RICE UNIVERSITY

Optimization Governed by Stochastic Partial Differential Equations

by

Drew P. Kouri

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Master of Arts

THESIS COMMITTEE:

[illegible]

Matthias Heinkenschloss, Chairman
Professor of Computational and Applied Mathematics

[illegible]

Danny C. Sorensen
Noah G. Harding Professor of Computational and Applied Mathematics

[illegible]

Beatrice M. Riviere
Associate Professor of Computational and Applied Mathematics

Dennis D. Cox

Dennis D. Cox
Professor of Statistics

HOUSTON, TEXAS

MAY, 2010

UMI Number: 1486555

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.

UMI
Dissertation Publishing

UMI 1486555
Copyright 2010 by ProQuest LLC.
All rights reserved. This edition of the work is protected against
unauthorized copying under Title 17, United States Code.

ProQuest

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Abstract

Optimization Governed by Stochastic Partial Differential Equations

by

Drew P. Kouri

This thesis provides a rigorous framework for the solution of stochastic elliptic partial differential equation (SPDE) constrained optimization problems. In modeling physical processes with differential equations, much of the input data is uncertain (e.g. measurement errors in the diffusivity coefficients). When uncertainty is present, the governing equations become a family of equations indexed by a stochastic variable. Since solutions of these SPDEs enter the objective function, the objective function usually involves statistical moments. These optimization problems governed by SPDEs are posed as a particular class of optimization problems in Banach spaces. This thesis discusses Monte Carlo, stochastic Galerkin, and stochastic collocation methods for the numerical solution of SPDEs and identifies the stochastic collocation method as particularly useful for the optimization of SPDEs. This thesis extends the stochastic collocation method to the optimization context and explores the decoupling nature of this method for gradient and Hessian computations.

Acknowledgements

I would like to acknowledge my family, who, through their love and support, have aided in all of my accomplishments. I would like to thank Dr. Jan Hewitt and Dr. William Symes for helping me strengthen the writing of this thesis. Also, I would like to thank Sean Hardesty for answering any question I may have had concerning Linux. I am extremely grateful for all the aid my adviser, Matthias Heinkenschloss, has given me, seeming to always point me in the right direction.

This research was supported by AFOSR grant FA9550-09-1-0225.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Many physical problems can be formulated as optimization problems. These problems are used to gain a more comprehensive understanding of physical systems and typically depend on some model. In many cases, these models are deterministic. In real world application, uncertainty plagues everything from modeling assumptions to experimental data. As such, many practitioners have developed stochastic models to accommodate for these uncertainties. With the addition of randomness to the models, the resulting optimization problems require some additional theory in order to make sense of and solve them. This thesis develops an adjoint based approach to solving optimization problems governed by stochastic partial differential equations (SPDE). This approach allows for the computation of first and second order derivative information. Furthermore, this thesis develops algorithms to exploit the parallel nature of the derivative computations. This derivative information is used to run an inexact Newton's method to solve the optimization problems of interest. Finally, a Hessian approximation is developed based on either a related deterministic problem or a problem in which the stochastic domain is smaller. This approximation leads to a formulation of Newton's method with inexact derivative information as well as a preconditioned Newton-CG method. The novelty of these methods is that they reduce storage immensely and reduce the computational cost of the conjugate gradient method when solving the Newton system.

The topic of SPDE constrained optimization is at the interface of two fields: stochastic partial differential equations and optimization theory. In order to solve an optimization problem governed by an SPDE, one must first solve the SPDE. As such, this thesis is structured in a progressive manner. In the first chapter, I give a comprehensive literature review. This review covers both fields (SPDEs and optimization theory) separately and then I discuss the literature at the interface of the two fields. In the second chapter, I discuss theoretical aspects of SPDEs. This chapter develops existence and uniqueness of solutions to SPDEs, as well as, reviews some necessary assumptions on the inputs of the SPDE. These assumptions will allow for numerical solution to the SPDE. The third chapter of this thesis develops three numerical methods for solving SPDEs. First I present two sampling based approaches (Monte Carlo finite element and the Stochastic Collocation finite element), and finish the chapter with the Stochastic Galerkin finite element method. In the final chapter, I rigorously develop the functional analytic framework required for optimization governed by SPDEs. I present an adjoint based approach for computing the gradient and Hessian information and describe two forms of the inexact Newton's method for solving such optimization problems. As mentioned, the algorithms presented allow for extremely large problems due to its minimal storage requirements and parallel nature. Finally, I conclude with a numerical example demonstrating the necessity of SPDE constrained optimization (as opposed to related deterministic problems) and illustrate the behavior of the inexact Newton's method.

Chapter 2

Literature Review

This thesis develops an adjoint equation approach for computing gradients of objective functions governed by stochastic elliptic partial differential equations (SPDE). This research is at the interface of two fields: stochastic partial differential equations and optimization. In the past century, both topics have been researched extensively, but few have attempted to combined the two subjects. Attacking these optimization problems is a daunting task, both theoretically and numerically. With the ever increasing computational power, numerical exploration of such problems is more reasonable. With a firm understanding of the theoretical aspects concerning these optimization problems, numerical solutions become more attainable. This research presents such theoretical and numerical background to these problems and analyzes the error in derivative computations for specific numerical SPDE methods (i.e. stochastic Galerkin, stochastic collocation, and Monte Carlo finite element methods). Any derivative based optimization algorithm will require the solution of the SPDE at each iteration. Thus, it is essential that one can numerically solve an SPDE efficiently and quickly. As such, this chapter is split into two distinct topics: numerical solutions of SPDEs and optimization. I will conclude the chapter with an overview of the current, albeit sparse, research at the interface of these topics.

2.1 Stochastic Partial Differential Equations

This thesis is concerned with stochastic partial differential equations with random field input data. The solutions to such equations are also random fields; therefore, it is necessary to understand how to represent random fields and processes. Two main representations exist and are implemented in practice. In 1938, Norbert Wiener developed his theory of "homogeneous chaos" or polynomial chaos expansion. The chaos expansion seeks to represent Gaussian random processes as an expansion of Hermite polynomials evaluated at Gaussian random variables [40]. Many recent studies have generated theory for "generalized chaos", which extends Weiner's chaos to non-Gaussian random processes [19, 23, 37]. About a decade following Weiner's chaos theory, Karhunen (1947) and Lòeve (1948) independently discovered an alternative representation of random fields [18, 22]. They proposed a generalized Fourier expansion representation based on the eigenvalues and eigenfunctions of the random field's covariance function. Due to the convergence properties of both representations, one can truncate the resulting series to approximate the random process. This allows the practitioner to represent a given random process arbitrarily well by only a finite number of random variables (called the finite dimensional noise assumption). Also, this finite dimensional representation allows for the investigation of numerical solutions and computer representation of SPDEs.

Early methods for solving stochastic partial differential equations included perturbation and Neumann expansion methods. Perturbation methods expand the solution to the stochastic operator equation in a Taylor's series and solve for the derivatives of the solution of the SPDE. This method requires undesirably harsh smoothness constraints on the solution with respect to the stochastic variable. On the other hand, Neumann expansion methods are based on the idea that if an operator is a "small enough" perturbation of the identity operator, then the operator can be inverted in a Neumann series. This method places strict requirements on the boundedness of the stochastic operator. Ghanem and Spanos present an overview of these methods in their book *Stochastic Finite Elements: A Spectral Approach* [13].

Although the perturbation and Neumann expansion methods are strong theoretical tools, numerical implementation of such techniques is very cumbersome. This led many researchers to investigate finite dimensional representation of stochastic processes via chaos and KL expansions. Ghanem and Spanos [13], Xiu and Karniadakis [43], and Xiu, Karniadakis, Su, Schwab, Lucor, and Todor [19] developed solution techniques for SPDEs by substituting truncated chaos expansions of the solution into the SPDE and solving for the coefficients of the expansion. In order to solve for the coefficients, one must solve a system of deterministic PDEs using standard finite element or finite differences methods. Babuska, Tempone, and Zouraris later generalized this "spectral" finite element scheme to allow for other representations of the random processes (for instance, piecewise polynomial functions) [3, 4]. Under the finite dimensional noise assumption, these methods are extremely successful, as long as the dimension of truncation (i.e. the stochastic dimension) is small.

When the stochastic dimension becomes large, these solution techniques become computationally intractable. This led to Xiu and Hesthaven to investigate the application of collocation techniques to these SPDEs [42]. Webster, Nobile, and Tempone [39, 26] have also investigated these collocation techniques and have proven that convergence of such techniques is at least as good as the previously mentioned techniques. Furthermore, Babuska, Nobile, and Tempone [2] even proved that under certain assumptions, Xiu and Hesthaven's stochastic collocation method is a generalization of the polynomial chaos finite element method presented in [3]. One main advantage of the stochastic collocation technique is that it allows for weaker assumptions on the input data than the polynomial chaos methods described above. Moreover, due to the high dimensionality of the stochastic L^2 space (i.e. the order of truncation of the KL or chaos expansion), many authors have investigated collocation techniques based on Smolyak's algorithm for reducing the dimensionality of tensor product operators [35]. These techniques reduce the dimensionality of the resulting deterministic PDE system drastically.

Smolyak's algorithm has equally received much interest. Gerstner and Griebel show that choosing extended Gauss Patterson quadrature for the basis of the Smolyak algorithm leads to superior approximations as opposed to other quadrature rules [12]. On the other hand, Novak and Ritter show that one can construct sparse grids using a subset of the quadrature rules required in the original Smolyak algorithm [29]. This leads to even further reduced tensor product quadrature rules. Finally, Barthelmann, Ritter, Novak, Wasilkowski, Wozniakowski, and others have shown improved error bounds on the original Smolyak algorithm [5, 38]. All of these advancements in the understanding of Smolyak's work allow for quicker solution of SPDEs using the collocation technique. These collocation and sparse grid techniques are quick and efficient, and therefore, well suited for SPDE constrained optimization routines.

2.2 Optimization

With the advent of computers, much research has been devoted to numerically solving unconstrained and constrained optimization problems (see Dennis and Schnabel [8] or Nocedal and Wright [27]). Furthermore, many engineering and physical applications result in problems in the calculus of variations and optimization problems governed by PDEs. This created the necessity for the theory of optimization in general Banach spaces (a nice introduction to these topics is [16]). This thesis applies the abstract techniques in the theory of optimization in Banach spaces to optimization problems governed by SPDEs.

Another problem that may arise in application is: given a deterministic PDE constrained optimization problem, how does one solve the problem if the deterministic governing equation is replaced with a stochastic equation? The objective function composed with the solution to an SPDE is a random variable, not a real number. This sort of objective function falls in the realm of stochastic optimization. The problem with random variable objective functions is that there is no ordering on the space of random variables and thus no sense of optimality. Stochastic optimizers seek to transform the objective function into a "deterministic substitute problem". These substitute problems involve mapping the random variable objective function to a real number. Many of such substitute problems involve computing statistical moments of the objective function. Marti and others have applied these substitute problems and standard optimization techniques to various solid mechanics and design problems [24, 25]. From these results, not much can be said about how to choose the deterministic substitute problem. This task is problem dependent. Rockafellar has given some results concerning this topic, stating that a deterministic substitute problem should be coherent (which will be defined in a later chapter) [31].

2.3 Interface of SPDEs and Optimization

The separate theories of SPDEs and optimization are well developed and understood, but few have attempted the interface of the two. PDE constrained optimization problems arise in many engineering and physical applications, and stochasticity arises in these problems as a modeling choice. As such, many engineers are actively applying standard optimization techniques to problems governed by SPDEs [33, 11]. These practitioners have little concern with the theoretical aspects of the problem. In 2008, Xiu presented the idea of approximating the objective function of such problems in a truncated polynomial chaos expansion. This approximation transforms the problem into a coefficient recovery problem for a finite number of coefficients [41]. Although this work presents a fast and efficient method of dealing with these stochastic optimization problems, no theory is developed. Xiu mentions nothing of convergence rates or how well the approximation is for the objective function or its derivatives. In [1], Anitescu performs a similar approximation to constrained parametric optimization problems. After writing down the Karush-Kuhn-Tucker (KKT) nonlinear optimality system, he proves that solving the KKT system is equivalent to solving a finite dimensional constrained optimization problem. Guaranteeing existence of a solution to the KKT system requires very restrictive assumptions on the solution. The novelty of Anitescu's work is that the resulting optimization problem requires far weaker assumptions on the solution to guarantee existence.

This thesis develops the background surrounding the problem of SPDE constrained optimization and presents an adjoint equation method for computing the derivatives of objective functions. First, I supply the necessary results from the theory of SPDEs and numerical solutions of such equations. Next, I present the theory of optimization in general Banach spaces applied to problems governed by SPDEs. With this background, I present the adjoint approach for computing derivatives of the objective function. Applying three numerical SPDE solution techniques (Monte Carlo, stochastic Galerkin, and stochastic collocation finite elements), I then approximate the derivatives of the objective function via the adjoint approach. Finally, I develop and analyze the errors in computing derivatives associated with each numerical SPDE scheme and supply numerical results.

Chapter 3

Elliptic PDEs with Stochastic Inputs

This thesis develops an adjoint based approach to solving optimization problems governed by stochastic partial differential equations. In order to solve such optimization problems, it is crucial to be able to understand and solve SPDEs. SPDEs are partial differential equations for which the input data (i.e. forcing terms, diffusion/advection/reaction coefficients, domain) are uncertain. This uncertainty could arise from modeling errors or errors in experimental data. In this section, I will focus on elliptic PDEs with stochastic forcing functions and coefficients.

Suppose $D \subset \mathbb{R}^d$ is some physical domain ($d = 1, 2, 3$) and $(\Omega, \mathcal{F}, \mu)$ is a probability space. That is, Ω is the set of events, $\mathcal{F}$ is a σ-algebra of sets in Ω, and μ is a positive measure satisfying $\mu(\Omega) = 1$. We wish to find a function $u : \Omega \times D \to \mathbb{R}$ that solves the stochastic elliptic partial differential equation almost surely:

$$\begin{aligned} -\nabla \cdot (a(\omega, x)\nabla u(\omega, x)) &= f(\omega, x) && \text{in } D \\ u(\omega, x) &= 0 && \text{on } \partial D. \end{aligned} \tag{3.0.1}$$

In this equation, f and a are functions mapping the product space $\Omega \times D$ to $\mathbb{R}$.

