_k^m(u_k))$
 - 7: $k = k + 1$
 - 8: **end while**
-

The advantages of this method are the low memory requirement, the low computational effort and the simplicity. Disadvantages on the other hand are potentially high variance in the computed gradients as just very little information about the full gradient is used for small batch sizes, and that the performance is very sensitive to the appropriate step size choice.

4.2 Projected Stochastic Variance Reduced Gradient

The main idea of the Stochastic Variance Reduced Gradient (SVRG) method [JZ13] is to compute the full gradient at the start of every iteration and use a slightly updated gradient for batch size many steps. This leads to the computed gradient being used almost unchanged for batch size many steps which results, as the name suggests, in a reduced variance of the descent direction which is a main advantage of this method over the PSG method. A downside of this is that it is necessary to evaluate the full gradient, which can be computationally expensive.

A suitable step size algorithm is the Barzilai-Borwein (BB) step size algorithm, which can be found, in [Tan+16] for instance. This step size rule is motivated by the so-called secant equation

$$B_k s_k = y_k,$$

where $s_k = u_k - u_{k-1}$ and $y_k = \nabla \hat{J}(u_k) - \nabla \hat{J}(u_{k-1})$. As solving this linear equation system is costly for large n_u , one approximates B_k by $\frac{1}{\alpha_k} I$ with $\alpha_k > 0$.

This leads to solving the following least-squares optimization problem

$$\min_{\alpha_k \in \mathbb{R}_{>0}} \|\alpha_k^{-1} s_k - y_k\|_2^2. \quad (4.9)$$

Theorem 4.4. Given \hat{J} of $(\hat{\mathbf{P}})$ the problem (4.9) has the unique solution

$$\alpha_k = \frac{\|s_k\|_2^2}{s_k^T y_k} = \frac{\|s_k\|_2^2}{s_k^T \nabla^2 \hat{J}(u_{k-1}) s_k}, \quad (4.10)$$

where $s_k = u_k - u_{k-1}$ and $y_k = \nabla \hat{J}(u_k) - \nabla \hat{J}(u_{k-1})$.

Proof. As (4.9) is a convex optimization problem the solution can be characterized by the sufficient first order condition. Define $g : \mathbb{R}_{>0} \rightarrow \mathbb{R}$, $\alpha \mapsto \|\alpha^{-1} s_k - y_k\|_2^2$. We get

$$g'(\alpha) = -\frac{2}{\alpha^2} s_k^T (\alpha^{-1} s_k - y_k) = 0 \quad \Leftrightarrow \quad \alpha s_k^T y_k = \|s_k\|_2^2. \quad (4.11)$$

As \hat{J} is a quadratic function and therefore $\nabla \hat{J}$ is a linear function it holds that

$$\nabla \hat{J}(u_k) - \nabla \hat{J}(u_{k-1}) = \nabla^2 \hat{J}(u_{k-1})(u_k - u_{k-1}). \quad (4.12)$$

Therefore we obtain by using (4.12)

$$\begin{aligned} s_k^T y_k &= (u_k - u_{k-1})^T (\nabla \hat{J}(u_k) - \nabla \hat{J}(u_{k-1})) \\ &= (u_k - u_{k-1})^T \nabla^2 \hat{J}(u_{k-1})(u_k - u_{k-1}) > 0 \end{aligned}$$

as $\nabla^2 \hat{J}$ is positive definite by Theorem 3.5. Thus we get by rearranging (4.11) that the Barzilai-Borwein step size is given as $\alpha_k = \frac{\|s_k\|_2^2}{s_k^T y_k}$. \square

This leads us to Algorithm 2, that presents the step size procedure.

Algorithm 2 Barzilai-Borwein (BB)

Require: $u_k, u_{k-1} \in \mathcal{U}_{\text{ad}}, \nabla \hat{J}(u_k), \nabla \hat{J}(u_{k-1}) \in \mathcal{U}$.

- 1: $s_k = u_k - u_{k-1}$
 - 2: $y_k = \nabla \hat{J}(u_k) - \nabla \hat{J}(u_{k-1})$
 - 3: $\alpha_k = \frac{\|s_k\|_2^2}{s_k^T y_k}$
-

One can see that Algorithm 2 can not be applied in the first iteration as two iterates, as well as two gradients, are needed. Therefore, the first step size has to be set in implementation. We will just use a fixed step size of $\alpha_0 = 1$ for the first iteration, but of course, there are other step size procedures that can be used alternatively.

The Barzilai-Borwein step size procedure is not applicable for the Stochastic Gradient method as there is no evaluation of the full gradient. One could suggest using the stochastic gradient G^ξ as an approximation but that does not work well because

it would lead to a high variance in the step lengths and thus unstable convergence behaviour. A way to still incorporate the Barzilai-Borwein step size procedure in a slightly changed stochastic gradient method is shown in [Tan+16].

The Projected Stochastic Variance Reduced Gradient algorithm is stated in Algorithm 3.

Algorithm 3 Projected Stochastic Variance Reduced Gradient (PSVRG)

Require: Initial control $u_0 \in \mathcal{U}_{\text{ad}}$, batch size m , Ω .

- 1: Set $k = 0$
 - 2: **while** stopping criterion not satisfied **do**
 - 3: Compute search direction $d_k = \nabla \hat{J}(u_k)$
 - 4: Compute step size α_k with Algorithm 2
 - 5: Generate i.i.d. ξ_k^1, \dots, ξ_k^m independent of $\xi_0^1, \dots, \xi_0^m, \dots, \xi_{k-1}^1, \dots, \xi_{k-1}^m$
 - 6: Generate $\nabla J(u_k, \xi_k^i)$ for $i = 1, \dots, m$
 - 7: $\tilde{u}_0 = u_k$
 - 8: **for** $i = 0, \dots, m - 1$ **do**
 - 9: $\tilde{d}_i = d_k - \nabla J(u_k, \xi_k^i) + \nabla J(\tilde{u}_i, \xi_k^i)$
 - 10: $\tilde{u}_{i+1} = \mathcal{P}(\tilde{u}_i - \frac{\alpha_k}{m} \tilde{d}_i)$
 - 11: **end for**
 - 12: $u_{k+1} = \tilde{u}_m$
 - 13: $k = k + 1$
 - 14: **end while**
-

Convergence Analysis

In the following, we will assume that $\nabla J(u, \xi)$ is an unbiased estimator of $\nabla \hat{J}(u)$, i.e. that it holds for all $u \in \mathcal{U}_{\text{ad}}$ that

$$\mathbb{E}[\nabla J(u, \xi)] = \nabla \hat{J}(u). \quad (4.13)$$

We again start by stating a bound on the expected difference of the control at iteration k to the optimal control.

Theorem 4.5. *Let \hat{J} be μ -strongly convex, $\nabla \hat{J}$ be Lipschitz continuous with Lipschitz constant L and \bar{u} be an interior point of \mathcal{U}_{ad} . Define*

$$C_k := \left(1 - 2\mu \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right)\right)^m + \frac{4 \frac{\alpha_k}{m} L^2}{\mu \left(1 - \frac{\alpha_k}{m} L\right)}.$$

Then the sequence $(u_k)_{k \in \mathbb{N}}$ generated by Algorithm 3 satisfies

$$\mathbb{E}[\|u_{k+1} - \bar{u}\|_2^2] < C_k \mathbb{E}[\|u_k - \bar{u}\|_2^2].$$

Proof. This proof is based on [Tan+16, Lemma 3.5]. Denote $\tilde{d}_i := d_k - \nabla J(u_k, \xi_k^i) + \nabla J(\tilde{u}_i, \xi_k^i)$ with $d_k = \nabla \hat{J}(u_k)$ as in Algorithm 3. Note that for u_k generated by Algorithm 3 it holds by the assumption (4.13) that

$$\mathbb{E}[\nabla J(u_k, \xi_k^i) | u_k] = \nabla \hat{J}(u_k) \quad (4.14)$$

and that

$$\nabla \hat{J}(\bar{u}) = 0, \quad (4.15)$$

see (4.6).

It holds that

$$\begin{aligned} & \mathbb{E}[\|\tilde{d}_i\|_2^2 | u_k, \tilde{u}_i] \\ &= \mathbb{E}[\|d_k - (\nabla J(u_k, \xi_k^i) - \nabla J(\bar{u}, \xi_k^i)) + (\nabla J(\tilde{u}_i, \xi_k^i) - \nabla J(\bar{u}, \xi_k^i))\|_2^2 | u_k, \tilde{u}_i] \\ &\leq 4\mathbb{E}[\|d_k\|_2^2 | u_k, \tilde{u}_i] + 4\mathbb{E}[\|\nabla J(u_k, \xi_k^i) - \nabla J(\bar{u}, \xi_k^i)\|_2^2 | u_k, \tilde{u}_i] \\ &\quad + 2\mathbb{E}[\|\nabla J(\tilde{u}_i, \xi_k^i) - \nabla J(\bar{u}, \xi_k^i)\|_2^2 | u_k, \tilde{u}_i] \\ &\leq 4L^2\mathbb{E}[\|u_k - \bar{u}\|_2^2 | u_k, \tilde{u}_i] + 4L^2\mathbb{E}[\|u_k - \bar{u}\|_2^2 | u_k, \tilde{u}_i] \\ &\quad + 2L\mathbb{E}[(\tilde{u}_i - \bar{u})^T (\nabla J(\tilde{u}_i, \xi_k^i) - \nabla J(\bar{u}, \xi_k^i)) | u_k, \tilde{u}_i] \\ &= 8L^2\|u_k - \bar{u}\|_2^2 + 2L(\tilde{u}_i - \bar{u})^T (\nabla \hat{J}(\tilde{u}_i) - \nabla \hat{J}(\bar{u})) \\ &= 8L^2\|u_k - \bar{u}\|_2^2 + 2L(\tilde{u}_i - \bar{u})^T \nabla \hat{J}(\tilde{u}_i), \end{aligned} \quad (4.16)$$

where we used the triangle inequality and $(a + b)^2 \leq 2a^2 + 2b^2$ twice in the first inequality, applied Lemma 2.21, used the Lipschitz continuity of $\nabla J(\cdot, \xi_k^i)$ and $\nabla \hat{J}$ as well as (4.15) in the second inequality, used (4.14) in the second equality and (4.15) for the last equality.

It further holds with Theorem 2.17 that

$$\begin{aligned} -(\tilde{u}_i - \bar{u})^T \nabla \hat{J}(\tilde{u}_i) &= \nabla \hat{J}(\tilde{u}_i)^T (\bar{u} - \tilde{u}_i) \\ &\leq -\mu \|\tilde{u}_i - \bar{u}\|_2^2 + \hat{J}(\bar{u}) - \hat{J}(\tilde{u}_i) \\ &\leq -\mu \|\tilde{u}_i - \bar{u}\|_2^2, \end{aligned} \quad (4.17)$$

where we also used that \bar{u} is the optimal solution.