The solution to the SPDE, as well as the inputs a and f, are random fields.

There are two ways to view a random field. One can view a random field as a family of random variables indexed by the spatial variable. Alternatively, one can view a random field as a family of realizations indexed by a stochastic variable. The latter view is used in this thesis.

3.1 The Weak Formulation

The solution u of (3.0.1) is a random field. For fixed ω, (3.0.1) is an elliptic equation and, under suitable conditions has a weak solution $u(\omega,\cdot) \in H_0^1(D)$. This generates a map

$$\Omega \ni \omega \mapsto u(\omega,\cdot) \in H_0^1(D).$$

Such functions that map measure spaces into Banach spaces motivate the development of Bochner spaces.

Definition 3.1.1 *Let X be a Banach space and let $(\Omega, \mathcal{F}, \mu)$ be a measure space. For $1 \leq p < \infty$, the linear space $L^p_\mu(\Omega; X)$ is defined as the space of Bochner integrable functions $u : \Omega \to X$ such that*

$$\int_\Omega \|u(\omega)\|_X^p \mathrm{d}\mu(\omega) < \infty.$$

If $p = \infty$, then $L^\infty_\mu(\Omega; X)$ is the space of Bochner integrable functions $u : \Omega \to X$ such that

$$\operatorname*{ess\,sup}_{\omega\in\Omega} \|u(\omega)\|_X < \infty.$$

Many desirable properties of Lebesgue L^p spaces follow through to Bochner L^p_μ spaces. For a more complete study see the appendix. Here, I will present only the most useful results for the discussion of SPDEs. Since the Sobolev space $W^{s,q}(D) \subset L^q(D)$, I will first present results concerning the case when $X = L^q(D)$.

Theorem 3.1.2 *The space $L^p(\Omega; L^q(D))$ for $p, q \in [1, \infty)$ is isomorphic to*

$$V = \left\{ v : \Omega \times D \to \mathbb{R}^d \; : \; \int_\Omega \left(\int_D |v(\omega, x)|^q \mathrm{d}x \right)^{p/q} \mathrm{d}\mu(\omega) < \infty \right\}.$$

In the case of SPDEs, this isomorphism allows for the characterization of L^p mappings from Ω to $L^p(D)$ as L^p mappings on the product space $\Omega \times D$. Elements on the product space are inherently easier to deal because they do not require the theory of Bochner measurability and integrability. This isomorphim does not generally hold when $p = q = \infty$ though.

Theorem 3.1.3 *$L^\infty(\Omega; L^\infty(D)) \subset L^\infty(\Omega \times D)$, but, in general, $L^\infty(\Omega; L^\infty(D)) \neq L^\infty(\Omega \times D)$.*

Stochastic Sobolev spaces are Bochner spaces for which $X = W^{s,q}(D)$. These spaces are defined as

$$\begin{aligned} &L^q_\mu(\Omega; W^{s,q}(D)) \\ &= \left\{ v : \Omega \longrightarrow W^{s,q}(D) \ : \ v \text{ Bochner measurable, } \int_\Omega \|v(\omega, \cdot)\|^q_{W^{s,q}(D)} \mathrm{d}\mu(\omega) < \infty \right\} \end{aligned}$$

and will be the solution spaces for SPDEs. Similar to the case when $X = L^q(D)$, stochastic Sobolev spaces exhibit the following isomorphic relation.

Theorem 3.1.4 *The space $L^p(\Omega; W^{s,q}(D))$ for $p, q \in [1, \infty)$ is isomorphic to the space*

$$V = \left\{ v : \Omega \times D \longrightarrow \mathbb{R}^d \ : \ \int_\Omega \left(\|v(\omega, \cdot)\|_{W^{s,q}(D)} \right)^p \mathrm{d}\mu(\omega) < \infty \right\}.$$

This result allows for the approximation of solutions to SPDEs as the tensor product of functions in $L^p_\mu(\Omega)$ and functions in $W^{s,q}(D)$.

Now, to pose the weak formulation of the linear elliptic SPDE, consider solutions in the Bochner space $V = L^2_\mu(\Omega; H^1(D))$, which is a Hilbert space with inner product:

$$\langle u, v \rangle_V = \int_\Omega \langle u(\omega, x), v(\omega, x) \rangle_{H^1(D)} \mathrm{d}\mu(\omega).$$

To satisfy the homogeneous Dirichlet boundary conditions consider the test space $V_0 = L^2_\mu(\Omega; H^1_0(D)) \subset V$. The weak problem is: find $u \in V_0$ such that

$$\int_\Omega \int_D a(\omega, x) \nabla u(\omega, x) \cdot \nabla v(\omega, x) \mathrm{d}x \mathrm{d}\mu(\omega) = \int_\Omega \int_D f(\omega, x) v(\omega, x) \mathrm{d}x \mathrm{d}\mu(\omega) \qquad (3.1.1)$$

holds for all $v \in V_0$. Here dx denotes Lebesgue integration. To derive the variational formulation, assume $u \in L^2_\mu(\Omega; H^2(D))$ and multiply (3.0.1) by a test function $v \in V_0$. Integrating this product and applying Green's identity yields

$$\begin{aligned}
&-\int_\Omega \int_D \nabla \cdot (a(\omega, x)\nabla u(\omega, x))v(\omega, x)\mathrm{d}x\mathrm{d}\mu(\omega) \\
&= \int_\Omega \int_D a(\omega, x)\nabla u(\omega, x) \cdot \nabla v(\omega, x) - \int_\Omega \int_{\partial D} \frac{\partial u}{\partial n}(\omega, x)v(\omega, x)\mathrm{d}x\mathrm{d}\mu(\omega).
\end{aligned}$$

Since $v \in V_0$, $v(\omega, x) = 0$ a.s. for $x \in \partial D$ the boundary integral in the previous identity is zero and (3.1.1) follows.

3.2 Existence and Uniqueness of Weak Solutions

From the weak formulation, define the bilinear form

$$\mathcal{A}(u, v) = \int_\Omega \int_D a\nabla u \cdot \nabla v \tag{3.2.1}$$

and the linear form

$$l(v) = \int_\Omega \int_D fv. \tag{3.2.2}$$

The weak form (3.1.1) can be reformulated as: find $u \in V_0$ such that

$$\mathcal{A}(u, v) = l(v)$$

for all $v \in V_0$. As with deterministic PDEs, one can establish existence and uniqueness using the Lax-Milgram theorem. To do this, I will determine, under what conditions, $\mathcal{A}$ is continuous and coercive, and l is continuous. First, recall that $L^\infty_\mu(\Omega; L^\infty(D)) \subset L^\infty(\Omega \times D)$. This result will help to prove continuity and coercivity of a.

Theorem 3.2.1 *Let $a \in L^\infty(\Omega \times D)$ and suppose there exists $\alpha_0 > 0$ such that $a \geq \alpha_0$ almost everywhere in $\Omega \times D$. Then $\mathcal{A}$ defined in (3.2.1) is continuous and coercive.*

Proof: First, notice that by theorems A.3.5 and A.3.7, $V \subset L^2(\Omega; L^2(D)) \cong L^2(\Omega \times D)$. Thus, by Cauchy-Schwarz and the fact that $a \in L^\infty(\Omega \times D)$, for all

$v, u \in V$,

$$\int_{\Omega\times D} |a(\omega,x)\nabla u(\omega,x)\cdot\nabla v(\omega,x)| \le \|a\|_{L^\infty(\Omega\times D)}\|\nabla u\|_{L^2(\Omega;L^2(D))}\|\nabla v\|_{L^2(\Omega;L^2(D))}.$$

Therefore, $a\nabla u\cdot\nabla v \in L^1(\Omega\times D)$ and the assumptions of Fubini's theorem are satisfied. Fubini's theorem may be applied to compute the iterated integrals

$$\begin{aligned}|\mathcal{A}(u,v)| &= \left|\int_\Omega\int_D a(\omega,x)\nabla u(\omega,x)\cdot\nabla v(\omega,x)\mathrm{d}x\mathrm{d}\mu(\omega)\right|\\ &= \left|\int_{\Omega\times D} a(\omega,x)\nabla u(\omega,x)\cdot\nabla v(\omega,x)\right|.\end{aligned}$$

Now, by repeated use of Cauchy-Schwarz and the fact that $\|w\|_{H^1(D)} \ge \|\nabla w\|_{L^2(D)}$ for all $w \in H^1(D)$,

$$\begin{aligned}|\mathcal{A}(u,v)| &\le \int_{\Omega\times D} |a(\omega,x)\nabla u(\omega,x)\cdot\nabla v(\omega,x)|\\ &\le \|a\|_{L^\infty(\Omega\times D)}\int_{\Omega\times D}|\nabla u(\omega,x)\cdot\nabla v(\omega,x)|\\ &\le \|a\|_{L^\infty(\Omega\times D)}\|u\|_V\|v\|_V.\end{aligned}$$

Thus, $\mathcal{A}$ is a continuous bilinear form.

To see coercivity, for all $v \in V$, Fubini's theorem implies

$$\begin{aligned}\mathcal{A}(v,v) &= \int_\Omega\int_D a(\omega,x)\nabla v(\omega,x)\cdot\nabla v(\omega,x)\mathrm{d}x\mathrm{d}\mu(\omega)\\ &= \int_{\Omega\times D} a(\omega,x)\nabla v(\omega,x)\cdot\nabla v(\omega,x).\end{aligned}$$

Since a is bounded from below by α_0 almost surely,

$$\begin{aligned}\mathcal{A}(v,v) &\ge \alpha_0\int_{\Omega\times D}\nabla v(\omega,x)\cdot\nabla v(\omega,x)\\ &= \alpha_0\int_\Omega\|\nabla v(\omega,\cdot)\|^2_{L^2(D)}\mathrm{d}\mu(\omega).\end{aligned}$$

Finally, by applying Poincare's inequality,

$$\mathcal{A}(v,v) \ge \frac{\alpha_0}{C_P^2}\int_\Omega\|v\|^2_{H^1(D)} = \frac{\alpha_0}{C_P^2}\|v\|^2_V$$

where C_P denotes the constant from Poincare's inequality. $\square$

To prove continuity of the linear functional l, first notice that one can write l as the integral of a duality product between a mapping, $f : \omega \longrightarrow (L^2(D))^*$ with a test function $v \in V_0$. That is,

$$l(v) = \int_\Omega \langle f(\omega), v(\omega, \cdot)\rangle_{(H^1(D))^*, H^1(D))} \mathrm{d}\mu(\omega).$$

From this notation, the sufficient condition on f for l to be continuous becomes clear.

Theorem 3.2.2 *If $f \in L^2_\mu(\Omega; (H^1(D))^*)$, then the linear functional defined in (3.2.2) is continuous.*

Proof: Suppose $f \in L^2_\mu(\Omega; (H^1(D))^*)$, then for all $v \in V$, Hölder's inequality from theorem A.3.5 ensures that

$$\begin{aligned} |l(v)| &= \left| \int_\Omega \langle f(\omega), v(\omega, \cdot)\rangle_{(H^1(D))^*, H^1(D))} \mathrm{d}\mu(\omega) \right| \\ &\leq \|f\|_{H^1_\mu(\Omega;(H^1(D))^*)} \|v\|_{L^2_\mu(\Omega;H^1(D))}. \end{aligned}$$

Thus, l is a continuous linear functional. $\square$

Under the conditions of Theorems 3.2.1 and 3.2.2, the Lax-Milgram theorem ensures the existence and uniqueness of the solution to the weak formulation.

Theorem 3.2.3 *Let $a \in L^\infty(\Omega \times D)$ be a function such that there exists a constant α_0 with $a \geq \alpha_0$ almost everywhere on $\Omega \times D$, and let $f \in L^2_\mu(\Omega; (H^1(D))^*)$. Furthermore, let $\mathcal{A}$ and l be defined by (3.2.1) and (3.2.2), respectively. Then the variational problem: find $u \in V$ such that*

$$\mathcal{A}(u, v) = l(v) \qquad \textit{for all } v \in V,$$

has a unique solution.

Proof: Apply the Lax-Milgram theorem to $\mathcal{A}$ and l use Theorems 3.2.1 and 3.2.2. $\square$

3.3 "Equivalence" between Strong and Weak Formulations

In this section, I will show that under certain conditions, the strong and weak formulations are equivalent. First, suppose $u : \Omega \times \bar{D} \to \mathbb{R}$ solves (3.0.1). Such a u must be twice differentiable and must satisfy the boundary conditions. That is, $u \in L^2_\mu(\Omega; H^1_0(D) \cap C^2(D))$. Also, the strong formulation (3.0.1) requires that the random field $a(\omega, \cdot) \in C^1(D)$ almost surely. As seen in the weak formulation section, these conditions ensure that u is also solves the variational problem (3.1.1). Now, suppose $u \in L^2_\mu(\Omega; H^1_0(D))$ is a solution to the weak problem (3.1.1). Since the strong problem requires second derivatives, the weak solution must have at least second order derivatives and must satisfy the homogeneous Dirichlet boundary conditions for almost every $\omega \in \Omega$ (i.e $u(\omega, \cdot) \in H^1_0(D) \cap C^2(D)$ almost surely). Under these assumptions, one can show that u is also a solution to the strong problem (3.0.1).

Theorem 3.3.1 *Suppose $u \in L^2_\mu(\Omega; H^1_0(D) \cap C^2(D))$ solves (3.1.1). Then u solves (3.0.1).*

Proof: Since u solves (3.1.1), it satisfies: for all $v \in V_0$,

$$\int_\Omega \int_D a(\omega, x)\nabla u(\omega, x) \cdot \nabla v(\omega, x) \mathrm{d}x \mathrm{d}\mu(\omega) = \int_\Omega \int_D f(\omega, x) v(\omega, x) \mathrm{d}x \mathrm{d}\mu(\omega).$$

Applying Green's identity to the left hand side yields

$$\int_\Omega \int_D a(\omega, x)\nabla u(\omega, x) \cdot \nabla v(\omega, x) \mathrm{d}x \mathrm{d}\mu(\omega) =$$
$$-\int_\Omega \int_D \nabla \cdot (a(\omega, x)\nabla u(\omega, x)) v(\omega, x) \mathrm{d}x \mathrm{d}\mu(\omega) + \int_\Omega \int_{\partial D} \frac{\partial u}{\partial n}(\omega, x) v(\omega, x) \mathrm{d}x \mathrm{d}\mu(\omega).$$

Since this holds for all $v \in V_0$, the boundary term is zero. Therefore, u satisfies

$$\int_\Omega \int_D (-\nabla \cdot (a(\omega, x)\nabla u(\omega, x)) - f(\omega, x)) v(\omega, x) \mathrm{d}x \mathrm{d}\mu(\omega) = 0$$

for all $v \in V_0$. Now, consider the space $L^2_\mu(\Omega; C^\infty_0(D))$ (i.e. $\phi(\omega, \cdot)$ is almost surely infinitely differentiable and has compact support). Since $C^\infty_0(D)$ is dense in $L^2(D)$,

$L^2_\mu(\Omega; C_0^\infty(D))$ is dense in $L^2_\mu(\Omega; L^2(D))$. Thus, for any function $\phi \in L^2_\mu(\Omega; L^2(D))$, there exists a sequence of $L^2_\mu(\Omega; C_0^\infty(D))$ functions that approximate ϕ arbitrarily closely. Take $\phi = (-\nabla \cdot (a\nabla u) - f) \in L^2_\mu(\Omega; L^2(D))$. Then there exists $\{\phi_n\}_{n=1}^\infty \subset L^2_\mu(\Omega; C_0^\infty(D))$ such that $\phi_n \to \phi$ as $n \to \infty$. This yields

$$0 = \int_\Omega \int_D (-\nabla \cdot (a(\omega,x)\nabla u(\omega,x)) - f(\omega,x))\phi_n(\omega,x)\mathrm{d}x\mathrm{d}\mu(\omega) \to$$
$$\int_\Omega \int_D (-\nabla \cdot (a(\omega,x)\nabla u(\omega,x)) - f(\omega,x))^2\mathrm{d}x\mathrm{d}\mu(\omega) \geq 0.$$

Since $(-\nabla(a\nabla u) - f))^2 \geq 0$, this implies $(-\nabla(a\nabla u) - f) = 0$ almost surely. Hence u solves the strong problem. u also solves the boundary conditions trivially since $u \in L^2_\mu(\Omega; H_0^1(D) \cap C^2(D))$. □

The previous theorem shows that under certain regularity assumptions, a weak solution u is also a strong solution. Similarly, a strong solution u is also a weak solution.

3.4 Finite Dimensional Noise

The assumption of finite dimensional noise will be absolutely crucial when applying numerical methods to solving these SPDEs. This assumption requires that the random fields a and f depend only on a finite number of random variables. Symbolically, one writes $a(\omega, x) = a(Y_1(\omega), ..., Y_N(\omega), x)$ and $f(\omega, x) = f(Y_1(\omega), ..., Y_N(\omega), x)$ where $Y = (Y_1, \dots, Y_N)$ is a random vector. For the purposes of this work, Y_i is assumed to have mean zero and unit variance for all $i = 1, \dots, N$. As mentioned, the finite dimensional noise assumption is crucial in the application of numerical methods to these SPDEs. Some methods allow for nonlinear dependence on the random vector Y while others do not. If a and f have linear dependence on Y, then they may be expanded as

$$a(\omega, x) = E[a](x) + \sum_{k=1}^N a_k(x)Y_k(\omega)$$

and

$$f(\omega, x) = E[f](x) + \sum_{k=1}^{N} f_k(x) Y_k(\omega)$$

where the functions a_k and f_k are assumed to be uniformly bounded real functions. Expansions of this type are discussed in the appendix (see for example, the Karhunen-Loeve (KL) expansion). For integration purposes, this work assumes that the image of each Y_k is a bounded interval in $\mathbb{R}$, $\Gamma_k = (a_k, b_k)$ with $a_k < b_k$. Thus, the range of Y is the product space $Y(\Omega) = \Gamma = \Gamma_1 \times \ldots \times \Gamma_N$, which is an N-dimensional rectangle.

The next theorem (known as the Doob-Dynkin's lemma [17]), allows for a rather nice classification of the solution of an SPDE when the coefficient functions a and f satisfy the finite noise assumption.

Theorem 3.4.1 *Suppose $(\Omega, \mathcal{F})$ and $(\Lambda, \mathcal{N})$ are measurable spaces and $\phi : \Omega \longrightarrow \Lambda$ is a measurable function. Then a function $\psi : \Omega \longrightarrow \mathbb{R}$ is $\phi^{-1}(\mathcal{F})$-measurable ($\phi^{-1}(\mathcal{F})$ is the σ-algebra generated by the set $\{\phi^{-1}(A) : A \in \mathcal{N}\}$) if and only if there exists a measurable function $: \Lambda \longrightarrow \mathbb{R}$ such that $\psi = h \circ \phi$.*

Applying this result to the case of SPDEs, one can show that the solution to an SPDE satisfies the finite dimensional noise assumption if the coefficient functions a and f do. Furthermore, the solution depends on the same random vector Y as a and f.

Also, under the finite dimensional noise assumption the task of integrating over some probability space becomes integrating over an N-dimensional rectangle. Returning to the weak formulation (3.1.1), notice that

$$\begin{aligned} \mathcal{A}(u, v) &= \int_\Omega \int_D a(\omega, x) \nabla u(\omega, x) \cdot \nabla v(\omega, x) \mathrm{d}x \mathrm{d}\mu(\omega) \\ &= \int_\Gamma \int_D a(y, x) \nabla u(y, x) \cdot \nabla v(y, x) \mathrm{d}x \mathrm{d}(\mu \circ Y^{-1})(y) \end{aligned}$$

Therefore, if $\mu \circ Y^{-1}$ is absolutely continuous with respect to the N-dimensional Lebesgue measure, the Radon-Nikodym theorem [10] ensures that there exists an almost everywhere unique function $\rho : \Gamma \to \mathbb{R}$ such that

$$\mathcal{A}(u,v) = \int_\Gamma \rho(y) \int_D a(y,x)\nabla u(y,x) \cdot \nabla v(y,x)\mathrm{d}x\mathrm{d}y.$$

The function ρ is called the Lebesgue density of the random vector Y. In this case, the weak formulation (3.1.1) becomes the following parametrized deterministic PDE: find $u \in L^2_\rho(\Gamma) \otimes H^1_0(D)$, such that for all $v \in L^2_\rho(\Gamma) \otimes H^1_0(D)$ the following holds

$$\int_\Gamma \rho(y) \int_D a(y,x)\nabla u(y,x) \cdot v(y,x)\mathrm{d}x\mathrm{d}y = \int_\Gamma \rho(y) \int_D f(y,x)v(y,x)\mathrm{d}x\mathrm{d}y. \tag{3.4.1}$$

It should be noted that ∇ here refers to the gradient with respect to x.

The computation of the KL expansion is discussed, e.g., in [18, 22, 34] and in the appendix. For the computation of the density function from μ and Y see, e.g., [17].

In this thesis, I assume that the random fields depend on the random vector and focus on the numerical solution of (3.4.1) as well as on optimization problems governed by equations like (3.4.1) . The weak problem (3.4.1) is the variational formulation for the following parametrized elliptic PDE:

$$\begin{aligned} -\nabla \cdot (a(y,x)\nabla u(y,x)) &= f(y,x) && \text{in } \Gamma \times D \\ u(y,x) &= 0 && \text{on } \Gamma \times \partial D. \end{aligned}$$

Although the assumption of finite noise transforms the stochastic problem into a parametrized family of deterministic PDEs, the parameter space Γ is large, which makes the numerical solution of (3.4.1) challenging.

Chapter 4

Numerical Solution of Linear Elliptic SPDEs

Introduction

In this section we will review three popular approaches for the numerical solution of SPDEs: the Monte Carlo finite element method, the stochastic collocation finite element method, and the stochastic Galerkin method. The numerical solution of SPDEs is an important ingredient in the numerical solution of optimization problems governed by SPDEs. The structure of the discretization schemes and their approximation properties strongly influences the efficiency with which the optimization problems can be solved.

Under the finite dimensional noise assumption, solving stochastic elliptic PDEs reduces to solving the following parametrized elliptic PDE

$$\begin{aligned} -\nabla \cdot (a(y,x)\nabla u(y,x)) &= f(y,x) \qquad \text{for } (y,x) \in \Gamma \times D \\ u(y,x) &= 0 \qquad \text{for } (y,x) \in \Gamma \times \partial D. \end{aligned} \tag{4.0.1}$$

4.1 Monte Carlo Finite Element

In general, Monte Carlo methods are sampling based techniques for computing statistical quantities. In the context of stochastic PDEs, one typically samples the parameter vector $y \in \Gamma$ and computes realizations of the parametric PDE. Markov's inequality, shows that the Monte Carlo method "converges" like $1/\sqrt{P}$ where P denotes the sample size. For this reason, Monte Carlo methods require a large sample size to determine "good" approximations of the solution.

Given a sample $y^k = (y_1^k, ..., y_M^k) \in \Gamma$ of the random variable y the resulting PDE to compute the corresponding solution $u(y^k, x)$ is

$$\begin{aligned} -\nabla \cdot (a(y^k, x)\nabla u(y^k, x)) &= f(y^k, x) \qquad \text{for } x \in D \\ u(y^k, x) &= 0 \qquad\qquad \text{for } x \in \partial D. \end{aligned} \tag{4.1.1}$$

To solve this PDE, any reasonable numerical PDE technique should be suitable. I will describe the finite element approach. The finite element formulation requires solutions of the weak formulation of (4.1.1) to have first order weak derivatives satisfying the zero Dirichlet boundary conditions, thus I choose the the weak solution space $V = H_0^1(D)$. For any finite dimensional subspace $V_h \subset V$ with basis $V_h = \text{span}\{\phi_1, ..., \phi_N\}$, one can compute the finite element solution by solving the matrix equation:

$$K(y^k)\vec{u}_k = F(y^k)$$

where

$$(K(y^k))_{i,j} = \int_D a(y^k, x)\nabla\phi_i(x) \cdot \nabla\phi_j(x)\mathrm{d}x \tag{4.1.2}$$

and

$$(F(y^k))_j = \int_D f(y^k, x)\phi_j(x)\mathrm{d}x. \tag{4.1.3}$$

This gives rise to the Monte Carlo FEM algorithm:

Algorithm 4.1.1 (Monte Carlo FEM)

Given P, a, f, D, Γ, and ϕ_ℓ for $\ell = 1, ..., N$

1. *Draw P samples $\{y^k\}_{k=1}^P$ of y from Γ;*
2. *For $k = 1 : P$*
   (a) *Compute $K(y^k)$ and $F(y^k)$ from equations (4.1.2) and (4.1.3);*
   (b) *Solve $K(y^k)\vec{u}^k = F(y^k)$;*
3. *Compute desired statistics.*

To compute the m^{th} moment of the solution $u(y,x)$, approximate:

$$E[u(\cdot,x)^m] \approx \frac{1}{P}\sum_{k=1}^{P} u(y^k,x)^m.$$

If a and f act linearly with respect to y, this formulation simplifies. That is, suppose a and f have the following expansions:

$$a(y,x) = a_0(x) + \sum_{i=1}^{M} y_i a_i(x)$$

and

$$f(y,x) = f_0(x) + \sum_{i=1}^{M} y_i f_i(x).$$

Then write the elements of $K(y^k)$ as

$$(K(y^k))_{\ell,j} = \int_D a_0(x)\nabla\phi_\ell(x)\cdot\nabla\phi_j(x)\mathrm{d}x + \sum_{i=1}^{M} y_i^k \int_D a_i(x)\nabla\phi_\ell(x)\cdot\nabla\phi_j(x)\mathrm{d}x$$

and the components of $F(y^k)$ as

$$(F(y^k))_j = \int_D f_0(x)\phi_j(x)\mathrm{d}x + \sum_{i=1}^{M} y_i^k \int_D f_i(x)\phi_j(x)\mathrm{d}x.$$

This allows for fast computation. We can precompute the matrices and vectors:

$$(K_i)_{\ell,j} = \int_D a_i(x)\nabla\phi_\ell(x)\cdot\nabla\phi_j(x)\mathrm{d}x \tag{4.1.4}$$

and

$$(F_i)_j = \int_D f_i(x)\phi_j(x)\mathrm{d}x \tag{4.1.5}$$

for $i = 0, ..., M$, then the construction of the stiffness matrix and the load vector becomes

$$K(y^k) = K_0 + \sum_{i=1}^{M} y_i^k K_i$$

and

$$F(y^k) = F_0 + \sum_{i=1}^{M} y_i^k F_i.$$

This leads to the following algorithm:

Algorithm 4.1.2 (Monte Carlo FEM with a and f in KL Expansion)

Given P, a, f, D, Γ, and ϕ_ℓ for $\ell = 1, ..., N$

1. *Compute K_i and F_i for $i = 0, ..., M$ from equations (4.1.4) and (4.1.5);*
2. *Draw P samples of y from Γ*
3. *For $k = 1 : P$, solve:*

$$(K_0 + \sum_{i=1}^{M} y_i^k K_i)\vec{u}^k = F_0 + \sum_{i=1}^{M} y_i^k F_i$$

4. *Compute desired statistics.*

Notice that the resulting FEM systems take the form the following block diagonal system:

$$\underbrace{\begin{pmatrix} K_0 + \sum_{i=1}^{M} y_i^1 K_i & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & K_0 + \sum_{i=1}^{M} y_i^P K_i \end{pmatrix}}_{=K} \underbrace{\begin{pmatrix} \vec{u}_1 \\ \vdots \\ \vec{u}_P \end{pmatrix}}_{=\vec{u}} = \underbrace{\begin{pmatrix} \vec{F}_1 \\ \vdots \\ \vec{F}_P \end{pmatrix}}_{=F}.$$

Let $I \in \mathbb{R}^{P\times P}$ be the identity matrix, then one can rewrite this system in Kronecker product notation as:

$$(I \otimes K_0 + \sum_{i=1}^{M} \operatorname{diag}(y_i^1, \dots, y_i^P) \otimes K_i)\vec{u} = F.$$

Although this is a compact manner of writing the Monte Carlo FEM system, it is not the best method for implementation. Since this system is block diagonal, the implementation can easily be parallelized. One advantage of this compact notation is that each of the following methods for numerically solving SPDEs results in a similar tensor product formulation.