Thus, conditioned on \tilde{u}_i and u_k we get that

$$\begin{aligned}
& \mathbb{E}[\|\tilde{u}_{i+1} - \bar{u}\|_2^2 | u_k, \tilde{u}_i] \\
&= \mathbb{E} \left[\left\| \mathcal{P} \left(\tilde{u}_i - \frac{\alpha_k}{m} \tilde{d}_i \right) - \bar{u} \right\|_2^2 \middle| u_k, \tilde{u}_i \right] \\
&\leq \|\tilde{u}_i - \bar{u}\|_2^2 - 2 \frac{\alpha_k}{m} \mathbb{E}[(\tilde{u}_i - \bar{u})^T \tilde{d}_i | u_k, \tilde{u}_i] + \frac{\alpha_k^2}{m^2} \mathbb{E}[\|\tilde{d}_i\|_2^2 | u_k, \tilde{u}_i] \\
&= \|\tilde{u}_i - \bar{u}\|_2^2 - 2 \frac{\alpha_k}{m} (\tilde{u}_i - \bar{u})^T (\nabla \hat{J}(u_k) - \nabla \hat{J}(u_k) + \nabla \hat{J}(\tilde{u}_i)) + \frac{\alpha_k^2}{m^2} \mathbb{E}[\|\tilde{d}_i\|_2^2 | u_k, \tilde{u}_i] \\
&\leq \|\tilde{u}_i - \bar{u}\|_2^2 - 2 \frac{\alpha_k}{m} (\tilde{u}_i - \bar{u})^T \nabla \hat{J}(\tilde{u}_i) + 8 \frac{\alpha_k^2}{m^2} L^2 \|u_k - \bar{u}\|_2^2 + 2 \frac{\alpha_k^2}{m^2} L (\tilde{u}_i - \bar{u})^T \nabla \hat{J}(\tilde{u}_i) \\
&= \|\tilde{u}_i - \bar{u}\|_2^2 - 2 \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right) (\tilde{u}_i - \bar{u})^T \nabla \hat{J}(\tilde{u}_i) + 8 \frac{\alpha_k^2}{m^2} L^2 \|u_k - \bar{u}\|_2^2 \\
&\leq \|\tilde{u}_i - \bar{u}\|_2^2 - 2\mu \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right) \|\tilde{u}_i - \bar{u}\|_2^2 + 8 \frac{\alpha_k^2}{m^2} L^2 \|u_k - \bar{u}\|_2^2 \\
&= \left(1 - 2\mu \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right)\right) \|\tilde{u}_i - \bar{u}\|_2^2 + 8 \frac{\alpha_k^2}{m^2} L^2 \|u_k - \bar{u}\|_2^2, \tag{4.18}
\end{aligned}$$

where we used Theorem 2.25 for the first inequality, (4.16) for the second inequality and (4.17) for the third inequality.

By taking the on u_k conditioned expected value in (4.18) on both sides we get with Theorem 2.34 that

$$\begin{aligned}
& E[\|\tilde{u}_{i+1} - \bar{u}\|_2^2 | u_k] \\
&\leq \left(1 - 2\mu \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right)\right) \mathbb{E}[\|\tilde{u}_i - \bar{u}\|_2^2 | u_k] + 8 \frac{\alpha_k^2}{m^2} L^2 \|u_k - \bar{u}\|_2^2. \tag{4.19}
\end{aligned}$$

By recursively applying (4.19) over i , and noting that as in Algorithm 3 $u_k = \tilde{u}_0$ and $u_{k+1} = \tilde{u}_m$, we obtain

$$\begin{aligned}
& \mathbb{E}[\|u_{k+1} - \bar{u}\|_2^2 | u_k] \\
&\leq \left(1 - 2\mu \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right)\right)^m \|u_k - \bar{u}\|_2^2 + 8L^2 \frac{\alpha_k^2}{m^2} \sum_{j=0}^{m-1} \left(1 - 2\mu \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right)\right)^j \|u_k - \bar{u}\|_2^2 \\
&< \left(\left(1 - 2\mu \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right)\right)^m + 8L^2 \frac{\alpha_k^2}{m^2} \sum_{j=0}^{\infty} \left(1 - 2\mu \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right)\right)^j \right) \|u_k - \bar{u}\|_2^2 \\
&= \left(\left(1 - 2\mu \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right)\right)^m + \frac{4 \frac{\alpha_k}{m} L^2}{\mu \left(1 - \frac{\alpha_k}{m} L\right)} \right) \|u_k - \bar{u}\|_2^2 \\
&= C_k \|u_k - \bar{u}\|_2^2,
\end{aligned}$$

where we used a geometric sum. The claim follows by taking the expected value on

both sides. □

By requiring a sufficiently large batch size m one can even conclude the following theorem.

Theorem 4.6. *Let the assumptions of Theorem 4.5 hold true. Further let $\theta \in (0, 0.5)$ be fixed and let the batch size m satisfy*

$$m > \max \left\{ \frac{1}{\log(1 - 2\theta) + 2\mu/L}, \frac{2L^2}{\theta\mu^2} + \frac{L}{2\mu} \right\}. \quad (4.20)$$

Let further $K := \|u_0 - \bar{u}\|_2^2$. Then it holds for all $k \geq 1$ that

$$\mathbb{E}[\|u_k - \bar{u}\|_2^2] < K(1 - \theta)^k. \quad (4.21)$$

Further it holds that

$$\mathbb{E}[\|u_k - \bar{u}\|_2] < \sqrt{K(1 - \theta)^k}. \quad (4.22)$$

Proof. This proof is based on [Tan+16, Theorem 3.8]. As in Algorithm 3, we will denote $d_k = \nabla \hat{J}(u_k)$. We start by showing that the Barzilai-Borwein step size is bounded. With the μ -strongly convexity of \hat{J} we get with Theorem 2.17 that

$$\begin{aligned} (d_k - d_{k-1})^T (u_k - u_{k-1}) &= -d_k^T (u_{k-1} - u_k) - d_{k-1}^T (u_k - u_{k-1}) \\ &\geq \hat{J}(u_k) - \hat{J}(u_{k-1}) + \mu \|u_{k-1} - u_k\|_2^2 + \hat{J}(u_{k-1}) - \hat{J}(u_k) + \mu \|u_k - u_{k-1}\|_2^2 \\ &= 2\mu \|u_{k-1} - u_k\|_2^2. \end{aligned}$$

Thus the Barzilai Borwein step size (4.10) with $s_k = u_k - u_{k-1}$ and $y_k = d_k - d_{k-1}$ is bounded from above as

$$\alpha_k = \frac{\|u_k - u_{k-1}\|_2^2}{(u_k - u_{k-1})^T (d_k - d_{k-1})} \leq \frac{1}{2\mu}.$$

As $\nabla \hat{J}$ is Lipschitz continuous it holds that

$$\begin{aligned}\alpha_k &= \frac{\|u_k - u_{k-1}\|_2^2}{(u_k - u_{k-1})^T (d_k - d_{k-1})} \\ &\geq \frac{\|u_k - u_{k-1}\|_2^2}{\|u_k - u_{k-1}\|_2 \|d_k - d_{k-1}\|_2} \\ &= \frac{\|u_k - u_{k-1}\|_2}{\|d_k - d_{k-1}\|_2} \\ &\geq \frac{1}{L},\end{aligned}$$

where we also have used the Cauchy-Schwarz inequality.

Therefore C_k from Theorem 4.5 can be bounded as

$$\begin{aligned}C_k &= \left(1 - 2\mu \frac{\alpha_k}{m} \left(1 - \frac{\alpha_k}{m} L\right)\right)^m + \frac{4 \frac{\alpha_k}{m} L^2}{\mu \left(1 - \frac{\alpha_k}{m} L\right)} \\ &\leq \left(1 - \frac{2\mu}{mL} \left(1 - \frac{L}{2m\mu}\right)\right)^m + \frac{2L^2}{m\mu^2 \left(1 - L/(2m\mu)\right)} \\ &\leq \exp\left(-\frac{2\mu}{mL} \left(1 - \frac{L}{2m\mu}\right) m\right) + \frac{2L^2}{m\mu^2 - L\mu/2} \\ &= \exp\left(-\frac{2\mu}{L} + \frac{1}{m}\right) + \frac{2L^2}{m\mu^2 - L\mu/2},\end{aligned}\tag{4.23}$$

where we used that $1 + x \leq \exp(x)$. Using (4.20) in (4.23) yields

$$\begin{aligned}C_k &< \exp\left(-\frac{2\mu}{L} + \log(1 - 2\theta) + \frac{2\mu}{L}\right) + \frac{2L^2}{\left(\frac{2L^2}{\theta\mu^2} + \frac{L}{2\mu}\right)\mu^2 - \frac{L\mu}{2}} \\ &= (1 - 2\theta) + \frac{2L^2}{\frac{2L^2}{\theta} + \frac{L\mu}{2} - \frac{L\mu}{2}} \\ &= (1 - 2\theta) + \theta = 1 - \theta.\end{aligned}$$

Applying Theorem 4.5 iteratively and noting that u_0 is deterministic yields (4.21) as

$$\mathbb{E}[\|u_k - \bar{u}\|_2^2] < \mathbb{E}[\|u_0 - \bar{u}\|_2^2] \prod_{j=0}^{k-1} C_j < \|u_0 - \bar{u}\|_2^2 (1 - \theta)^k.\tag{4.24}$$

Finally (4.22) follows by applying Jensen's inequality (Theorem 2.32) with $f(x) = x^2$. \square

Similar to Corollary 4.3 for the PSG method, one can extend the statement on the iterates to a statement on the function values of \hat{J} .

Corollary 4.7. *Let again the assumptions of Theorem 4.5 hold true and let θ , m and K be as in Theorem 4.6. Assume further that \bar{u} is an interior point of \mathcal{U}_{ad} . Then it holds for all $k \geq 1$ that*

$$\mathbb{E}[|\hat{J}(u_k) - \hat{J}(\bar{u})|] < \frac{LK}{2}(1 - \theta)^k.$$

Proof. This can be proven analogous to Corollary 4.3. □

4.3 Projected Stochastic Adaptive Sampling

The unique feature of the Stochastic Adaptive Sampling (SAS) method [BBN17] is that the batch size is not fixed but can be adjusted in every iteration. This allows us to start with a small batch size and therefore low computational effort, but also to use large batch sizes or even the full gradient when being close to the minimum.