4.2 Collocation Methods

The idea behind collocation techniques is to approximate the solution to an equation (be it differential, integral, or algebraic) in a finite dimensional subspace of the solution space and require the approximation to satisfy the original equation at finitely many predetermined points. This finite dimensional subspace is called the *collocation space* and finite set of points the *collocation points*. For the purposes of this thesis, I restrict the collocation space to the space of degree $P-1$ polynomials, $\mathbb{P}^{P-1}(\Gamma)$.

Consider the parametrized PDE

$$\begin{aligned} -\nabla \cdot (a(y,x)\nabla u(y,x)) &= f(y,x) \qquad \text{for } (y,x) \in \Gamma \times D \\ u(y,x) &= 0 \qquad\qquad \text{for } (y,x) \in \Gamma \times \partial D \end{aligned}$$

and collocation space $\mathbb{P}^{P-1}(\Gamma) \subset L^2_\rho(\Gamma)$. If $\{y^j\}_{j=1}^P \subset \Gamma$ are collocation points, then the set of Lagrange basis functions built on the collocation points, $L_j(y^i) = \delta_{ij}$, form a basis of $\mathbb{P}^{P-1}(\Gamma)$ and, due to the nodal nature of the Lagrange polynomials, we can require the approximate solution to the SPDE to be exact at the collocation points. With this collocation framework, the approximate solution is merely the Lagrange interpolant:

$$u_P(y,x) = \sum_{k=1}^{P} u(y^k,x)L_k(y)$$

where $u(y^k,x)$, $k=1,\ldots,P$, are the solutions to:

$$\begin{aligned} -\nabla \cdot a(y^k,x)\nabla u(y^k,x) &= f(y^k,x) \text{ in } D \\ u(y^k,x) &= 0 \text{ on } \partial D. \end{aligned}$$

As in Monte Carlo, this thesis uses finite elements to compute the P solutions to the above system of PDEs. In the FEM framework, we have to construct P stiffness matrices $K(y^k)$ and P load vectors $F(y^k)$ where

$$(K(y^k))_{\ell,j} = \int_D a(y^k, x)\nabla\phi_\ell(x) \cdot \nabla\phi_j(x)\mathrm{d}x \tag{4.2.1}$$

and

$$(F(y^k))_j = \int_D f(y^k, x)\phi_j(x)\mathrm{d}x. \tag{4.2.2}$$

The stochastic collocation finite element algorithm is as follows:

Algorithm 4.2.1 (Stochastic Collocation FEM)

```
*Given $P$, $a$, $f$, $D$, $\Gamma$, $\{y^k\}_{k=1}^P$ and $\phi_\ell$ for $\ell = 1, ..., N$*
  1. For $k = 1 : P$
      (a) Compute $K(y^k)$ and $F(y^k)$ from equations (4.2.1) and (4.2.2);
      (b) Solve $K(y^k)\vec{u}^k = F(y^k)$;
  2. Compute desired statistics.
```

The approximate solution of the Stochastic PDE using stochastic collocation and finite elements is given by

$$u_{hP}(y, x) = \sum_{k=1}^{P} \underbrace{\sum_{\ell=1}^{N} u_\ell^k \phi_\ell(x)}_{\equiv u_h(y^k, x)} L_k(y)$$

where u_ℓ^k is the ℓ^{th} component of $\vec{u}^k$.

If a and f have the expansions:

$$a(y, x) = a_0(x) + \sum_{i=1}^{M} y_i a_i(x)$$

and

$$f(y, x) = f_0(x) + \sum_{i=1}^{M} y_i f_i(x)$$

then the elements of $K(y^k)$ are

$$(K(y^k))_{\ell,j} = \int_D a_0(x)\nabla\phi_\ell(x)\cdot\nabla\phi_j(x)\mathrm{d}x + \sum_{i=1}^{M} y_i^k \int_D a_k(x)\nabla\phi_\ell(x)\cdot\nabla\phi_j(x)\mathrm{d}x$$

and the components of $F(y^k)$ as

$$(F(y^k))_j = \int_D f_0(x)\phi_j(x)\mathrm{d}x + \sum_{i=1}^{M} y_i^k \int_D f_k(x)\phi_j(x)\mathrm{d}x.$$

Thus, by defining the matrices and vectors:

$$(K_i)_{\ell,j} = \int_D a_i(x)\nabla\phi_\ell(x)\cdot\nabla\phi_j(x)\mathrm{d}x \tag{4.2.3}$$

and

$$(F_i)_j = \int_D f_i(x)\phi_j(x)\mathrm{d}x \tag{4.2.4}$$

for $i = 0, ..., M$, then the stiffness matrix and the load vector are

$$K(y^k) = K_0 + \sum_{i=1}^{M} y_i^k K_i$$

and

$$F(y^k) = F_0 + \sum_{i=1}^{M} y_i^k F_i.$$

This leads to the following algorithm:

Algorithm 4.2.2 (Stochastic Collocation FEM with a and f in KL Expansion)

Given P, a, f, D, Γ, $\{y^k\}_{k=1}^P$ and $\{\phi_\ell\}_{\ell=1}^N$

1. *Compute K_i and F_i for $i = 0, ..., M$ from equations (4.2.3) and (4.2.4);*

2. *For $k = 1 : P$, solve:*

$$(K_0 + \sum_{i=1}^{M} y_i^k K_i)\vec{u}_k = F_0 + \sum_{i=1}^{M} y_i^k F_i;$$

3. *Compute desired statistics.*

Notice that the resulting FEM systems take the form the following block diagonal system:

$$\underbrace{\begin{pmatrix} K_0+\sum_{i=1}^M y_i^1 K_i & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & K_0+\sum_{k=1}^M y_i^P K_i \end{pmatrix}}_{=K} \underbrace{\begin{pmatrix} \vec{u}_1 \\ \vdots \\ \vec{u}_P \end{pmatrix}}_{=\vec{u}} = \underbrace{\begin{pmatrix} \vec{F}_1 \\ \vdots \\ \vec{F}_P \end{pmatrix}}_{=F}.$$

Let $I \in \mathbb{R}^{P\times P}$ be the identity matrix, then one can cleverly rewrite this system in Kronecker product notation as:

$$(I \otimes K_0 + \sum_{i=1}^M \mathrm{diag}(y_i^1, \ldots, y_i^P) \otimes K_i)\vec{u} = F.$$

Again, this is not the best formulation for implementation since this system is easily parallelized.

In order to compute the m^{th} statistical moments of the solution, one must compute

$$\begin{aligned} E[u(\cdot,x)^m] &\approx E[u_{hP}(\cdot,x)^m] \\ &= E\left[\Big(\sum_{k=1}^P L_k(\cdot)u_h(y^k,x)\Big)^m\right] \\ &= \sum_{k_1=1}^P \cdots \sum_{k_m=1}^P E[L_{k_1}(\cdot)\cdots L_{k_m}(\cdot)]u_h(y^{k_1},x)\cdots u_h(y^{k_m},x). \end{aligned}$$

Suppose the collocation points $\{y^k\}$ are chosen as the Gaussian quadrature nodes corresponding to the weight function ρ and let $\{\omega_k\}$ denote the weights, $\omega_k = \int_\Gamma \rho(y)L_k(y)\mathrm{d}y$, then the quadrature formula

$$\int_\Gamma \rho(y)g(y)\mathrm{d}y \approx Q_P g := \sum_{k=1}^P \omega_k g(y^k)$$

integrates (with respect to ρ) degree P polynomials exactly. Hence, the m^{th} statistical moment calculation becomes:

$$\begin{aligned}
E[u(\cdot,x)^m] &\approx E[u_{hP}(\cdot,x)^m] \\
&= \sum_{k_1=1}^{P} \cdots \sum_{k_m=1}^{P} E[L_{k_1}(\cdot)\cdots L_{k_m}(\cdot)] u_h(y^{k_1},x)\cdots u_h(y^{k_m},x) \\
&\approx \sum_{k_1=1}^{P} \cdots \sum_{k_m=1}^{P} \sum_{k=1}^{P} \omega_k L_{k_1}(y^k)\cdots L_{k_m}(y^k) u_h(y^{k_1},x)\cdots u_h(y^{k_m},x) \\
&= \sum_{k_1=1}^{P} \cdots \sum_{k_m=1}^{P} \sum_{k=1}^{P} \omega_k \delta_{k_1 k}\cdots \delta_{k_m k} u_h(y^{k_1},x)\cdots u_h(y^{k_m},x) \\
&= \sum_{k=1}^{P} \omega_k u_h(y^k,x)^m.
\end{aligned}$$

Thus, the computation of high order moments is merely quadrature.

4.2.1 Quadrature Rules

As can be seen from the above discussion, the amount of work required by the collocation method depends on the quadrature formula used to define the collocation points, $\{y^j\}$. Under the assumption of independence for the finite random variables $\{Y_j\}$, the weight function becomes

$$\rho(y) = \prod_{j=1}^{M} \rho_j(y_j).$$

Thus, it suffices to consider a tensor product of 1D quadrature rules. Let $\{y_i^j\}_{i=1}^{p_j}$ and $\{\omega_i^j\}_{i=1}^{p_j}$ for denote the 1D Gaussian quadrature nodes with respect to the weight functions ρ_j for $j = 1,\ldots,M$ and define the quadrature operator as:

$$Q^j g = \sum_{i=1}^{p_j} \omega_i^j g(y_i^j).$$

If we wish to approximate the integral,

$$\int_\Gamma \rho(y)g(y)\mathrm{d}y = \int_{\Gamma_1} \rho_1(y_1)\cdots \int_{\Gamma_M} \rho_M(y_M) g(y_1,\ldots,y_M)\mathrm{d}y_M \cdots \mathrm{d}y_1$$

we can approximate each 1D integral with the above 1D quadrature rules:

$$\begin{aligned}\int_\Gamma \rho(y)g(y)\mathrm{d}y &\approx \int_{\Gamma_1} \rho_1(y_1)\cdots\int_{\Gamma_{M-1}} \rho_{M-1}(y_{M-1})Q^M g(y_1,\dots,y_M)\mathrm{d}y_{M-1}\cdots\mathrm{d}y_1 \\ &\approx \ldots \approx Q^1\cdots Q^M g \\ &= (Q^1 \otimes \ldots \otimes Q^M)g.\end{aligned}$$

Here, $\otimes$ refers to tensor product of operators. That is, if $T_i : S_i \to \mathbb{R}$ for $i = 1, 2$ and $g : S_1 \times S_2 \to \mathbb{R}$, then

$$(T_1 \otimes T_2)g = T_1(T_2 g).$$

Now, let $\mathcal{Q} = Q_1 \otimes \ldots \otimes Q_M$, then $\mathcal{Q}$ defines a quadrature rule with $P = \prod_{j=1}^M p_j$ nodes and weights defined as

$$\mathcal{N} = \{y_i^1\}_{i=1}^{p_1} \times \ldots \times \{y_i^M\}_{i=1}^{p_M}$$

and

$$\mathcal{W} = \left\{\omega_{i_1}^1 \cdots \omega_{i_M}^M \ : \ i_1 \in \{1,\dots,p_1\},\dots,i_M \in \{1,\dots p_M\}\right\}.$$

This tensor product quadrature formulation can require an immense number of nodes and hence the collocation method would require the same number of FEM solves. For example, if $p_1 = \ldots = p_M = 2$ and $M = 20$, then $P \approx 10^6$. Notice that this rule would only be exact for polynomials of degree 1 in each of the 20 directions. Thus, for the collocation method to truly reach its full potential, high degree quadrature rules with only a small number of nodes is required. In this subsection, I will present three such methods: the Smolyak algorithm and two low order methods by Stroud. The Stroud rules have the advantage of requiring a very small number of nodes, but they have a fixed degree of polynomial exactness. On the other hand, the Smolyak algorithm allows for a high order of polynomial exactness.

4.2.1.1 The Smolyak Algorithm

The Smolyak algorithm [35] is a beautiful trick based on two simple analysis results. Suppose $\{a_i\}_{i=1}^\infty$ is a convergent sequence of real numbers that converges to $a \in \mathbb{R}$, then each element a_i can be written as the telescoping sum

$$a_i = \sum_{j=1}^{i} a_j - a_{j-1}$$

where $a_0 = 0$. Since a_i converges to a, the following series is convergent

$$\sum_{j=1}^{\infty} a_j - a_{j-1}.$$

The second analysis result has to do with products of sequences. Suppose $\{a_k^1\}, \ldots, \{a_k^M\}$ are convergent sequences of real numbers with limits $a^1, \ldots, a^M$, respectively. Then the product sequence $\{a_k^1 \cdot \ldots \cdot a_k^M\}$ converges to $a^1 \cdot \ldots \cdot a^M$. Writing each element in the product sequence in its telescoping sum representation yields:

$$a_k^1 \cdot \ldots \cdot a_k^M = \sum_{j_1=1}^{k} \cdots \sum_{j_M=1}^{k} (a_{j_1}^1 - a_{j_1-1}^1) \cdot \ldots \cdot (a_{j_M}^M - a_{j_M-1}^M).$$

Approximating the k^{th} element of this sequence of partial sums by maintaining only the indices that sum to less than or equal to k, i.e.

$$\mathcal{A}(k,M) = \sum_{M \le j_1 + \ldots + j_M \le k} (a_{j_1}^1 - a_{j_1-1}^1) \cdot \ldots \cdot (a_{j_M}^M - a_{j_M-1}^M)$$

gives an new sequence $\{\mathcal{A}(k,M)\}$ that also converges to $a^1 \cdot \ldots \cdot a^M$. By expanding each term in $\mathcal{A}(k,M)$, one can rewrite

$$\mathcal{A}(k,M) = \sum_{k-M+1 \le |\mathbf{j}| \le k} (-1)^{q-|\mathbf{j}|} \binom{M-1}{k-|\mathbf{j}|} a_{j_1} \cdot \ldots \cdot a_{j_M}$$

where $\mathbf{j} = (j_1, \ldots, j_M)$ and $|\mathbf{j}| = j_1 + \ldots + j_M$.

A similar result holds when $\{a_i\}$ is replaced by a sequence of operators (say Q^i from above). Thus, suppose $\{T_i\}_{i=1}^{\infty}$ is a convergent sequence of operators that converges to the operator T, then

$$T_i = \sum_{j=1}^{i} (T_j - T_{j-1}) = \sum_{j=1}^{i} \Delta_j$$

where $T_0 = 0$ and $\Delta_j = T_j - T_{j-1}$. Then, one can show that $\underbrace{T \otimes \ldots \otimes T}_{M}$ is the limit of the sequence of operators taking the form

$$T_{i_1} \otimes \ldots \otimes T_{i_M} = \sum_{j_1=1}^{p_{i_1}} \cdots \sum_{j_M=1}^{p_{i_M}} \Delta_{j_1} \cdots \Delta_{j_M}.$$

Notice that this sequence contains the tensor product operators

$$\underbrace{T_i \otimes \ldots \otimes T_i}_{M}.$$

In fact, the z^{th} order partial sum of the sequence above is an approximation of the tensor product rule with total degree z. This is, exactly Smolyak's algorithm. Now, let us define these truncations as

$$\mathcal{A}(z, M) = \sum_{|\mathbf{i}| \leq z} \Delta_{i_1} \otimes \ldots \otimes \Delta_{i_M}.$$

In Lemma 1 of Wasilkowski and Woźniakowski's 1995 paper [38] as well as other earlier sources, present the following simplification of $\mathcal{A}$:

$$\mathcal{A}(z, M) = \sum_{z-M+1 \leq |\mathbf{i}| \leq z} (-1)^{q-|\mathbf{i}|} \binom{M-1}{z-|\mathbf{i}|} T_{i_1} \otimes \ldots \otimes T_{i_M}.$$

This result is the same as the result presented for sequences of real numbers.

In the case of quadrature, the T_i operators denote successive degrees of the 1D quadrature rule. For this to really be useful for the collocation method, one needs to know the resulting quadrature nodes and weights from this form. Let $Y^i = \{y_1^i, \ldots, y_{p_i}^i\}$ denote the 1D quadrature nodes of polynomial exactness p_i, then the "sparse grid" (quadrature nodes) are

$$\mathcal{H}(z, M) = \bigcup_{z-M+1 \leq |\mathbf{i}| \leq z} Y^{i_1} \times \ldots \times Y^{i_M}.$$

The quadrature weights are computed by a similar formula, taking into account the scaling factors $(-1)^{q-|\mathbf{i}|}\binom{M-1}{z-|\mathbf{i}|}$. The number of nodes and weights for Smolyak's algorithm is given by

$$n(z, M) \leq \sum_{z-M+1 \leq |\mathbf{i}| \leq z} p_{i_1} \cdots p_{i_M}.$$

This is indeed an improvement from the tensor product formulation (see figure 4.1). For more information on how much of an improvement, see [30].

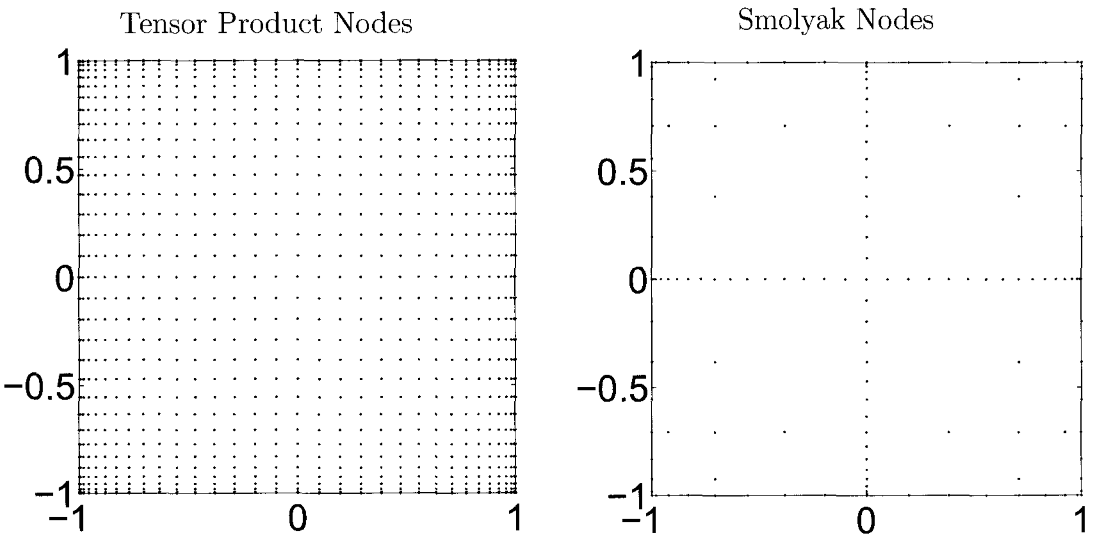


Figure 4.1: Tensor product (left) and Smolyak (right) quadrature nodes built on 1D Clenshaw-Curtis quadrature nodes. The tensor product nodes are exact for polynomials of degree 9 in each direction ($P = 1089$), while the Smolyak nodes are exact for polynomials of total degree 9 ($P = 145$).

4.2.1.2 Stroud 2- and 3-Quadrature

In his 1957 work, *Remarks on the Disposition of Points in Numerical Integration Formulas*, Stroud presents two theorems that give minimal quadrature rules of order 2 and 3 for high dimensions [36]. In this context, minimal refers to the number of nodes/weights required to achieve the desired accuracy. For degree 2 quadrature, the minimal number of nodes is $M + 1$ and for degree 3, the minimal number on nodes is $2M$. For the proofs of the following theorems, see [36]. Suppose $R \subset \mathbb{R}^M$ is a symmetric region, then Stroud proved the following necessary and sufficient condition of a quadrature rule of $M + 1$ equally weighted points to be exact for quadratics.

Theorem 4.2.3 $M+1$ *equally weighted points form a degree 2 quadrature formula in*

R if and only if these points are the vertices of an M-simplex with centroid coinciding with that of R and lie on the surface of a sphere of radius $r = \sqrt{MI_2/I_0}$ where $I_0 = \int_R \mathrm{d}v$ and $I_2 = \int_R x_1^2 \mathrm{d}v = \ldots = \int_R x_M^2 \mathrm{d}v$.

Similarly, for $R \subset \mathbb{R}^M$ symmetric, he gave the following necessary and sufficient conditions for a quadrature rule of $2M$ equally weighted points to be exact for cubics.

Theorem 4.2.4 *$2M$ equally weighted points, $\pm x_1, \ldots, \pm x_M$, form a degree 3 quadrature formula in R if and only if these points are the vertices of a Q_M region with centroid coinciding with that of R and lie on the surface of a sphere of radius $r = \sqrt{MI_2/I_0}$. Here Q_n denotes regions defined by 2^M inequalities of the form $\pm x_1 \pm \ldots \pm x_M \leq a$ for some $a \in \mathbb{R}$.*

In many applications, the random variables of interest are uniformly distributed on the M cube, $\Gamma = [-1,1]^M$. The quadrature rules for this space are as follows: for the Stroud 2-rule, the kth node $y_k \in \Gamma = [-1,1]^M$ has components

$$(y_k)_{2i-1} = \sqrt{\frac{2}{3}} \cos \frac{2ik\pi}{M+1}$$

and

$$(y_k)_{2i} = \sqrt{\frac{2}{3}} \sin \frac{2ik\pi}{M+1}$$

for $i = 1, \ldots, \lfloor \frac{M}{2} \rfloor$. If M is odd, then $(y_k)_M = (-1)^k/\sqrt{3}$. The weights for this rule are given by $w_1 = \ldots = w_{M+1} = 2^M/(M+1)$. Similarly for the Stroud 3-rule, the kth node $y_k \in \Gamma = [-1,1]^M$ has components

$$(y_k)_{2i-1} = \sqrt{\frac{2}{3}} \cos \frac{(2i-1)k\pi}{M}$$

and

$$(y_k)_{2i} = \sqrt{\frac{2}{3}} \sin \frac{(2i-1)k\pi}{M}$$

for $i = 1, \ldots, \lfloor \frac{M}{2} \rfloor$. If M is odd, then $(y_k)_M = (-1)^k/\sqrt{3}$. The weights for this rule are given by $w_1 = \ldots = w_{2M} = 2^M/(2M)$.

The nodes (figure 4.2) and weights given above form quadrature rules for integrating on the cube $\Gamma = [-1,1]^M$ with weight function $\rho(y) = 1$. As such, these formulas (with appropriate scaling) can be used when the random variables are uniformly distributed in the M-dimensional cube $[-1,1]^M$.

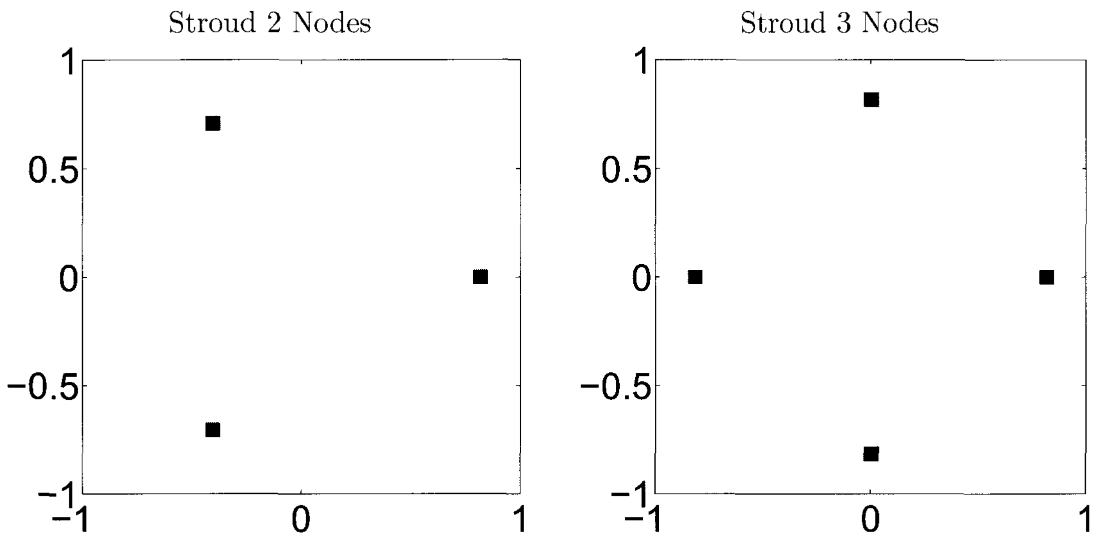


Figure 4.2: Stroud 2 (left) and Stroud 3 (right) quadrature. The Stroud 2 nodes are exact for polynomials of degree 2 in each direction ($P = 3$), while the Stroud 3 nodes are exact for polynomials of degree 3 in each direction ($P = 4$).

4.3 Stochastic Galerkin Method

As opposed to the Monte Carlo and collocation techniques, which sample in some manner Γ and require the solutions to the parametrized PDE:

$$\begin{aligned} -\nabla \cdot (a(y,x)\nabla u(y,x)) &= f(y,x) \text{ in } \Gamma \times D \\ u(y,x) &= 0 \text{ on } \Gamma \times \partial D. \end{aligned}$$

at these sample points, the stochastic Galerkin method works on the principal of Galerkin orthogonality.

Suppose $W_i \subset L^2_{\rho_i}(\Gamma_i)$ with dimension p_i for $i = 1, \ldots, M$ and $V \subset H^1_0(D)$ with dimension N. Further, let $\{\psi^i_n\}_{n=1}^{p_i}$ for $i = 1, \ldots, M$ be bases of W_i and $\{\phi_i\}_{i=1}^N$ a basis of V. One can define the finite dimensional tensor product space $W_1 \otimes \ldots \otimes W_M \otimes V$ as the space spanned by the functions $\{\psi^1_{n_1} \cdots \psi^M_{n_M} \phi_i\}$ for all $n_1 \in \{1, \ldots, p_1\}, \ldots, n_M \in \{1, \ldots, p_M\}$ and $i \in \{1, \ldots, N\}$. To simplify notation, let $\mathbf{n}$ denote a multi-index whose k^{th} component $n_k \in \{1, \ldots, p_k\}$ and $\mathcal{I}$ denote the set of such multi-indices, then the basis functions for the tensor product space have the form

$$v_{\mathbf{n}i}(y, x) = \Psi_{\mathbf{n}}(y)\phi_i(x)$$

where

$$\Psi_{\mathbf{n}} = \prod_{k=1}^{M} \psi^k_{n_k}(y_k).$$

The Galerkin method (using the inner product defined in the previous chapter) then seeks a solution in $\hat{u} \in W_1 \otimes \ldots \otimes W_M \otimes V$ such that

$$\int_\Gamma \rho(y) \int_D a(y, x) \nabla u_{hP}(y, x) \cdot \nabla v_{\mathbf{n}i}(y, x) \mathrm{d}x \mathrm{d}y = \int_\Gamma \rho(y) \int_D f(y, x) v_{\mathbf{n}i}(y, x) \mathrm{d}x \mathrm{d}y$$

for all $\mathbf{n} \in \mathcal{I}$ and $i = 1, \ldots, N$. Plugging in the explicit formula for the basis functions, we see that this equation is equivalent to

$$\int_\Gamma \rho(y)\Psi_{\mathbf{n}}(y) \int_D a(y, x) \nabla u_{hP}(y, x) \cdot \nabla \phi_i(x) \mathrm{d}x \mathrm{d}y = \int_\Gamma \rho(y)\Psi_{\mathbf{n}}(y) \int_D f(y, x)\phi_i(x) \mathrm{d}x \mathrm{d}y$$

for all $\mathbf{n} \in \mathcal{I}$ and $i = 1, \ldots, N$.