Let in the following ξ^1, \dots, ξ^m be i.i.d random vectors and G the stochastic gradient defined as in (4.7) and (4.8). The batch size is adjusted depending on two tests, an *orthogonality test* and an *inner product test*. The inner product test is, generally speaking, used to control the variance of the used gradients. The *exact inner product test* at an iterate u is given by

$$\frac{\mathbb{E}[(G^{\xi^i}(u)^T \nabla \hat{J}(u) - \|\nabla \hat{J}(u)\|_2^2)^2 | u]}{m} \leq \rho^2 \|\nabla \hat{J}(u)\|_2^4, \quad (4.25)$$

where m is the batch size and $\rho > 0$ is a constant. This condition on the variance of the gradients ensures that

$$\mathbb{E}[(G^m(u)^T \nabla \hat{J}(u) - \|\nabla \hat{J}(u)\|_2^2)^2 | u] \leq \rho^2 \|\nabla \hat{J}(u)\|_2^4. \quad (4.26)$$

In practice, when having obtained the iterate u , we approximate the left-hand side of (4.25) with the sample variance and use stochastic gradient G^m as an approximation of $\nabla \hat{J}$ such that we get

$$\frac{\text{Var}_{1 \leq i \leq m}(G^{\xi^i}(u)^T G^m(u))}{m} \leq \rho^2 \|G^m(u)\|_2^4, \quad (4.27)$$

where

$$\text{Var}_{1 \leq i \leq m}(G^{\xi^i}(u)^T G^m(u)) = \frac{1}{m-1} \sum_{i=1}^m \left(G^{\xi^i}(u)^T G^m(u) - \|G^m(u)\|_2^2 \right)^2.$$

For further information see [BBN17, Formula (2.6)].

The second test, the so-called orthogonality test, is used to obtain a better quality of the descent direction by ensuring that the used gradient is not orthogonal to the full gradient, which would lead to no descent. The *exact orthogonality test* at an iterate u is given by

$$\frac{\mathbb{E} \left[\left\| G^{\xi^i}(u) - \frac{G^{\xi^i}(u)^T \nabla \hat{J}(u)}{\|\nabla \hat{J}(u)\|_2^2} \nabla \hat{J}(u) \right\|_2^2 \middle| u \right]}{m} \leq \nu^2 \|\nabla \hat{J}(u)\|_2^2, \quad (4.28)$$

where m is the current batch size and $\nu > 0$ is a constant. This condition is used to ensure that

$$\mathbb{E} \left[\left\| G^m(u) - \frac{G^m(u)^T \nabla \hat{J}(u)}{\|\nabla \hat{J}(u)\|_2^2} \nabla \hat{J}(u) \right\|_2^2 \middle| u \right] \leq \nu^2 \|\nabla \hat{J}(u)\|_2^2. \quad (4.29)$$

As for the inner product test, we will, in application, use an approximation given by

$$\frac{\text{Var}_{1 \leq i \leq m} \left(G^{\xi^i}(u) - \frac{G^{\xi^i}(u)^T G^m(u)}{\|G^m(u)\|_2^2} G^m(u) \right)}{m} \leq \nu^2 \|G^m(u)\|_2^2, \quad (4.30)$$

where

$$\begin{aligned} & \text{Var}_{1 \leq i \leq m} \left(G^{\xi^i}(u) - \frac{G^{\xi^i}(u)^T G^m(u)}{\|G^m(u)\|_2^2} G^m(u) \right) \\ &= \frac{1}{m-1} \sum_{i=1}^m \left\| G^{\xi^i}(u) - \frac{G^{\xi^i}(u)^T G^m(u)}{\|G^m(u)\|_2^2} G^m(u) \right\|_2^2, \end{aligned}$$

which is explained in more detail in [BBN17, Formula (3.3)].

Both tests together are used to determine whether and by which factor the batch size is increased. If either (4.27) or (4.30) is not fulfilled with the current batch of realizations one calculates the ratios

$$\text{ip-ratio} := \frac{\text{Var}_{1 \leq i \leq m} (G^{\xi^i}(u)^T G^m(u))}{\rho^2 \|G^m(u)\|_2^4} \quad (4.31)$$

and

$$\text{orth-ratio} := \frac{\text{Var}_{1 \leq i \leq m} \left(G^{\xi^i}(u) - \frac{G^{\xi^i}(u)^T G^m(u)}{\|G^m(u)\|_2^2} G^m(u) \right)}{\nu^2 \|G^m(u)\|_2^2}, \quad (4.32)$$

which result by rearranging (4.27) and (4.30) such that the batch size m is on the right-hand side of the inequalities. Then the new batch size m_{new} is chosen as

$$m_{new} = \max(\text{ip-ratio}, \text{orth-ratio}) \cdot m. \quad (4.33)$$

Intuitively speaking, m_{new} is chosen such that the new batch should fulfil both the inner product test and the orthogonality test if it is similar to the old batch regarding the computed variances and norms. A more detailed explanation can be found in [BBN17, Section 4.3].

Additionally, if the batch size remains constant for r steps, both of the ratios (4.31) and (4.32) are computed with an average gradient

$$g_{\text{avg}} := \frac{1}{r} \sum_{j=k-r+1}^k G_j^{m_j}(u_j) \quad (4.34)$$

where $G_{m_j}^j$ is the realization of the stochastic gradient in the j -th iteration with the batch size m_j .

If one of them is larger than 1 the new batch size is computed as in (4.33). The full method is presented in Algorithm 4.

Algorithm 4 Projected Stochastic Adaptive Sampling (PSAS)

Require: Initial control $u_0 \in \mathcal{U}_{\text{ad}}$, initial batch size m_0 , Parameters: $\rho > 0$, $\nu > 0$, $r > 0$, $\gamma \in (0, 1)$.

- 1: Set $k = 0$
- 2: **while** stopping criterion not satisfied **do**
- 3: Generate i.i.d. $\xi_k^1, \dots, \xi_k^{m_k}$ independent of $\xi_0^1, \dots, \xi_0^{m_0}, \dots, \xi_{k-1}^1, \dots, \xi_{k-1}^{m_{k-1}}$
- 4: Generate $G^{m_k}(u_k)$
- 5: Compute step size α_k
- 6: $u_{k+1} = \mathcal{P}(u_k - \alpha_k G^{m_k}(u_k))$
- 7: $k = k + 1$
- 8: Set $m_k = m_{k-1}$
- 9: Generate i.i.d. $\xi_k^1, \dots, \xi_k^{m_k}$ independent of $\xi_0^1, \dots, \xi_0^{m_0}, \dots, \xi_{k-1}^1, \dots, \xi_{k-1}^{m_{k-1}}$
- 10: Generate $G^{m_k}(u_k)$
- 11: **if** ip-test or orthogonality-test not satisfied **then**
- 12: Compute m_k using (4.33)
- 13: **end if**
- 14: **if** batch size constant for r steps **then**
- 15: Compute the average of the previous r search directions g_{avg} as in (4.34)
- 16: **if** $\|g_{\text{avg}}\| < \gamma \|G^{m_k}(u_k)\|_2$ **then**
- 17: **if** ip-test or orthogonality-test not satisfied using g_{avg} **then**
- 18: Compute m_k using (4.33) with g_{avg}
- 19: **end if**
- 20: **end if**
- 21: **end if**
- 22: **end while**

Various step size algorithms can be used with the SAS method, but for the convergence theory, we will just use a fixed step size, meaning $\alpha_k = \alpha$ for all k .

Convergence Analysis

For the convergence analysis of the PSAS method, a two times continuously differentiable function is required. This holds true for $(\hat{\mathbf{P}})$, see Theorem 3.5.

We start with a statement of the descent of the function values in expectation.

Lemma 4.8. *Suppose that \hat{J} is twice continuously differentiable and that there exists an upper bound $L \in \mathbb{R}_{>0}$ for the largest eigenvalue of the symmetric matrix $\nabla^2 \hat{J}(u)$ for all $u \in \mathcal{U}_{\text{ad}}$.*

Let $(u_k)_{k \in \mathbb{N}}$ be the iterates generated by Algorithm 4 such that u_k is an interior point for all k , where the batch size is chosen such that the exact inner product test as well as the exact orthogonality test are satisfied for given constants $\rho > 0$ and $\nu > 0$ at

each iteration. Moreover let the step size satisfy

$$\alpha_k = \alpha \leq \frac{1}{(1 + \rho^2 + \nu^2)L}. \quad (4.35)$$

Then, for $k \in \mathbb{N}$, it holds that

$$\mathbb{E}[\hat{J}(u_{k+1})] \leq \mathbb{E}[\hat{J}(u_k)] - \frac{\alpha}{2} \mathbb{E}[\|\nabla \hat{J}(u_k)\|_2^2]. \quad (4.36)$$

Proof. This proof is based on [BBN17, Lemma 3.1]. As the exact orthogonality test is satisfied, (4.29) holds true for every iterate u_k and the corresponding batch size m_k , which we will, for readability, denote as u and m for the first part of the proof. Therefore

$$\begin{aligned} \nu^2 \|\nabla \hat{J}(u)\|_2^2 &\geq \mathbb{E} \left[\left\| G^m(u) - \frac{G^m(u)^T \nabla \hat{J}(u)}{\|\nabla \hat{J}(u)\|_2^2} \nabla \hat{J}(u) \right\|_2^2 \middle| u \right] \\ &= \mathbb{E}[\|G^m(u)\|_2^2 | u] - \frac{2}{\|\nabla \hat{J}(u)\|_2^2} \mathbb{E}[(G^m(u)^T \nabla \hat{J}(u))^2 | u] \\ &\quad + \frac{1}{\|\nabla \hat{J}(u)\|_2^4} \mathbb{E}[(G^m(u)^T \nabla \hat{J}(u))^2 \nabla \hat{J}(u)^T \nabla \hat{J}(u) | u] \\ &= \mathbb{E}[\|G^m(u)\|_2^2 | u] - \frac{1}{\|\nabla \hat{J}(u)\|_2^2} \mathbb{E}[(G^m(u)^T \nabla \hat{J}(u))^2 | u]. \end{aligned}$$

Thus, it holds that

$$\mathbb{E}[\|G^m(u)\|_2^2 | u] \leq \nu^2 \|\nabla \hat{J}(u)\|_2^2 + \frac{1}{\|\nabla \hat{J}(u)\|_2^2} \mathbb{E}[(G^m(u)^T \nabla \hat{J}(u))^2 | u]. \quad (4.37)$$

As the inner product test (4.26) holds true we can bound the second part of the right-hand side of (4.37) as

$$\begin{aligned} \rho^2 \|\nabla \hat{J}(u)\|_2^4 &\geq \mathbb{E}[(G^m(u)^T \nabla \hat{J}(u) - \|\nabla \hat{J}(u)\|_2^2)^2 | u] \\ &= \mathbb{E}[(G^m(u)^T \nabla \hat{J}(u))^2 | u] - 2\|\nabla \hat{J}(u)\|_2^2 \mathbb{E}[G^m(u)^T \nabla \hat{J}(u) | u] + \|\nabla \hat{J}(u)\|_2^4 \\ &= \mathbb{E}[(G^m(u)^T \nabla \hat{J}(u))^2 | u] - 2\|\nabla \hat{J}(u)\|_2^2 \|\nabla \hat{J}(u)\|_2^2 + \|\nabla \hat{J}(u)\|_2^4 \\ &= \mathbb{E}[(G^m(u)^T \nabla \hat{J}(u))^2 | u] - \|\nabla \hat{J}(u)\|_2^4, \end{aligned}$$

where we used that

$$\mathbb{E}[G^m(u)^T \nabla \hat{J}(u) | u] = \nabla \hat{J}(u)^T \nabla \hat{J}(u). \quad (4.38)$$

Therefore it holds that

$$\mathbb{E}[(G^m(u)^T \nabla \hat{J}(u))^2 | u] \leq (1 + \rho^2) \|\nabla \hat{J}(u)\|_2^4. \quad (4.39)$$

By using (4.39) in (4.37) we obtain

$$\mathbb{E}[\|G^m(u)\|_2^2 | u] \leq (1 + \nu^2 + \rho^2) \|\nabla \hat{J}(u)\|_2^2. \quad (4.40)$$

Now we resume to denote the iterate in iteration k as u_k as well as the batch size as m_k .