Now, writing the approximate solution as

$$u_{hP}(y, x) = \sum_{\mathbf{m} \in \mathcal{I}} \sum_{j=1}^{N} u_{\mathbf{m}j} \Psi_{\mathbf{m}}(y) \phi_j(x)$$

and substituting this into the above equation yields

$$\sum_{\mathbf{m} \in \mathcal{I}} \sum_{j=1}^{N} u_{\mathbf{m}j} \int_\Gamma \rho(y)\Psi_{\mathbf{n}}(y)\Psi_{\mathbf{m}}(y) \int_D a(y, x) \nabla \phi_j(x) \cdot \nabla \phi_i(x) \mathrm{d}x \mathrm{d}y =$$
$$\int_\Gamma \rho(y)\Psi_{\mathbf{n}}(y) \int_D f(y, x)\phi_i(x) \mathrm{d}x \mathrm{d}y$$

for all $\mathbf{n} \in \mathcal{I}$ and $i = 1, \ldots, N$. This notation can be simplified. Let $P = |\mathcal{I}| = \prod_{k=1}^{M} p_k$, then there is a natural bijection between $\{1, \ldots, P\}$ and $\mathcal{I}$. From here on, I will refer to this mapping as the global to local mapping γ. This mapping inputs an integer between 1 and P and outputs a multi-index. This allows for the more natural notation

$$\sum_{m=1}^{P} \sum_{j=1}^{N} u_{mj} \int_\Gamma \rho(y)\Psi_n(y)\Psi_m(y) \int_D a(y,x)\nabla\phi_j(x) \cdot \nabla\phi_i(x)\mathrm{d}x\mathrm{d}y = \int_\Gamma \rho(y)\Psi_n(y) \int_D f(y,x)\phi_i(x)\mathrm{d}x\mathrm{d}y.$$

From this representation of the Galerkin projection, one can define the block "stiffness" matrices K^{nm} with components

$$(K^{nm})_{ij} = \int_\Gamma \rho(y)\Psi_n(y)\Psi_m(y) \int_D a(y,x)\nabla\phi_j(x) \cdot \nabla\phi_i(x)\mathrm{d}x\mathrm{d}y$$

and the block "load" vectors as $\vec{F}^n$ with components

$$(\vec{F}^n)_i = \int_\Gamma \rho(y)\Psi_n(y) \int_D f(y,x)\phi_i(x)\mathrm{d}x\mathrm{d}y.$$

The resulting full finite element system becomes:

$$\underbrace{\begin{pmatrix} K^{11} & \cdots & K^{1P} \\ \vdots & \ddots & \vdots \\ K^{P1} & \cdots & K^{PP} \end{pmatrix}}_{=K} \underbrace{\begin{pmatrix} \vec{u}_1 \\ \vdots \\ \vec{u}_P \end{pmatrix}}_{=\vec{u}} = \underbrace{\begin{pmatrix} \vec{F}^1 \\ \vdots \\ \vec{F}^P \end{pmatrix}}_{=F}.$$

This give rise to the following algorithm:

Algorithm 4.3.1 *Given P, a, f, D, Γ, and $\{\phi_i\}_{i=1}^N$*

1.) Compute K^{nm} and $\vec{F}^n$for $n, m = 1, ..., P$

2.) Build the matrix K and vector F

3.) Solve $K\vec{u} = F$

4.) Compute desired statistics.

Notice that as M or the p_i grow larger, this method becomes very computationally expensive due to the size of K.

Now, suppose a and f have the expansion:

$$a(y,x) = a_0(x) + \sum_{i=1}^{M} y_i a_i(x)$$

and

$$f(y,x) = f_0(x) + \sum_{i=1}^{M} y_i f_i(x)$$

then one can rewrite the Galerkin projection as

$$\sum_{m=1}^{P}\sum_{j=1}^{N} u_{m,j}\left((K_0)_{i,j}\int_\Gamma \rho(y)\Psi_n(y)\Psi_m(y)\mathrm{d}y + \sum_{k=1}^{M}(K_k)_{i,j}\int_\Gamma \rho(y)\Psi_n(y)\Psi_m(y)y_k\mathrm{d}y\right)$$
$$= (F_0)_i\int_\Gamma \rho(y)\Psi_n(y)\mathrm{d}y + \sum_{k=1}^{M}(F_k)_i\int_\Gamma \rho(y)\Psi_n(y)y_k\mathrm{d}y$$

where

$$(K_k)_{i,j} = \int_D a_k(x)\nabla\phi_i(x)\cdot\nabla\phi_j(x)\mathrm{d}x$$

and

$$(F_k)_i = \int_D f_k(x)\phi_i(x)\mathrm{d}x$$

for $k = 0,\ldots,M$. The task now is to somehow decouple this system of equations. Suppose, for instance, functions Ψ_n existed such that

$$\int_\Gamma \rho(y)\Psi_n(y)\Psi_m(y)\mathrm{d}y = \delta_{nm}$$

and

$$\int_\Gamma \rho(y)\Psi_n(y)\Psi_m(y)y_k\mathrm{d}y = C_{kn}\delta_{nm}.$$

Then the prior equation becomes

$$\sum_{m=1}^{P}\sum_{j=1}^{N} u_{m,j}\left((K_0)_{i,j} + \sum_{k=1}^{M} C_{kn}(K_k)_{i,j}\right)\delta_{nm}$$
$$= (F_0)_i\int_\Gamma \rho(y)\Psi_n(y)\mathrm{d}y + \sum_{k=1}^{M}(F_k)_i\int_\Gamma \rho(y)\Psi_n(y)y_k\mathrm{d}y$$

or, equivalently,

$$\sum_{j=1}^{N} u_{m,j}\left((K_0)_{i,j} + \sum_{k=1}^{M} C_{kn}(K_k)_{i,j}\right)$$
$$= (F_0)_i \int_\Gamma \rho(y)\Psi_n(y)\mathrm{d}y + \sum_{k=1}^{M}(F_k)_i \int_\Gamma \rho(y)\Psi_n(y)y_k\mathrm{d}y.$$

This gives rise to the algorithm

Algorithm 4.3.2 *Given P, a, f, D, Γ, and $\{\phi_i\}_{i=1}^N$*

1.) Compute K_k and F_k for $k = 0, ..., M$ from equation (4.2.3) from (4.2.4)

2.) Compute the coefficients C_{ki} for $i = 1, \dots, P$ and $k = 1, \dots, M$ and the basis functions Ψ_i for $i = 1, \dots, P$

3.) Compute the vectors $\vec{F}_i$ for $i = 1, \dots, P$ 4.) For $i = 1 : P$, solve:

$$(K_0 + \sum_{k=1}^{M} C_{ki}K_k)\vec{u}_i = \vec{F}_i$$

5.) Compute desired statistics.

and the block diagonal system

$$\underbrace{\begin{pmatrix} K_0 + \sum_{k=1}^M C_{k1}K_k & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & K_0 + \sum_{k=1}^M C_{kP}K_k \end{pmatrix}}_{=K} \underbrace{\begin{pmatrix} \vec{u}_1 \\ \vdots \\ \vec{u}_P \end{pmatrix}}_{=\vec{u}} = \underbrace{\begin{pmatrix} \vec{F}_1 \\ \vdots \\ \vec{F}_P \end{pmatrix}}_{=F}.$$

As in collocation and Monte Carlo approach, let $I \in \mathbb{R}^{P\times P}$ be the identity matrix. Then one can cleverly rewrite this system in Kronecker product notation as:

$$(I \otimes K_0 + \sum_{k=1}^{M} \mathrm{diag}(C_{k1}, \dots, C_{kP}) \otimes K_k)\vec{u} = F.$$

As before, this is not the way to implement the stochastic Galerkin method. Adopting the notation

$$u_k(x) = \sum_{j=1}^{N} (\vec{u}_k)_j\ \phi_j(x),$$

one can compute the m^{th} statistical moments of the solution as

$$\begin{aligned} E[u(\cdot,x)^m] &\approx E\left[\Big(\sum_{k=1}^{P}\Psi_k(\cdot)u_k(x)\Big)^m\right] \\ &= \sum_{k_1=1}^{P}\cdots\sum_{k_m=1}^{P} E[\Psi_{k_1}(\cdot)\cdots\Psi_{k_m}(\cdot)]u_{k_1}(x)\cdots u_{k_m}(x). \end{aligned}$$

The foundation of this simplification is that one can find functions that satisfy the two previously stated orthogonality properties:

$$\int_\Gamma \rho(y)\Psi_n(y)\Psi_m(y)\mathrm{d}y = \delta_{nm}$$

and

$$\int_\Gamma \rho(y)\Psi_n(y)\Psi_m(y)y_k\mathrm{d}y = C_{kn}\delta_{nm}.$$

If W_i chosen to be continuous polynomial spaces, then we are in luck. One only needs to compute the "double" orthogonal polynomials in 1D for each W_i and, under the assumption of independence, the new tensor product polynomials are products of the 1D polynomials. These 1D polynomials may be solved for as an eigenvalue problem, but Babuška, Nobile, and Tempone present the following useful characterization in their paper [2]:

Lemma 4.3.3 *Let $\Sigma \subset \mathbb{R}$ and suppose $w : \Sigma \to \mathbb{R}$ is a function satisfying $w \geq 0$ almost everywhere. Furthermore, assume $\{\psi_k\}_{k=1}^p$ is a set of degree p polynomials satisfying*

$$\int_\Sigma w(x)\psi_i(x)\psi_j(x)\mathrm{d}x = \delta_{ij}$$

and

$$\int_\Sigma w(x)\psi_i(x)\psi_j(x)x\mathrm{d}x = C_i\delta_{ij}.$$

Then the values C_i for $i = 1,\ldots,p$ are the p^{th} order Gaussian quadrature nodes corresponding to the weight function w and the polynomials ψ_k are the Lagrange polynomials built on these nodes, up to a scalar multiple, i.e.

$$\psi_k(x) = \alpha_k \prod_{\substack{i=1\\ i\neq k}}^{p} \frac{x - C_i}{C_k - C_i}$$

for $\alpha_k \in \mathbb{R}$ *and* $k = 1, \ldots, p$.

Proof: Setting equal both orthogonality equations yields,

$$\int_\Sigma w(x)\psi_i(x)\psi_j(x)x\mathrm{d}x = C_i \int_\Sigma w(x)\psi_i(x)\psi_j(x)\mathrm{d}x$$

or equivalently

$$\int_\Sigma w(x)\psi_i(x)\psi_j(x)(x - C_i)\mathrm{d}x = 0$$

Since this is true for all $i, j = 1, \ldots, p$ and $\{\theta_k\}_{k=1}^p$ span the space of degree p or less polynomials, we have that

$$\int_\Sigma w(x)\psi_i(x)q(x)(x - C_i)\mathrm{d}x = 0$$

for any $q \in \mathbb{P}^p(\Sigma)$. Let

$$q(x) = \prod_{\substack{k=1\\k\neq i}}^p x - C_k$$

then

$$\begin{aligned} 0 &= \int_\Sigma w(x)\psi_i(x)q(x)(x - C_i)\mathrm{d}x \\ &= \int_\Sigma w(x)\psi_i(x)\prod_{k=1}^p x - C_k\mathrm{d}x. \end{aligned}$$

Since this holds for all $i = 1, \ldots, p$, $\prod_{k=1}^p x - C_k$ is w-orthogonal to $\mathbb{P}^p(\Sigma)$. Also, notice that the polynomials $(x - C_k)\psi_k(x)$ are also orthogonal to $\mathbb{P}^p(\Sigma)$ by the double orthogonality conditions.

Now, notice that the orthogonal compliment of $\mathbb{P}^p(\Sigma)$ over $\mathbb{P}^{p+1}(\Sigma)$ is one dimensional. Thus, both $(x - C_k)\psi_k(x)$ and $\prod_{k=1}^p x - C_k$ span the orthogonal compliment. Hence, we have the following equality

$$(x - C_k)\psi_k(x) = \alpha_k \prod_{k=1}^p x - C_k.$$

Dividing through by $(x - C_k)$ gives

$$\psi_k(x) = \alpha_k \prod_{\substack{k=1\\k\neq i}}^p x - C_k$$

for $k = 1, \ldots, p$. $\square$

Thus, in order to compute these multidimensional double orthogonal polynomials, one merely needs to compute the 1D Gaussian quadrature nodes corresponding to the weights ρ_i then use the global to local function γ to distribute the coefficients C_{kn} accordingly.

Chapter 5

Stochastic Optimization

With an understanding, theoretically and numerically, of how to solve SPDEs, one can formulate an optimization problem governed by an SPDE. This chapter focuses on the formulation of SPDE constrained optimization problems as constrained optimization problems in general Banach spaces. Applying adjoint theory to the resulting optimization problems yield algorithms for both gradient and Hessian computations. First, I will present some theory from the study of equality constrained optimization problems in general Banach spaces. With this theory, I will develop the functional analytic setting for optimization problems governed by SPDEs. I will present algorithms for gradient and Hessian computations using adjoints. When using stochastic collocation to solve these optimization problems, gradient and Hessian computations can be parallelized. I will present a new parallel algorithm for solving the state, adjoint, and computing the objective function. Also, I will present a new parallel algorithm for the Hessian vector product. These algorithms are parallel, but there is no apparent way to make Newton's method parallel. I will also point out an approximation of the Newton system that results in a form of Newton's method with inexact Hessian information. Finally, I present a numerical example displaying the necessity of solve the stochastic problem. For this numerical example, I also compare the numerical convergence history of the Newton's method with inexact derivative information compared with that of Newton's method.

5.1 Optimization in General Banach Spaces

Suppose Z, U, Y are Banach space and Z, U are reflexive. Also, let $J : U \times Z \longrightarrow \mathbb{R}$ and $e : U \times Z \longrightarrow Y$ be continuous. This thesis will consider optimization problems of the form

$$\min_{(u,z)\in U\times Z} J(u,z) \tag{5.1.1}$$

subject to

$$e(u,z) = 0,\; u \in U_{ad},\; z \in Z_{ad}, \tag{5.1.2}$$

where U_{ad} and Z_{ad} denote admissible subsets of the state and control spaces, U and Z. If $U \subset \mathbb{R}^l$, $Z \subset \mathbb{R}^m$, and $Y \subset \mathbb{R}^n$, then this is a finite dimensional optimization problem. Much is known about such problems and under certain assumptions, one can prove existence and uniqueness of solutions to this problem. This thesis deals with the case when U, Z, Y are functions spaces. As with the finite dimensional problem, one can show existence and uniqueness of solutions given the right assumptions. The following theorem corresponds to Theorem 1.45 from [16].

Theorem 5.1.1 *Consider problem (5.1.1) and suppose*

1. *$Z_{ad} \subset Z$ is convex, bounded, and closed;*
2. *$U_{ad} \subset U$ is convex, closed, and contains a feasible point;*
3. *$e(u,z) = 0$ has a bounded solution operator, $u : Z \longrightarrow U$;*
4. *the mapping $(u,z) \mapsto e(u,z)$ is continuous under weak convergence;*
5. *J is sequentially lower semicontinuous.*

Then there exists a solution to (5.1.1).

The above result ensures existence of an optimal solution, but uniqueness of the solution is problem dependent. If the solution exists how does one compute the solution? As with finite dimensional optimization, there are first and second order necessary and sufficient conditions for a point to be a solution. These conditions involve the gradient of the objective function being zero at the point and the Hessian being positive. As such, one must have first and second order derivative information. These computations require the notions of Gâteaux and Fréchet differentiability.

Definition 5.1.2 *Suppose $F : X \to Y$ where X, Y are Banach spaces.*

1. *F is Gâteaux differentiable at $x \in X$ if*

$$dF(x,h) = \lim_{t \to 0^+} \frac{F(x+th) - F(x)}{t} \in Y$$

and the mapping $F'(x) : h \mapsto dF(x,h)$ is bounded and linear.

2. *F is Fréchet differentiable at $x \in X$ if it is Gâteaux differentiable at $x \in X$ and*

$$\|F(x+h) - F(x) - F'(x)h\|_Y = o(\|h\|_X) \text{ as } \|h\|_X \to 0.$$

With these definitions, one can compute derivatives of the objective function J. Many times, one can invoke the implicit function theorem to find a solution $u(z)$ to the constraint equation $e(u,z) = 0$. This leads to an implicitly defined objective function $\widehat{J}(z) := J(u(z), z)$. I will now focus on computing derivatives of $\widehat{J}$ [14].

Computing the Fréchet derivative of $\widehat{J}$ applied to a direction $s \in Z$ yields

$$\langle \widehat{J}'(z), s \rangle_{Z^*,Z} = \langle J_u(u(z),z), u_z(z)s \rangle_{U^*,U} + \langle J_z(u(z),z), s \rangle_{Z^*,Z}.$$

Now, computing the derivative of the constraint e with respect to z and applying it to $(u(z), z)$, the implicit solution to $e(u,z) = 0$,

$$e_z(u(z),z)s = e_u(u(z),z)u_z(z)s + e_z(u(z),z)s = 0.$$

Solving for $u_z(z)$, gives

$$u_z(z)s = -e_u(u(z),z)^{-1}e_z(u(z),z)s.$$

Plugging this into the expression for $\widehat{J}'(z)$,

$$\langle \widehat{J}'(z), s\rangle_{Z^*,Z} = -\langle J_u(u(z),z), e_u(u(z),z)^{-1}e_z(u(z),z)s\rangle_{U^*,U} + \langle J_z(u(z),z), s\rangle_{Z^*,Z}.$$

and applying the adjoint to the $u_z(z)$ term,

$$\begin{aligned}\langle \widehat{J}'(z), s\rangle_{Z^*,Z} &= -\langle e_z(u(z),z)^* e_u(u(z),z)^{-*} J_u(u(z),z), s\rangle_{Z^*,Z} + \langle J_z(u(z),z), s\rangle_{Z^*,Z}\\ &= \langle -e_z(u(z),z)^* e_u(u(z),z)^{-*} J_u(u(z),z) + J_z(u(z),z), s\rangle_{Z^*,Z}.\end{aligned}$$

Defining the adjoint state, p, as the solution to

$$e_u(u(z),z)^* p = -J_u(u(z),z)$$

the derivative of $\widehat{J}$ becomes

$$\widehat{J}'(z) = e_z(u(z),z)^* p + J_z(u(z),z).$$

One can derive this equation from the Lagrangian as well. Define the Lagrangian as

$$L(u,z,p) = J(u,z) + \langle p, e(u,z)\rangle_{U^*,U}.$$

Let $u = u(z)$ be the solution to $e(u,z) = 0$ via the implicit function theorem, then

$$\widehat{J}(z) = J(u(z),z) = J(u(z),z) + \langle p, e(u(z),z)\rangle_{U^*,U} = L(u(z),z,p).$$

From Lagrangian multiplier theory, the optimal solution is obtained when the gradient of the Lagrangian is equal to zero. Thus, computing the Fréchet derivative of $\widehat{J}'(z)$ and equating to zero, one has that for any $s \in Z$,

$$\begin{aligned}\langle \widehat{J}'(z), s\rangle_{Z^*,Z} =& \langle L_u(u(z),z,p), s\rangle_{Z^*,Z}\\ =& \langle L_u(u(z),z,p), u_z(z)s\rangle_{U^*,U} + \langle L_z(u(z),z), s\rangle_{Z^*,Z}.\end{aligned}$$

Explicitly writing down the derivative, for any $v \in U$

$$\begin{aligned}\langle L_u(u(z),z,p), v\rangle_{U^*,U} &= \langle J_u(u(z),z), v\rangle_{U^*,U} + \langle p, e_u(u(z),z)v\rangle_{Y^*,Y}\\ &= \langle J_u(u(z),z) + e_u(u(z),z)^* p, v\rangle_{U^*,U}.\end{aligned}$$

Therefore, finding $p = p(z)$ such that $L_u(u(z), z, p) = 0$ (using the implicit function theorem) is equivalent to solving the adjoint equation. Furthermore, we notice that

$$\begin{aligned} \langle L_z(u(z), z, p(z)), s\rangle_{Z^*,Z} &= \langle J_z(u(z), z), s\rangle_{Z^*,Z} + \langle p, e_z(u(z), z)s\rangle_{Y^*,Y} \\ &= \langle J_z(u(z), z) + e_z(u(z), z)^* p, s\rangle_{Z^*,Z}. \end{aligned}$$

This gives rise to the following algorithm for computing the derivative of $\widehat{J}(z)$

Algorithm 5.1.3 (Derivative Computation via Adjoints)

1. *Solve* $e(u, z) = 0$ *for* $u = u(z)$
2. *Solve* $e_u(u(z), z)^* p = -J_u(u(z), z)$ *for* $p = p(z)$
3. *Compute* $\widehat{J}'(z) = e_z(u(z), z)^* p(z) + J_z(u(z), z)$.

The Lagrangian approach is a powerful tool in the Hessian computations to come. Since $\widehat{J}(z) = L(u(z), z, p(z))$, one can compute the Hessian as

$$\widehat{J}''(z) = L_{zu}(u(z), z, p(z))u_z(z) + L_{zz}(u(z), z, p(z)) + L_{zp}(u(z), z, p(z))p_z(z).$$

From the equality $L_u(u(z), z, p(z)) = 0$, differentiating $L_u(u(z), z, p(z))$ with respect to u gives

$$L_{uu}(u(z), z, p(z))u_z(z) + L_{uz}(u(z), z, p(z)) + L_{up}(u(z), z, p(z))p_z(z) = 0. \tag{5.1.3}$$

Since $L_u(u, z, p) = J_u(u, z) + e_u(u, z)^* p$, notice that

$$L_{up}(u, z, p) = J_{up}(u, z) + e_u(u, z)^* + e_{up}(u, z)^* p = e_u(u, z)^*.$$

Also, from the gradient computation,

$$u_z(z) = -e_u(u(z), z)^{-1} e_z(u(z), z)$$

and plugging this into (5.1.3), gives

$$\begin{aligned} &-L_{uu}(u(z), z, p(z))e_u(u(z), z)^{-1} e_z(u(z), z) \\ &\qquad + L_{uz}(u(z), z, p(z)) + e_u(u(z), z)^* p_z(z) = 0. \end{aligned}$$

Solving this for p_z, yields

$$p_z(z) = e_u(u(z),z)^{-*}[L_{uu}(u(z),z,p(z))e_u(u(z),z)^{-1}e_z(u(z),z) - L_{uz}(u(z),z,p(z))].$$

Combining all of these facts, the Hessian of $\widehat{J}(z)$ is

$$\begin{aligned}\widehat{J}''(z) = & - L_{zu}(u(z),z,p(z))e_u(u(z),z)^{-1}e_z(u(z),z) + L_{zz}(u(z),z,p(z)) \\ & + L_{zp}(u(z),z,p(z))e_u(u(z),z)^{-*}[L_{uu}(u(z),z,p(z))e_u(u(z),z)^{-1}e_z(u(z),z) \\ & - L_{uz}(u(z),z,p(z))].\end{aligned}$$

Notice that $L_{zp}(u,z,p) = e_z(u,z)$ and hence

$$\begin{aligned}\widehat{J}''(z) = & - L_{zu}(u(z),z,p(z))e_u(u(z),z)^{-1}e_z(u(z),z) + L_{zz}(u(z),z,p(z)) \\ & + e_z(u(z),z)^*e_u(u(z),z)^{-*}[L_{uu}(u(z),z,p(z))e_u(u(z),z)^{-1}e_z(u(z),z) \\ & - L_{uz}(u(z),z,p(z))].\end{aligned}$$

By defining

$$W(u(z),z) = \begin{pmatrix} -e_u(u,z)^{-1}e_z(u,z) \\ I \end{pmatrix},$$

the Hessian becomes

$$\widehat{J}''(z) = W(u(z),z)^T \begin{pmatrix} L_{uu}(u(z),z,p(z)) & L_{uz}(u(z),z,p(z)) \\ L_{zu}(u(z),z,p(z)) & L_{zz}(u(z),z,p(z)) \end{pmatrix} W(u(z),z).$$

This form is not typically used to compute the Hessian directly, but rather the Hessian applied to a vector v, $\widehat{J}''(z)v$. This yields the following algorithm.

Algorithm 5.1.4 (Hessian Times a Vector Computation)

1. *Solve* $e(u,z) = 0$ *for* $u = u(z)$
2. *Solve* $e_u(u(z),z)^*p = -J_u(u(z),z)$ *for* $p = p(z)$
3. *Solve* $e_u(u(z),z)w = e_z(u(z),z)v$ *for* $w = w(z,v)$

4. *Solve* $e_u(u(z), z)^* q = L_{uu}(u(z), z, p(z))w - L_{uz}(u(z), z, p(z))v$ *for* $q = q(z, v)$

5. *Compute* $\widehat{J}''(z)v = e_z(u(z), z)^* q - L_{zu}(u(z), z, p(z))w + L_{zz}(u(z), z, p(z))v$.

With an understanding of the first and second order derivatives of the objective function, one can formulate a general version of Newton's method. Suppose $z^* \in Z$ is a minimizer of (5.1.1), then $\widehat{J}'(z^*) = 0$ and $\widehat{J}''(z^*)$ is positive semidefinite (i.e. for all $v \in Z$, $\langle \widehat{J}''(z^*)v, v\rangle_{Z^*,Z} \geq 0$). By continuity of the derivatives of $\widehat{J}$, the Hessian is also positive semidefinite in some neighborhood of z^*. Thus, if $z_0 \in Z$ is "close enough" to z^*, then $\widehat{J}''(z_0)$ is positive and one can apply Newton's method to the gradient $\widehat{J}'(z)$ in order to recover z^*. Newton's method seeks the roots of nonlinear equations by linearizing the equations and finding roots of the linear model. Given the equation $\widehat{J}'(z) = 0$ and a point z_0, the linearization is

$$-\widehat{J}'(z_0) = \widehat{J}''(z_0)(z_0 - z).$$

Finding the zero of this linearized model gives the Newton step. To compute the Newton step, one can employ the following algorithm:

Algorithm 5.1.5 (Newton's Method)

1. *Given the current iterate* z_k

2. *Solve* $\widehat{J}''(z_k)p_k = -\widehat{J}'(z_k)$

3. *Compute* $z_{k+1} = z_k + p_k$.

The main difficulty of this algorithm is step 2, computing p_k. Since there is an algorithm for computing Hessian vector products, one can apply the conjugate gradient algorithm to compute this step. The resulting algorithm is called Newton-CG. The conjugate gradient method only approximately solves the Newton system. This fact leads to a general class of methods called inexact Newton's methods. These methods do not require exact Hessian information, but rather Hessian approximations $H_k \approx \widehat{J}''(z_k)$.

Algorithm 5.1.6 (Inexact Newton's Method)

1. *Given the current iterate* z_k

2. *Solve* $H_k p_k = -\widehat{J}'(z_k)$

3. *Compute* $z_{k+1} = z_k + p_k$.

The convergence of such methods properties of such methods may be seen in the the following lemma.