Using the Taylor formula, see [DR11, Satz 9.12], on

$$\phi: [0, \alpha] \rightarrow \mathbb{R}, x \mapsto \hat{J}(u_k - sG^{m_k}(u_k))$$

we obtain that there exists a $c \in (0, \alpha)$ such that

$$\begin{aligned} \hat{J}(u_{k+1}) &= \phi(\alpha) = \phi(0) + \alpha\phi'(0) + \frac{\alpha^2}{2}\phi''(c) \\ &= \hat{J}(u_k) - \alpha G^{m_k}(u_k)^T \nabla \hat{J}(u_k) \\ &\quad + \frac{\alpha^2}{2} G^{m_k}(u_k)^T \nabla^2 \hat{J}(u_k - cG^{m_k}(u_k)) G^{m_k}(u_k) \end{aligned} \quad (4.41)$$

where we also used that the iterates are interior points.

Taking the expected value conditioned on u_k in (4.41) we obtain

$$\begin{aligned} \mathbb{E}[\hat{J}(u_{k+1}) | u_k] &= \hat{J}(u_k) - \alpha \mathbb{E}[G^{m_k}(u_k)^T \nabla \hat{J}(u_k) | u_k] \\ &\quad + \frac{\alpha^2}{2} \mathbb{E}[G^{m_k}(u_k)^T \nabla^2 \hat{J}(u_k - cG^{m_k}(u_k)) G^{m_k}(u_k) | u_k] \\ &\leq \hat{J}(u_k) - \alpha \|\nabla \hat{J}(u_k)\|_2^2 + \frac{L\alpha^2}{2} \mathbb{E}[\|G^{m_k}(u_k)\|_2^2 | u_k] \\ &\leq \hat{J}(u_k) - \alpha \|\nabla \hat{J}(u_k)\|_2^2 + \frac{L\alpha^2}{2} (1 + \nu^2 + \rho^2) \|\nabla \hat{J}(u_k)\|_2^2 \\ &\leq \hat{J}(u_k) - \frac{\alpha}{2} \|\nabla \hat{J}(u_k)\|_2^2 \end{aligned} \quad (4.42)$$

where we used the bound on the second derivative of \hat{J} and (4.38) in the first inequality, (4.40) in the second inequality and (4.35) in the last inequality.

Taking the expected value in (4.42) yields (4.36). \square

We are now able to state the convergence of the function values towards the optimal function value $\hat{J}(\bar{u})$.

Theorem 4.9. *Let the assumptions of Lemma 4.8 hold. Assume that additionally there exists a constant μ with $0 < \mu \leq L$ which is a lower bound of the smallest eigenvalue of the symmetric matrix $\nabla^2 \hat{J}(u)$ for all $u \in \mathcal{U}_{\text{ad}}$. Then it holds for every $k \in \mathbb{N}$ that*

$$\mathbb{E}[\hat{J}(u_k) - \hat{J}(\bar{u})] \leq (1 - 2\mu\alpha)^k (\hat{J}(u_0) - \hat{J}(\bar{u})).$$

Proof. This proof is based on [BBN17, Theorem 3.2]. Note that, by assumption, \hat{J} is μ -strongly convex, see [Vol19, Satz 3.5]. With Theorem 2.19 it therefore holds for all k that

$$\|\nabla \hat{J}(u_k)\|_2^2 \geq 4\mu(\hat{J}(u_k) - \hat{J}(\bar{u})). \quad (4.43)$$

Subtracting $\hat{J}(\bar{u})$ on both sides of (4.36) gives us

$$\begin{aligned} \mathbb{E}[\hat{J}(u_{k+1}) - \hat{J}(\bar{u})] &\leq \mathbb{E}[\hat{J}(u_k) - \hat{J}(\bar{u})] - \frac{\alpha}{2} \mathbb{E}[\|\nabla \hat{J}(u_k)\|_2^2] \\ &\leq \mathbb{E}[\hat{J}(u_k) - \hat{J}(\bar{u})] - \frac{\alpha}{2} \left(4\mu \mathbb{E}[\hat{J}(u_k) - \hat{J}(\bar{u})] \right) \\ &= \mathbb{E}[\hat{J}(u_k) - \hat{J}(\bar{u})] (1 - 2\mu\alpha), \end{aligned} \quad (4.44)$$

where we used (4.43) for the second inequality. Applying (4.44) iteratively over k yields the claim. \square

This theorem shows that the expected difference of the function value $\hat{J}(u_k)$ at iteration k to the optimal function value depends on the starting point u_0 as well as on the convexity parameter μ and the chosen step size α . A large step size, a strongly convex function or a good starting point lead to a low expected difference. Now we will show how the expected difference of the current iterate u_k to the optimal solution \bar{u} can be bounded.

Corollary 4.10. *Let the assumptions of Theorem 4.9 hold true. Then there exists a constant $\tilde{C} > 0$ such that for $k \in \mathbb{N}$ it holds that*

$$\mathbb{E}[\|u_k - \bar{u}\|_2] \leq \tilde{C} \sqrt{(1 - 2\mu\alpha)^k}.$$

Proof. Using Theorem 2.23 and the μ -strong convexity of \hat{J} we get that

$$\|u_k - \bar{u}\|_2^2 \leq \frac{1}{\mu} (\hat{J}(u_k) - \hat{J}(\bar{u}) - \nabla \hat{J}(\bar{u})^T (u_k - \bar{u})) \leq \frac{1}{\mu} (\hat{J}(u_k) - \hat{J}(\bar{u})). \quad (4.45)$$

By defining $\tilde{C} := \sqrt{\frac{\hat{J}(u_0) - \hat{J}(\bar{u})}{\mu}}$ and using (4.45) and Theorem 4.9 we get

$$\mathbb{E}[\|u_k - \bar{u}\|_2^2] \leq \frac{1}{\mu} \mathbb{E}[\hat{J}(u_k) - \hat{J}(\bar{u})] \leq \tilde{C}^2 (1 - 2\mu\alpha)^k.$$

The claim follows by using Jensen's inequality (Theorem 2.32) with $f(x) = x^2$. \square

Chapter 5

Numerical Experiments and Results

In this chapter, the setting for the numerical experiments will be described. Then, for each of the stochastic descent methods, we will test different parameter choices regarding computational time and convergence behaviour on this specific problem. Finally, we will compare the stochastic descent methods with the best parameter choice we have found against each other and the deterministic Gradient Descent Method with projected Armijo line search, which is briefly summarized in 6 in the appendix.

Simulations were again run on FEniCS [Aln+15] on a laptop with Intel Core i7-8650u Processor with 16 GB RAM.

For the experiments the domain $\mathcal{D} = (0, 1)^2$ and the admissible set

$$\mathcal{U}_{\text{ad}} = \{u = (u_i)_{1 \leq i \leq n_u} \in \mathbb{R}^{n_u} \mid -1 \leq u \leq 1 \in \mathbb{R}^{n_u}\},$$

with $n_u = 4$ is used. Further, the control shape functions are given by indicator functions

$$b_i : \mathcal{D} \rightarrow \mathbb{R}, x \mapsto \mathbb{1}_{B_i}(x) \quad \text{for } i \in \{1, \dots, n_u\},$$

where $B_i \subset \mathcal{D}$ such that

$$B_i \cap B_j = \emptyset \quad (i \neq j) \quad \bigcup_{i=1}^{n_u} B_i = \mathcal{D}.$$

More specific, for the case $n_u = 4$ we will use

$$\begin{aligned}
B_1 &= (0, 0.5] \times (0, 0.5], \\
B_2 &= (0.5, 1) \times (0, 0.5], \\
B_3 &= (0, 0.5] \times (0.5, 1), \\
B_4 &= (0.5, 1) \times (0.5, 1).
\end{aligned}$$

Note that this choice of control shape functions leads to

$$\left\| \sum_{i=1}^4 u_i b_i \right\|_H^2 = \frac{1}{4} \|u\|_2^2.$$

Furthermore, the functions a and f are as described in Example 3.6 and the desired state is given by

$$y^d(x) = -\sin(2\pi x_1) \sin(2\pi x_2). \tag{5.1}$$

Further $\lambda = 0.5$ is used. The space \mathcal{D} is, as in Example 3.6, discretized by the FE method with piecewise linear elements and 961 vertices.

As a random vector we will use $\xi : \Omega \rightarrow \mathbb{R}^4$, which is again normally distributed in every component with an expected value of two, a standard deviation of 0.5 and truncated between 0.5 and 3.5. Furthermore, as in Example 3.6, the components are uncorrelated and therefore independent. Further, we will use Theorem 3.9 and the discussion thereafter to generate a Laplace space in \mathbb{R}^m by using $K = 500$ realizations of ξ , which we will denote as $\Omega_K = \{\xi_1, \dots, \xi_K\}$.

With the obtained Laplace space $(\Omega_K, 2^{\Omega_K}, \mathbb{P}_K)$, $(\hat{\mathbf{P}})$ is discretized as

$$\min_{u \in \mathcal{U}_{\text{ad}}} j(u) := \frac{1}{2K} \sum_{i=1}^K \|y_u(\cdot; \xi_i) - y^d\|_H^2 + \frac{\lambda}{2} \left\| \sum_{i=1}^4 u_i b_i \right\|_H^2, \tag{\hat{\mathbf{P}}_{\text{num}}}$$

where y_u is the solution to

$$\begin{aligned}
-\nabla \cdot (a(\mathbf{x}; \tilde{\xi}) \nabla y_u(\mathbf{x}; \tilde{\xi})) &= f(\mathbf{x}; \tilde{\xi}) + \sum_{i=1}^{n_u} u_i b_i(\mathbf{x}), & (\mathbf{x}, \tilde{\xi}) \in \mathcal{D} \times \Omega_K, \\
y_u(\mathbf{x}; \tilde{\xi}) &= 0, & (\mathbf{x}, \tilde{\xi}) \in \partial\mathcal{D} \times \Omega_K.
\end{aligned}$$

As in (3.11), the associated (full) gradient is given as

$$\nabla j(u) = \lambda W u - \left(\frac{1}{K} \sum_{j=1}^K \langle p_u(\cdot; \xi_j), b_i \rangle_H \right)_{1 \leq i \leq n_u} \in \mathcal{U} \quad \text{for } u \in \mathcal{U}_{\text{ad}}.$$

The evaluation of the gradient requires solving the adjoint and hence also the state equation for every realization used. Thus, utilizing the non-stochastic gradient descent method is computationally expensive provided K or the number of FE vertices is large. Therefore, as mentioned at the beginning of Chapter 4, the use of stochastic descent methods is necessary to solve this problem in a reasonable time.

As $(\hat{\mathbf{P}}_{\text{num}})$ satisfies the conditions stated in Chapter 4 for $(\hat{\mathbf{P}})$, there exists an optimal solution and the established convergence results hold true.

To start, we have a look at the "optimal" solution \bar{u} , which is a solution computed with the projected Gradient Descent method with projected Armijo line search, see Algorithm 6 in the appendix.

The obtained solution, as well as the desired state, are displayed in Figure 5.1.

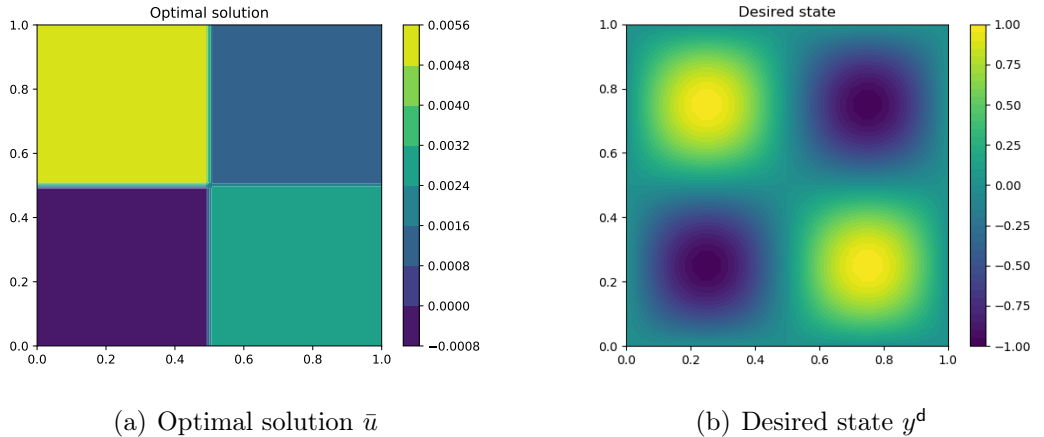


Figure 5.1: Optimal solution \bar{u} to $(\hat{\mathbf{P}}_{\text{num}})$ for $n_u = 4$ and the desired state y^d .

5.1 Projected Stochastic Gradient

We will start with the Projected Stochastic Gradient (PSG), for which we have established an expected convergence of the iterates towards the optimal solution with respect to $\|\cdot\|_2$ with a rate of $\mathcal{O}(k^{-0.5})$ and an expected convergence of the function values with respect to $|\cdot|$ towards the optimal function value with a rate of $\mathcal{O}(k^{-1})$ in Theorem 4.2 and Corollary 4.3. We will try different step sizes of the form

$\alpha_k = \frac{\vartheta}{k}$, for which, as stated in Theorem 4.2, it needs to hold that $\vartheta > 1/\lambda = 2$. We will use $\vartheta \in \{2.5, 10, 25\}$ and a maximum number of iterations of 600. The obtained convergences are displayed in Figure 5.2.

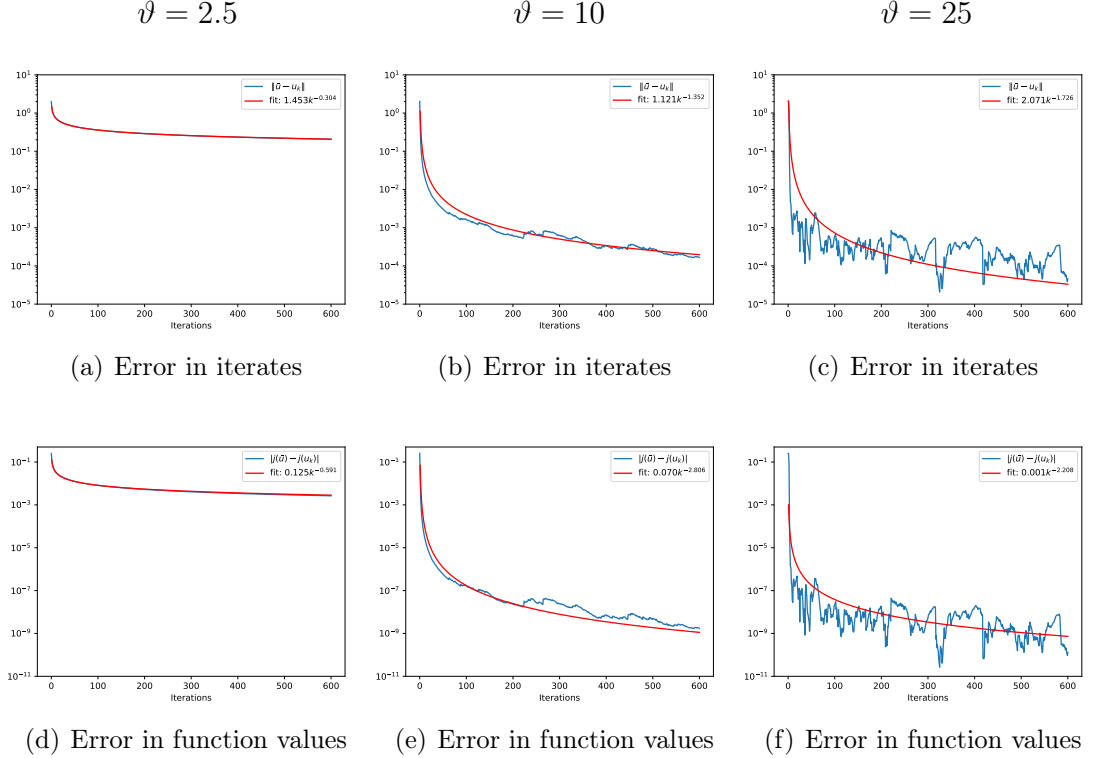


Figure 5.2: Convergences of the iterates towards the optimal control \bar{u} measured by $\|\bar{u} - u_k\|_2$ and of the corresponding function value towards $j(\bar{u})$ for the PSG method with different choices of the step size parameter ϑ . The domain \mathcal{D} is discretized with 961 vertices and Ω_K has a size of 500.

Notice that, as the iterates are random, running the same algorithm another time will not lead to exactly the same results, but the general performance will be similar. Figure 5.2 shows that for all chosen ϑ the iterates converge towards the optimal solution. The two major differences between the three runs are on the one hand the convergence rate and on the other hand the volatility of the difference towards the optimal solution throughout the iterations.

For the smallest ϑ considered, $\vartheta = 2.5$, we observe the lowest rate of convergence, but also the lowest volatility in the sense that the difference of the current iterate towards the optimal solution decreases smoothly without large downwards or upwards jumps. In contrast to that, for $\vartheta = 25$, we obtain a high convergence rate but also the most volatile convergence behaviour as there are many upward and downward jumps in the difference towards the optimal solution. This results in many iterates not being as close to the optimal solution as the convergence rate suggests for this specific

iterate.

A way to decrease volatile behaviour is to adjust the batch size. A higher batch size will lead to a smaller volatility, but of course also to a higher computational effort. As the other two methods will already use larger batch sizes we will leave it at a batch size of one for the PSG method.

Another point of interest is the time required to reach the optimal solution up to certain precision levels. The average required computational time to reach a precision of 10^{-3} and 10^{-6} and the corresponding coefficient of variation, see Definition 2.35, of three runs for each parameter choice, are listed in Table 5.1.

| | Time (in sec.) to reach | |
|-------------------|------------------------------------|------------------------------------|
| | $\ u_k - \bar{u}\ _2 \leq 10^{-3}$ | $\ u_k - \bar{u}\ _2 \leq 10^{-6}$ |
| $\vartheta = 2.5$ | not reached | not reached |
| $\vartheta = 10$ | 10.63 (0.11) | not reached |
| $\vartheta = 25$ | 0.98 (0.21) | not reached |

Table 5.1: Average computational time and the corresponding coefficient of variation in parenthesis to reach the optimal control up to a precision of 10^{-3} and 10^{-6} for three runs of the PSG method for different choices of the step size parameter ϑ .

Table 5.1 confirms the behaviour we have seen in Figure 5.2. A larger step size will lead to a faster convergence on average but also to a larger coefficient of variation, i.e. a larger relative volatility. We use the coefficient of variation instead of the volatility as the means of the quantities we are comparing differ significantly.

It seems that a moderate step size parameter, such as $\vartheta = 10$, provides a good trade off between a high convergence rate and a low coefficient of variation for the problem $(\hat{\mathbf{P}}_{\text{num}})$.

5.2 Projected Stochastic Variance Reduced Gradient

Now we will have a look at the Projected Stochastic Variance Reduced Gradient (PSVRG) method. In Theorem 4.6 and Corollary 4.7 we have shown that the expected convergence rate of the controls towards the optimal control with respect to $\|\cdot\|_2$ is $\mathcal{O}((1-\theta)^{\frac{k}{2}})$ and that the function values converge towards the optimal function value with respect to $|\cdot|$ in expectation with a convergence rate of $\mathcal{O}((1-\theta)^k)$.

As an initial step size we will choose $\alpha_0 = 1$, the step sizes thereafter will be computed with the Barzilai-Borwein procedure. We will use batch sizes $m \in \{25, 100, 250\}$ to

test how the batch size influences the convergence behaviour. One may notice that we do not use a batch size which fulfils the requirements of Theorem 4.6. This is due to the small probability space which we use here ($K = 500$), for which we would get unreasonably large batch sizes when choosing m such that (4.20) holds, which would slow down the convergence. Nevertheless, we are able to observe convergence rates of the form stated in Theorem 4.6 and Corollary 4.7, see Figure 5.3.

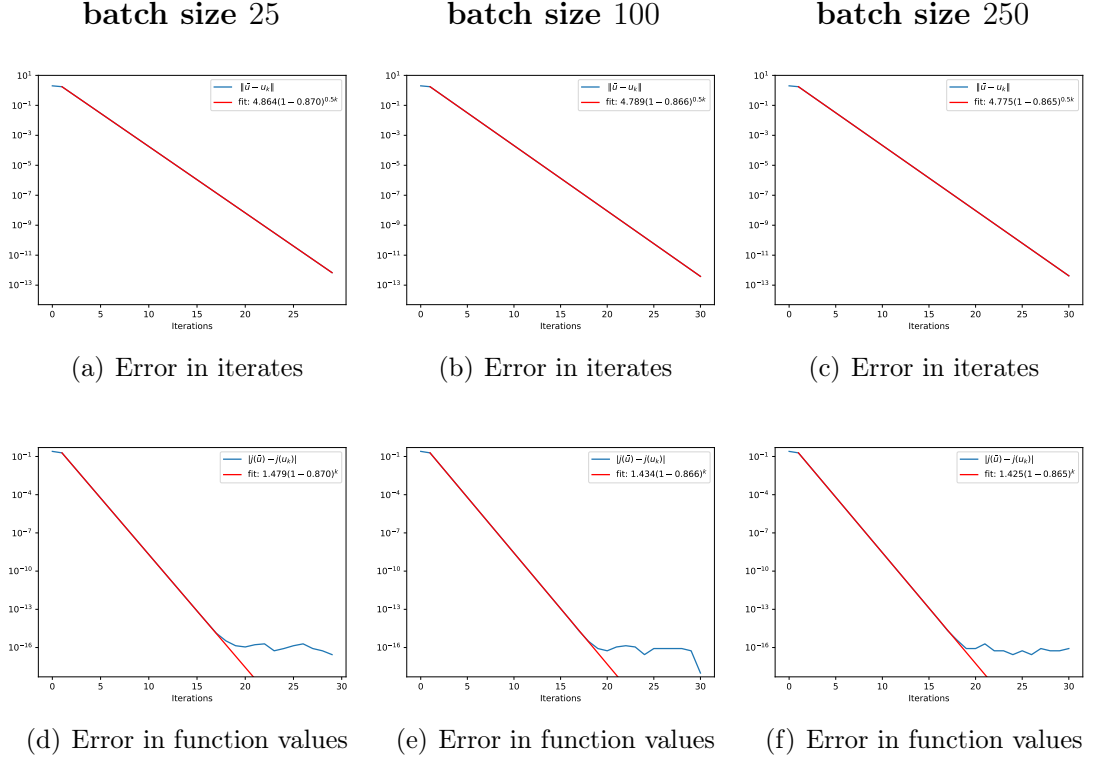


Figure 5.3: Convergence of the iterates towards the optimal control \bar{u} measured by $\|\bar{u} - u_k\|_2$ and of the corresponding function value towards $j(\bar{u})$ for the PSVRG method. The domain \mathcal{D} is discretized with 961 vertices and Ω_K has a size of 500.

We see that for all batch sizes a convergence rate of the form $\mathcal{O}((1 - \theta)^{\frac{k}{2}})$ for the iterates and $\mathcal{O}((1 - \theta)^k)$ for the function values is obtained and that there is almost no deviation from the fitted convergence line (except when the difference is as small as the machine precision of about 10^{-16}), in comparison to the results of the PSG method.

The highest convergence rate here is obtained for a batch size of 25. This is also confirmed by the required computational times listed in Table 5.2.

| | Time in seconds to reach | | |
|----------------|------------------------------------|------------------------------------|------------------------------------|
| | $\ u_k - \bar{u}\ _2 \leq 10^{-3}$ | $\ u_k - \bar{u}\ _2 \leq 10^{-6}$ | $\ u_k - \bar{u}\ _2 \leq 10^{-9}$ |
| batch size 25 | 338.89 (0.02) | 576.12 (0.02) | 779.45 (0.02) |
| batch size 100 | 385.46 (0.01) | 655.29 (0.01) | 925.11 (0.01) |
| batch size 250 | 494.89 (0.03) | 841.31 (0.03) | 1187.73 (0.03) |

Table 5.2: Average computational time and the corresponding coefficient of variation in parenthesis to reach the optimal control up to a precision of 10^{-3} , 10^{-6} , 10^{-9} for three runs of the PSVRG method for different choices of the batch size.

Next to the average computational time, which is, as expected from Figure 5.3 the lowest for a batch size of 25, we can see that the coefficient of variation is overall low. This is due to a full gradient being used in every iteration and thus obtaining low volatility in the convergence behaviour.

5.3 Projected Stochastic Adaptive Sampling

Lastly, the Projected Stochastic Adaptive Sampling (PSAS) method will be examined. For the PSAS method we have shown that the iterates converge to the optimal control in expectation with a convergence rate of $\mathcal{O}((1 - 2\lambda\alpha)^{\frac{k}{2}})$ and that the function values converge towards the optimal function value with a convergence rate of $\mathcal{O}((1 - 2\lambda\alpha)^k)$ in expectation. The parameters are set as proposed in [BBN17] as $\rho = 0.9$, $\nu = 5.84$, $r = 10$, $\gamma = 0.38$ and we will start with a batch size of 2.

We will use the fixed step sizes $\alpha_k = \alpha \in \{2, 4, 8\}$. Note that these step sizes do not fulfil condition (4.35) of Lemma 4.8. We would need to choose much smaller step sizes (in the range of 10^{-3}) to satisfy this condition, which would drastically slow down the convergence. We therefore focus on larger step sizes and are still able to obtain converge rates of the form stated in Theorem 4.9 and Corollary 4.10, as we can see in Figure 5.4.

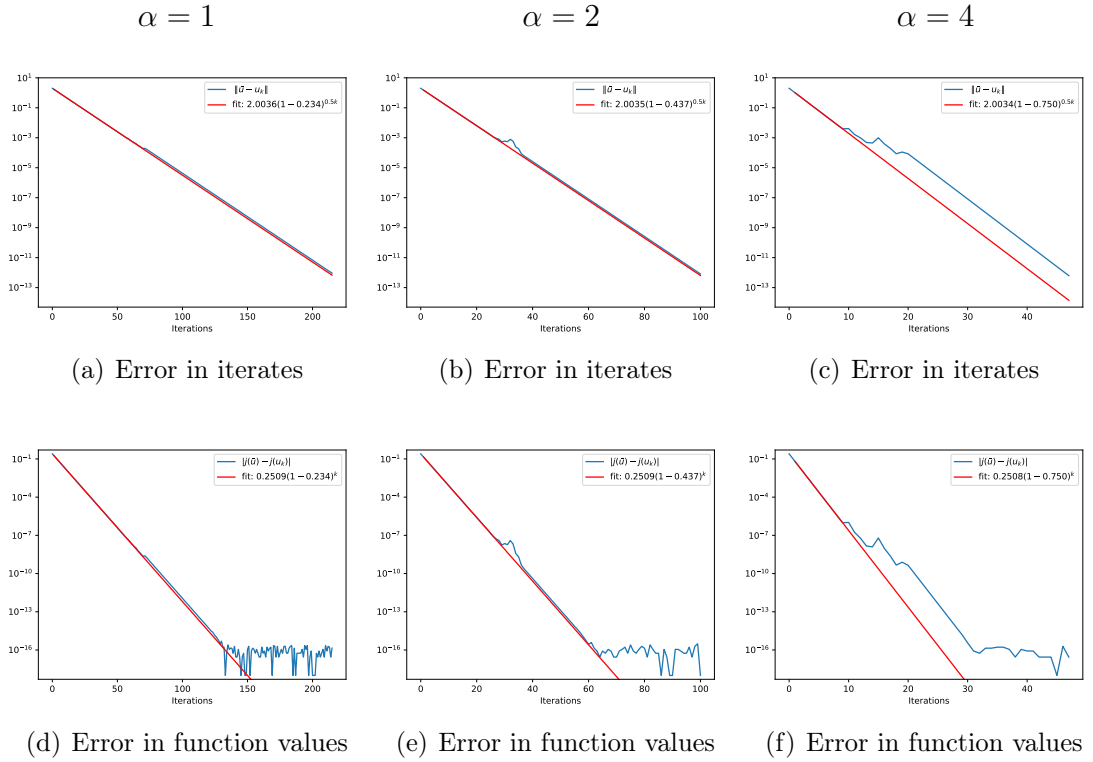


Figure 5.4: Convergence of the iterates towards the optimal control \bar{u} measured by $\|\bar{u} - u_k\|_2$ and of the corresponding function value towards $j(\bar{u})$ for the PSAS method. The domain \mathcal{D} is discretized with 961 vertices and Ω_K has a size of 500.

Figure 5.4 shows that a larger step size leads to a higher convergence rate but also to a larger deviation from the fitted convergence line, similar to the convergence behaviour we have seen for the PSG method. This can also be seen in Table 5.3. Nevertheless, a larger step size seems to be a better choice for the problem $(\hat{\mathbf{P}}_{\text{num}})$ as a significantly lower computational time for all precision levels is reached.

| | Time (in sec.) to reach | | |
|--------------|------------------------------------|------------------------------------|------------------------------------|
| | $\ u_k - \bar{u}\ _2 \leq 10^{-3}$ | $\ u_k - \bar{u}\ _2 \leq 10^{-6}$ | $\ u_k - \bar{u}\ _2 \leq 10^{-9}$ |
| $\alpha = 1$ | 45.83 (0.01) | 1529.51 (0.03) | 3336.14 (0.02) |
| $\alpha = 2$ | 23.98 (0.07) | 705.39 (0.08) | 1664.63 (0.04) |
| $\alpha = 4$ | 9.61 (0.13) | 343.12 (0.10) | 689.91 (0.05) |

Table 5.3: Average computational time and the corresponding coefficient of variation in parenthesis to reach the optimal control up to a precision of 10^{-3} , 10^{-6} , 10^{-9} for three runs of the PSAS method for different choices of the fixed step size α .

In the case of the PSAS method one can also have a look at the evolution of the batch size compared to the evolution of the difference of the current iterate to the optimal solution. This is displayed in Figure 5.5.

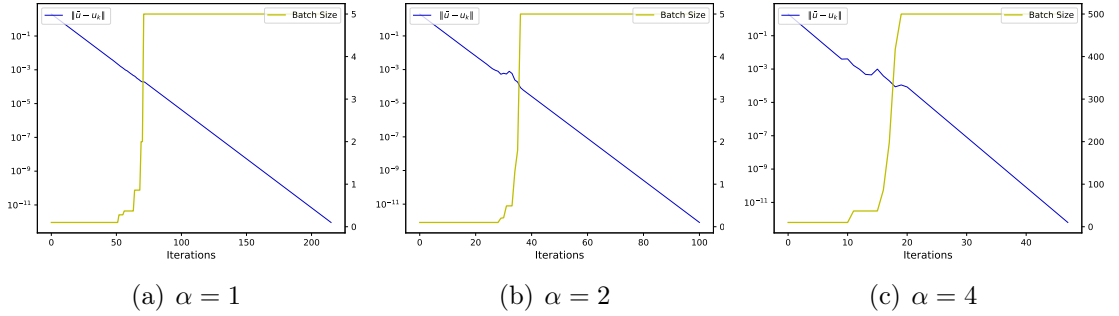


Figure 5.5: Convergence of the iterates towards the optimal control \bar{u} measured by $\|\bar{u} - u_k\|_2$ and the corresponding batch size at that iteration for three runs of the PSAS method.

We can see that the batch size for all of the runs increases slowly until the distance of the current iterate to the optimal solution reaches a precision of about 10^{-4} , then the batch size increases in a few iterations up to the full batch size. It can be interpreted that, in a region around the optimal solution of the size 10^{-4} , the small batch size is not sufficient to reliably compute a direction which leads to descent and thus the batch size is increased. The mentioned region can also be observed for the PSG method in Figure 5.2, a volatile behaviour of the iterates around a precision of around 10^{-4} is visible where for the step size parameter $\theta = 25$.

We can additionally see that many iterations with the full batch size have to be computed, which are the key contributor to the computational effort, see Table 5.3. As mentioned before, the Barzilai-Borwein step size procedure is not leading to good results when using small batch sizes. As the PSAS method increases the batch size over the iterations, it might be worth using a fixed step size up to a point where a sufficiently large batch size is reached and then switch to the Barzilai-Borwein step size. This could help avoiding many steps with the full batch size, which we see in Figure 5.5. We will test this modified PSAS method and use the Barzilai-Borwein step size when the batch size reaches 70% of the size of the probability space.

The results are displayed in Figure 5.6, in which we can see that the convergence behaviour at the beginning is similar to the one of the PSAS method but after reaching a precision of about 10^{-4} the difference to the optimal control greatly decreases in only a few iterations.

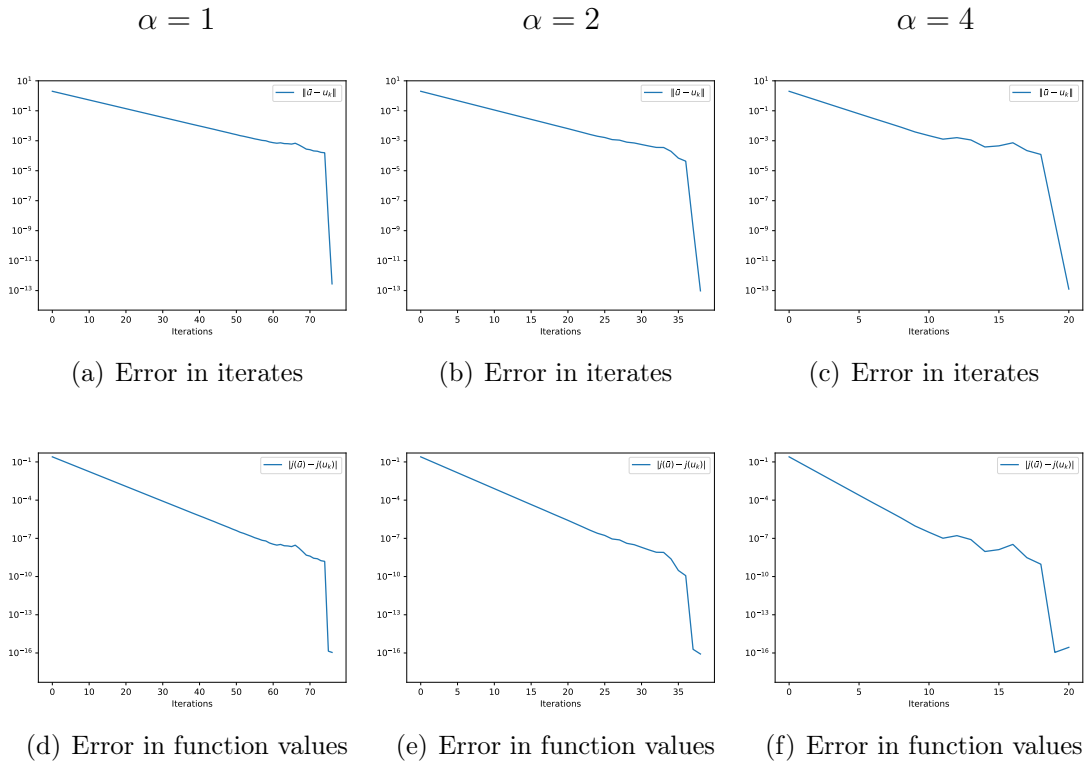


Figure 5.6: Convergence of the iterates towards the optimal control \bar{u} measured by $\|\bar{u} - u_k\|_2$ and of the corresponding function value towards $j(\bar{u})$ for the modified PSAS method. The domain \mathcal{D} is discretized with 961 vertices and Ω_K has a size of 500.

As we will see in Figure 5.7, this is due to the switch to the Barzilai-Borwein step size procedure. This leads to a reduced computational time in comparison to the regular PSAS method. This is also visible when comparing the computational times listed in Table 5.4 with the ones of the regular PSAS method in Table 5.3.

| | Time (in sec.) to reach | | |
|--------------|------------------------------------|------------------------------------|------------------------------------|
| | $\ u_k - \bar{u}\ _2 \leq 10^{-3}$ | $\ u_k - \bar{u}\ _2 \leq 10^{-6}$ | $\ u_k - \bar{u}\ _2 \leq 10^{-9}$ |
| $\alpha = 1$ | 54.75 (0.03) | 284.61 (0.1) | 328.10 (0.1) |
| $\alpha = 2$ | 25.00 (0.10) | 148.65 (0.15) | 188.23 (0.12) |
| $\alpha = 4$ | 10.75 (0.15) | 132.31 (0.23) | 157.12 (0.10) |

Table 5.4: Average computational time and the corresponding coefficient of variation in parenthesis to reach the optimal control up to a precision of 10^{-3} , 10^{-6} , 10^{-9} for three runs of the modified PSAS method for different choices of the fixed starting step size α .

We can see that for a precision level of 10^{-3} , there is no significant difference to the regular PSAS method, which is not surprising as the modification only makes

a difference when large batch sizes are used. For the higher precision levels the modified PSAS method obtains lower computational times, especially for the smaller step sizes, but also the coefficient of variation is larger. As for the PSAS method, we have a look at the evolution of the batch size compared to the evolution of the difference of the current iterate to the optimal solution. This is displayed in Figure 5.7.

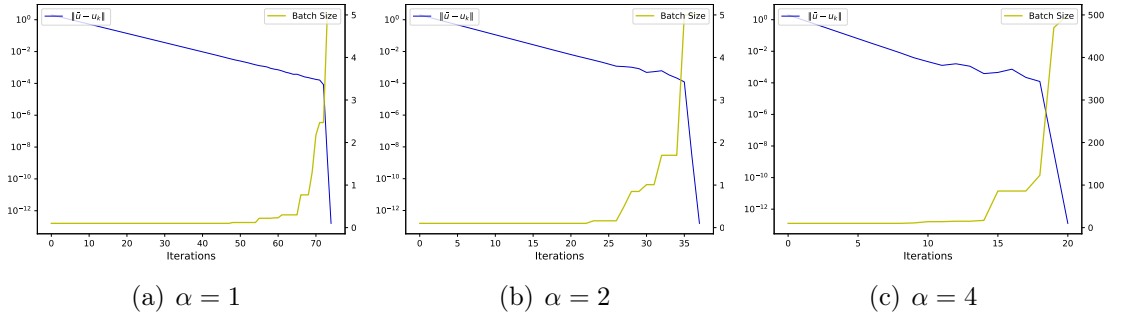


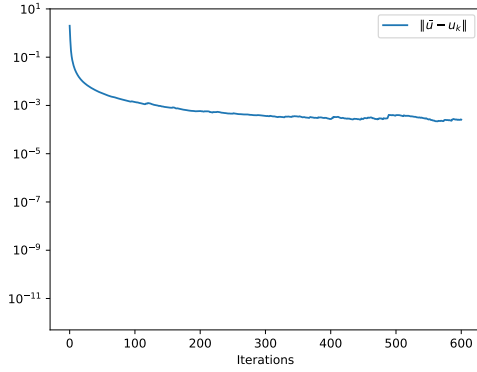
Figure 5.7: Convergence of the iterates towards the optimal control \bar{u} measured by $\|\bar{u} - u_k\|_2$ and the corresponding batch size at that iteration for three runs of the modified PSAS method.

We can see that the modification works as planned, as there are very few iterations in which the full gradient is used. This is the reason for the drastically reduced computational times in Table 5.4. Therefore, it seems as the modified PSAS method is a better choice than the regular PSAS method.

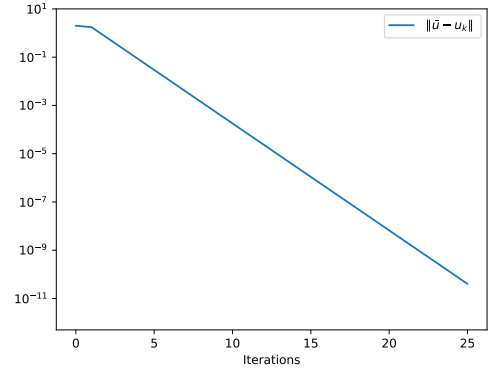
5.4 Comparison of the Descent Methods

In this section, we will compare the three stochastic descent methods PSG, PSVRG and modified PSAS, as well as the deterministic projected Gradient Descent (PGD) method with the projected Armijo line search to see which method performs best on $(\hat{\mathbf{P}}_{\text{num}})$ regarding computational time and precision.

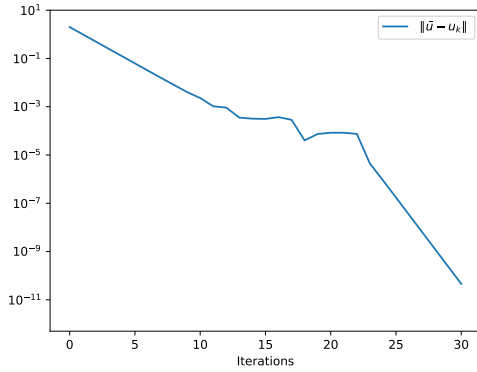
We will use a Laplace space with $K = 5000$ for this comparison. Furthermore, the parameter $\vartheta = 10$ for the PSG method and a batch size of 25 for the PSVRG method will be used. For the modified PSAS method we will use a fixed step size $\alpha = 4$ until the batch size reaches 70% of the size of Ω_K , i.e. $0.7 \cdot 5000 = 3500$, then the Barzilai-Borwein step size is used. The convergence of the iterates towards the optimal control of each of the methods is displayed in Figure 5.8.



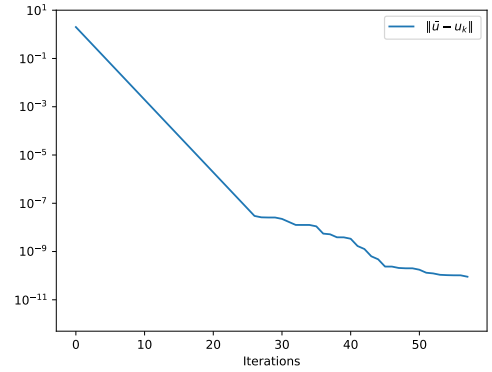
(a) PSG method; $\vartheta = 10$



(b) PSVRG method; batch size 25



(c) Modified PSAS method; $\alpha = 4$ until the batch size reaches 70% of the size of Ω_K , then BB step size



(d) PGD method with projected Armijo line search

Figure 5.8: Convergence of the iterates towards the optimal control \bar{u} measured by $\|\bar{u} - u_k\|_2$ for the descent methods. The domain \mathcal{D} is discretized with 961 vertices and Ω_K has a size of 5000.

Overall we can see the same convergence behaviour of the stochastic descent methods as in the previous section. For the projected Gradient Descent method the convergence slows down after reaching a precision of about 10^{-7} . This is likely due to the small step sizes generated with the projected Armijo line search when the function values are close to the minimal function value.

Except for the PSG method, all of the methods reach a precision of 10^{-10} . Next to the ability to reach a high precision, the required computational times, which are listed in Table 5.5, are of interest.

| | Time (in sec.) to reach | | |
|-----------|------------------------------------|------------------------------------|------------------------------------|
| | $\ u_k - \bar{u}\ _2 \leq 10^{-3}$ | $\ u_k - \bar{u}\ _2 \leq 10^{-6}$ | $\ u_k - \bar{u}\ _2 \leq 10^{-9}$ |
| PSG | 9.95 (0.14) | not reached | not reached |
| PSVRG | 3476.82 (0.02) | 5910.60 (0.02) | 7996.69 (0.02) |
| mod. PSAS | 9.40 (0.1) | 1489.19 (0.19) | 3032.98 (0.09) |
| GD | 7824.32 (0) | 14344.59 (0) | 69332.42 (0) |

Table 5.5: Average computational time and the corresponding coefficient of variation in parenthesis to reach the optimal control up to a precision of 10^{-3} , 10^{-6} , 10^{-9} for three runs of each of the three stochastic descent methods and the deterministic projected Gradient Descent method with projected Armijo line search.

We can see that all of the models beat the deterministic projected Gradient Method with projected Armijo line search when it comes to the computational time required until a precision of 10^{-3} is reached. The PSG method fails to reach a precision of 10^{-6} . The PSVRG and the modified PSAS also reach 10^{-6} and 10^{-9} significantly faster than the PGD method. The modified PSAS method reaches all of the considered precision levels the fastest, but most of the time also obtains the highest coefficient of variation. Nevertheless, it seems that the modified PSAS method is the best choice for $(\hat{\mathbf{P}}_{\text{num}})$ as it is able to reach a region of the solution fast on the hand, but is also able to obtain a high precision on the other hand, while still being faster than the other methods considered.

Chapter 6

Conclusion and Outlook

In this thesis, we have introduced a convex, elliptic PDE-constrained and box-constrained optimal control problem with uncertain coefficients and have shown that a globally optimal solution exists. We have further introduced three stochastic descent methods, the Projected Stochastic Gradient (PSG) method, the Projected Stochastic Variance Reduced Gradient (PSVRG) method and the Projected Stochastic Adaptive Sampling (PSAS) method including the relevant convergence analysis. Then we have conducted numerical tests which led to an overall comparison of the convergence behaviour as well as the required computational time of the three stochastic descent methods and the deterministic Gradient Descent method with the projected Armijo line search.

We found that the PSG method can be used to quickly obtain a solution when a relatively low precision is sufficient. The PSVRG method in combination with the Barzilai-Borwein step size is reliable in converging to the optimal solution in the sense that it reaches arbitrarily close to the optimal solution and that there is low variation in the convergence behaviour. The downside is that the computational time is high compared to the other stochastic methods. The PSAS method, especially the modified version which includes the Barzilai-Borwein step size procedure when large batch sizes are used, combines the advantages of the PSG and PSVRG method. It is able to reach a region of the optimal solution quickly and then increases the batch size and uses the Barzilai-Borwein step size to obtain a higher precision. This results in a faster convergence than both of the other stochastic descent methods, which, however, are still able to reach into a region of the solution faster than the deterministic Gradient Descent method with projected Armijo line search.

To conclude, it seems that the modified PSAS method is the best choice for the problem considered in this thesis.

In the context of optimal control under uncertainty, there are many extensions to the problem considered in this thesis, on which further research would be interesting. In general, one could test whether the considered methods perform similarly for non-quadratic problems and include other stochastic descent approaches to the comparison such as the Stochastic Average Gradient method or stochastic second-order approaches.

One could also compare different discretization methods to obtain the functional form (3.2) of a and f such as higher-order Taylor approximations or the Empirical Interpolation Method.

Additionally, one could include time dependence in the model. On the one hand, this would extend the application field. On the other hand, it would lead to a drastically higher computational effort, which would increase the need for further computational time reducing methods such as the Reduced Basis method, which was already briefly introduced in Section 3.1.

Appendix - Projected Gradient Descent

This chapter, which is based on [Kel99, Chapter 5], is used to state the algorithm of the projected Gradient Descent (PGD) method with projected Armijo line search. The Armijo line search is used to ensure a sufficient decrease in the function values. We specify this statement at an iterate u as

$$\hat{J}(u(t)) - \hat{J}(u) \leq \frac{-\gamma}{t} \|u - u(t)\|_2^2, \quad (6.1)$$

where $u(t) := \mathcal{P}(u - t\nabla\hat{J}(u))$.

Algorithm 5 Projected Armijo line search

Require: Iterate $u \in \mathcal{U}_{\text{ad}}$, $\alpha^0 > 0$, $\gamma > 0$, $\beta \in (0, 1)$.

- 1: Set $\alpha = \alpha^0$
 - 2: **while** (6.1) is not satisfied with $t = \alpha$ **do**
 - 3: Set $\alpha = \alpha\beta$
 - 4: **end while**
-

Parameters are chosen as $\alpha^0 = 4$, $\gamma = 10^{-8}$, $\beta = 0.5$.

Then, the PGD method is described in Algorithm 6.

Algorithm 6 Projected Gradient Descent (PGD) with projected Armijo line search

Require: starting value $u_0 \in \mathcal{U}_{\text{ad}}$.

- 1: Set $k = 0$
 - 2: **while** stopping criterion not satisfied **do**
 - 3: Compute step size $\alpha_k > 0$ with projected Armijo line search
 - 4: $u_{k+1} = \mathcal{P}(u_k - \alpha_k \nabla\hat{J}(u_k))$
 - 5: $k = k + 1$
 - 6: **end while**
-

As a stopping criterion we used

$$\|u - \mathcal{P}(u - \nabla \hat{J}(u))\|_2 \leq 10^{-14},$$

where again \mathcal{P} is the projection on the admissible set \mathcal{U}_{ad} .

Bibliography

- [Aln+15] Martin S. Alnæs et al. “The FEniCS Project Version 1.5”. In: *Archive of Numerical Software* 3.100 (2015).
- [Bar+04] Maxime Barrault et al. “An ‘empirical interpolation’ method: application to efficient reduced-basis discretization of partial differential equations”. In: *Comptes Rendus Mathématique* 339.9 (2004), pp. 667–672.
- [BBN17] Raghu Bollapragada, Richard Byrd, and Jorge Nocedal. *Adaptive Sampling Strategies for Stochastic Optimization*. 2017. arXiv: 1710.11258 [math.OC].
- [BCN18] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. *Optimization Methods for Large-Scale Machine Learning*. 2018. arXiv: 1606.04838.
- [Bor17] Andrei N. Borodin. *Stochastic Processes*. Cham: Birkhäuser, 2017.
- [Byr+15] Richard H. Byrd et al. *A Stochastic Quasi-Newton Method for Large-Scale Optimization*. 2015. arXiv: 1401.7020.
- [DR11] Robert Denk and Reinhard Racke. *Kompendium der Analysis. Band 1: Differential- und Integralrechnung, Gewöhnliche Differentialgleichungen*. Vieweg+Teubner Verlag, 2011.
- [DR12] Robert Denk and Reinhard Racke. *Kompendium der Analysis. Band 2: Maß- und Integrationstheorie, Funktionentheorie, Funktionalanalysis, Partielle Differentialgleichungen*. Vieweg+Teubner Verlag, 2012.
- [Dzi10] Gerhard Dziuk. *Theorie und Numerik partieller Differentialgleichungen*. De Gruyter, 2010.
- [GP19] Caroline Geiersbach and Georg Pflug. *Projected Stochastic Gradients for Convex Constrained Problems in Hilbert Spaces*. 2019. arXiv: 1807.09132.
- [HRS16] Jan Hesthaven, Gianluigi Rozza, and Benjamin Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. SpringerBriefs in Mathematics, 2016.

- [JZ13] Rie Johnson and Tong Zhang. “Accelerating Stochastic Gradient Descent using Predictive Variance Reduction”. In: *Advances in Neural Information Processing Systems*. Vol. 26. Curran Associates, Inc., 2013, pp. 315–323.
- [Kel99] Carl T. Kelley. *Iterative methods for optimization*. SIAM, 1999.
- [Kle13] Achim Klenke. *Wahrscheinlichkeitstheorie*. 3. Auflage. Springer-Verlag, 2013.
- [KMR19] Dmitry Kovalev, Konstantin Mishchenko, and Peter Richtárik. *Stochastic Newton and Cubic Newton Methods with Simple Local Linear-Quadratic Rates*. 2019. arXiv: 1912.01597.
- [QMN16] Alfio Quarteroni, Andrea Manzoni, and Federico Negri. *Reduced Basis Methods for Partial Differential Equations: An Introduction*. Vol. 92. Springer International Publishing, 2016.
- [RM51] Herbert Robbins and Sutton Monro. “A Stochastic Approximation Method”. In: *The Annals of Mathematical Statistics* 22.3 (1951), pp. 400–407.
- [Rüs16] Ludger Rüschendorf. *Wahrscheinlichkeitstheorie*. Springer-Verlag, 2016.
- [Sai17] Belkacem Said-Houari. *Linear Algebra*. Birkhäuser, 2017.
- [Sin20] Sebastian Sinnwell. “Reduced-Basis Methods for PDE-Constrained Elliptic Optimal Control Problems with Uncertain Coefficients”. MA thesis. University of Konstanz, 2020.
- [SRB16] Mark Schmidt, Nicolas Le Roux, and Francis Bach. *Minimizing Finite Sums with the Stochastic Average Gradient*. 2016. arXiv: 1309.2388.
- [Tan+16] Conghui Tan et al. *Barzilai-Borwein Step Size for Stochastic Gradient Descent*. 2016. arXiv: 1605.04131.
- [UU11] Michael Ulbrich and Stefan Ulbrich. *Nichtlineare Optimierung*. Springer Basel, 2011.
- [Vol19] Stefan Volkwein. *Optimierung*. Lecture notes. University of Konstanz, 2019.
- [Zho18] Xingyu Zhou. *On the Fenchel Duality between Strong Convexity and Lipschitz Continuous Gradient*. 2018. arXiv: 1803.06573.