Lemma 5.1.7 *Suppose* $\widehat{J}$ *is twice continuously differentiable,* $z_* \in Z$ *is a point at which the second order sufficiency optimality conditions are satisfied, and suppose the iterates* $z_{k+1} = z_k + s_k$ *where* s_k *solves*

$$H_k s_k = -\widehat{J}'(z_k).$$

Furthermore, suppose $\widehat{J}''(z)$ *is Lipschitz with constant* L *and* H_k *is invertible for all* k*, then*

$$\|z_{k+1} - z_*\| \leq \frac{L}{2}\|H_k^{-1}\|\|z_k - z_*\|^2 + \|H_k^{-1}(H_k - \widehat{J}''(z_k))\|\|z_k - z_*\|.$$

Proof: Note that since the second order optimality conditions are satisfied, $\widehat{J}'(z_*) = 0$. Notice that

$$\begin{aligned}
z_{k+1} - z_* =& z_k + s_k - z_* \\
=& (x_k - x_*) + H_k^{-1}\widehat{J}'(z_k) \\
=& H_k^{-1}\Big[H_k(z_k - z_*) + (\widehat{J}'(z_*) - \widehat{J}'(z_k))\Big] \\
=& H_k^{-1}\Bigg[\widehat{J}''(z_k)(z_k - z_*) \\
& - \int_0^1 \widehat{J}''(z_k + t(z_k - z_*))(z_k - z_*)\mathrm{d}t + (H_k - \widehat{J}''(z_k))(z_k - z_*)\Bigg] \\
=& H_k^{-1}\Bigg[\int_0^1 (\widehat{J}''(z_k) - \widehat{J}''(z_k + t(z_k - z_*)))(z_k - z_*)\mathrm{d}t \\
& + (H_k - \widehat{J}''(z_k))(z_k - z_*)\Bigg].
\end{aligned}$$

Taking norms on both sides and applying submultiplicativity and the triangle inequality yields

$$\begin{aligned}
\|z_{k+1} - z_*\| &= \|H_k^{-1}\| \int_0^1 \|(\widehat{J}''(z_k) - \widehat{J}''(z_k + t(z_k - z_*)))\| \|z_k - z_*\| \mathrm{d}t \\
&\quad + \|H_k^{-1}(H_k - \widehat{J}''(z_k))\| \|z_k - z_*\| \\
&\leq \|H_k^{-1}\| \int_0^1 L\|z_k - t(z_k - z_*)\| \mathrm{d}t \|z_k - z_*\| \\
&\quad + \|H_k^{-1}(H_k - \widehat{J}''(z_k))\| \|z_k - z_*\| \\
&= \|H_k^{-1}\| \frac{L}{2} \|z_k - z_*\|^2 + \|H_k^{-1}(H_k - \widehat{J}''(z_k))\| \|z_k - z_*\|.
\end{aligned}$$

Thus, giving the desired inequality. □

Now, suppose there exists $R \in (0,1)$ such that $\|H_k^{-1}(H_k - \widehat{J}''(z_k))\| \leq R$, then the inexact Newton's method converges q-linearly. More results concerning inexact Newton's method may be found in [20].

The approximate Hessian described above can either result from the use of an exact method for solving the Newton system or can result from a Hessian approximation scheme. One possible scheme, which will be described later, is to approximate the Hessian, $\widehat{J}''(z_k)$, with some matrix H_k. This research focuses on an approximation H_k, in the discrete sense, arising from a course discretization in the stochastic domain. One may also use this this approximation to precondition the conjugate gradient method when solving the Newton system. The preconditioned Newton-CG scheme is as follows.

Algorithm 5.1.8 (Preconditioned Newton-CG)

```
1. Given the current iterate $z_k$
2. Solve $\widehat{J}''(z_k)p_k = -\widehat{J}'(z_k)$ using preconditioned CG with preconditioner $H_k^{-1}$;
3. Compute $z_{k+1} = z_k + p_k$.
```

In general, one cannot compute the approximation H_k, but can compute products of the form $H_k v$. Thus, in order to precondition the conjugate gradient method, one can only approximately apply the preconditioner (i.e. using CG to solve $H_k v = w$). This complication requires the flexible conjugate gradient method. In the following algorithm, $w = H_k^{-1} v$ is approximated by $w \approx \mathcal{H} v$ where $\mathcal{H}$ refers to some nonlinear function that approximately applies the preconditioner H^{-1}. Furthermore, the following algorithm is presented for a general linear system $Au = b$.

Algorithm 5.1.9 (Flexible Conjugate Gradient)

```
1. Given $u_0$ arbitrary and $m_{\max}$ a nonnegative integer;
2. Initialize $r_0 = b - Au_0$ and $m_0 = 0$;
3. for $i = 0, 1, \dots$
    (a) Apply preconditioner:
            $w_i = \mathcal{H}(r_i);$
    (b) Compute new direction:
            $d_i = w_i - \sum_{k=i-m_i}^{i-1} \frac{w_i^\top A d_k}{d_k^\top A d_k} d_k;$
    (c) Update approximate solution:
            $u_{i+1} = u_i + \frac{d_i^\top r_i}{d_i^\top A d_i} d_i;$
    (d) Update residual:
            $r_{i+1} = r_i - \frac{d_i^\top r_i}{d_i^\top A d_i} A d_i;$
    (e) Update truncation parameter:
            $m_i = \max(1, i \mod (m_{\max} + 1)).$
```

For more information on flexible conjugate gradient, see [28].

In practice, some globalization technique for Newton's (or inexact Newton's) method should be employed. Linesearch techniques are popular choices for their ease of implementation and relatively low computational cost. A linesearch algorithm attempts to find an optimal step size α_k and generates the iterate $z_{k+1} = z_k + \alpha_k p_k$. The step size is required to satisfy the Armijo condition (or sufficient decrease condition)

$$J(z_k + \alpha_k p_k) \leq J(z_k) + c\alpha_k J'(z_k)p_k,$$

where $c \in (0,1)$ and is typically quite small, e.g. $c = 10^{-4}$. For more information on linesearch methods see [7]

5.2 Optimization Governed by Linear Elliptic SPDEs

The above theory is readily applied to optimization problems governed by SPDEs. In this section, I will consider problems of the form:

$$\min_{z \in Z} \widehat{J}(z) \equiv J(u(z), z)$$

where $u(z;\cdot,\cdot) = u(\cdot,\cdot)$ is the weak solution of

$$\begin{aligned} -\nabla \cdot \Big(a(\omega, x)\nabla u(\omega, x)\Big) &= z(x) && \text{for } x \in D, P\text{-a.e.} \\ u(\omega, x) &= 0 && \text{for } x \in \partial D, P\text{-a.e.} \end{aligned}$$

First of all, one must define the spaces of the control and state variables. As seen in previous chapters the solutions to linear elliptic SPDEs live in the space $U = L^2_P(\Omega; H^1_0(D))$. I will assume finite dimensional noise and thus the state space is equivalent to $U = L^2_\rho(\Gamma; H^1_0(D))$. Furthermore, I will assume the control space $Z = L^2(D)$. Note that the space $Z = L^2(D) \subset L^2_P(\Gamma; L^2(D))$. The elements of Z may be thought as degenerate random variables, or constant mappings from Γ to Z. The constraint function is defined as the weak form of the SPDE, for all $v \in U$,

$$\langle e(u,z), v\rangle_{U^*,U} = \int_\Gamma \rho(y) \int_D \Big(a(y,x)\nabla u(y,x) \cdot \nabla v(y,x) - z(y,x)v(y,x)\Big)\mathrm{d}x\mathrm{d}y.$$

Thus, the constraint function $e : U \times Z \to U^*$ and $Y = U^*$.

In the previous section, I formulate the gradient and Hessian, for the general problem. The gradient is given by

$$\widehat{J}'(z) = -e_z(u(z), z)^* e_u(u(z), z)^{-*} J_u(u(z), z) + J_z(u(z), z)$$

and thus one needs to compute the derivatives $e_z(u, z)$ and $e_u(u, z)$. Notice that the constraint function e acts linearly with respect to the state u and the control z, thus the corresponding derivative of e with respect to z applied to the direction $\delta z \in Z$ is give by: for all $v \in U$

$$\langle e_z(u, z)\delta z, v\rangle_{U^*,U} = -\int_\Gamma \rho(y) \int_D \delta z(y, x)v(y, x)\mathrm{d}x\mathrm{d}y$$

and, similarly, for any direction $\delta u \in U$ and for all $v \in U$, the derivative of e with respect to u is given by

$$\langle e_u(u, z)\delta z, v\rangle_{U^*,U} = \int_\Gamma \rho(y) \int_D a(y, x)\nabla\delta u(y, x) \cdot \nabla v(y, x)\mathrm{d}x\mathrm{d}y.$$

Notice that both of these derivatives are symmetric, so $e_u(u, z)^* = e_u(u, z)$ and $e_z(u, z)^* = e_z(u, z)$. From here, one can compute the adjoint as

$$e_u(u(z), z)p = -J_u(u(z), z).$$

Applying this to some direction $v \in U$,

$$\langle e_u(u(z), z)p, v\rangle_{U^*,U} = \int_\Gamma \rho(y) \int_D a(y, x)\nabla p(y, x) \cdot \nabla v(y, x)\mathrm{d}x\mathrm{d}y = -J_u(u(z), z)v.$$

Therefore, the adjoint is a solution to an SPDE with the same left hand side as the state equation. Under the finite dimensional noise assumption, the Doob-Dynkins lemma implies that the adjoint is characterized by the same finite number of random variables that characterize the state.

In computing the Hessian, one needs second order derivative information of the constraint function as well. Namely, we will need e_{zz},$e_{zu} = e_{uz}$, and e_{uu}. Since the constraint is linear in both z and u, the second order derivatives, e_{zz} and e_{uu} are zero. This is easy to see from the first order derivatives, e_u and e_z. Next notice that $e_{uz}(u,z) = 0$ since the first term of e does not depend on z and the second term of e does not depend on u. With these ideas, one can compute the first and second order derivatives of a given objective function. In the following section, I will construct a simple example of a distributed control problem governed by a linear elliptic SPDE and discuss how to solve the example numerically.

5.3 Derivatives of the Objective Function

For optimization problems governed by elliptic SPDEs, the state space is the stochastic Sobolev space $U = L^2_\rho(\Gamma, H^1(D))$. Furthermore, for these problems, I will consider the control space $Z = L^2(D)$, where Z is associated with the space of constant mappings from Γ to $L^2(D)$. In this section, $\langle\cdot,\cdot\rangle_O$ will denote some inner product on $H^1(D)$ and $\|\cdot\|_O$ will denote the norm induced by the inner product. This inner product will typically be the $O = H^1(D)$ or $O = L^2(D)$ inner product. Now, define the objective function, for $\alpha > 0$, as

$$J(u,z) = \frac{1}{2}E\Big[f\Big(\|u(y,\cdot) - u_0\|_O^2\Big)\Big] + \frac{1}{2}g\Big(E\Big[\|u(y,\cdot) - u_0\|_O^2\Big]\Big) + \frac{\alpha}{2}\|z\|^2_{L^2(D)} \quad (5.3.1)$$

where $u_0 \in H^1(D)$ is some desired distribution and

$$f : [0,\infty) \to [0,\infty),$$

$$g : [0,\infty) \to [0,\infty),$$

are twice continuously differentiable functions.

Remark 5.3.1 • *For the expected value*

$$J(u,z) = \frac{1}{2}E\Big[\|u(y,\cdot) - u_0\|_O^2\Big] + \frac{\alpha}{2}\|z\|^2_{L^2(D)},$$

take $f(s) = s$ and $g(s) = 0$.

- *For the variance*

$$J(u,z) = \frac{1}{2} Var\Big[\|u(y,\cdot) - u_0\|_O^2\Big] + \frac{\alpha}{2}\|z\|_{L^2(D)}^2,$$

take $f(s) = s^2$ and $g(s) = -s^2$.

Let

$$\begin{aligned} \widehat{J}(z) &\equiv J(u(z;\cdot,\cdot),z) \\ &= \frac{1}{2}E\Big[f\Big(\|u(z;y,\cdot) - u_0\|_O^2\Big)\Big] + \frac{1}{2}g\Big(E\Big[\|u(yz;,\cdot) - u_0\|_O^2\Big]\Big) + \frac{\alpha}{2}\|z\|_{L^2(D)}^2, \end{aligned} \tag{5.3.2}$$

where $u(z;\cdot,\cdot) = u(\cdot,\cdot)$ is the weak solution of

$$\begin{aligned} -\nabla\cdot\Big(a(y,x)\nabla u(y,x)\Big) &= z(x) && \text{for } (y,x) \in \Gamma \times D \\ u(y,x) &= 0 && \text{for } (y,x) \in \Gamma \times \partial D, \end{aligned}$$

and consider the distributed control problem:

$$\min_{z\in Z} \widehat{J}(z).$$

This section is devoted to computing the derivatives of objective functions (5.3.2). First, by the chain rule, the derivative of J with respect to u in the direction δu is

$$\begin{aligned} \langle J_u(u,z), \delta u\rangle_{U^*,U} =& E\Big[f'\Big(\|u(y,\cdot) - u_0\|_O^2\Big)\langle u(y,\cdot) - u_0, \delta u(y,\cdot)\rangle_O\Big] \\ &+ g'\Big(E\Big[\|u(y,\cdot) - u_0\|_O^2\Big]\Big)E\Big[\langle u(y,\cdot) - u_0, \delta u(y,\cdot)\rangle_O\Big] \end{aligned}$$

Similarly, the derivative of J with respect to to z in the $\delta z \in Z$ direction is

$$\langle J_z(u,z), \delta z\rangle_{Z^*,Z} = \alpha\langle z, \delta z\rangle_Z.$$

With these computations, one can compute the first derivative of $\widehat{J}(z)$ via the adjoint approach. Recall that

$$\widehat{J}'(z) = e_z(u(z;\cdot,\cdot),z)p(z;\cdot,\cdot) + J_z(u(z;\cdot,\cdot),z)$$

where $p(\cdot,\cdot) = p(z;\cdot,\cdot) \in L^2_\rho(\Gamma; H^1_0(D))$ solves the adjoint equation

$$e_u(u(z;\cdot,\cdot), z)p = -J_u(u(z;\cdot,\cdot), z).$$

or equivalently

$$\begin{aligned}
&\int_\Gamma \rho(y) \int_D a(y,x)\nabla p(y,x) \cdot \nabla v(y,x)\mathrm{d}x\mathrm{d}y \\
&\quad = -\int_\Gamma \rho(y) f'\Big(\|u(z;y,\cdot) - u_0\|^2_O\Big)\langle u(z;y,\cdot) - u_0, v(y,\cdot)\rangle_O \mathrm{d}y \\
&\quad - g'\Big(\int_\Gamma \rho(y)\|u(z;y,\cdot) - u_0\|^2_O \mathrm{d}y\Big)\int_\Gamma \rho(y)\langle u(z;y,\cdot) - u_0, v(y,\cdot)\rangle_O \mathrm{d}y \\
&\quad \forall v \in L^2_\rho(\Gamma; H^1_0(D)).
\end{aligned}$$

By the Doob-Dynkin's lemma, the stochasticity of the solution to the adjoint equation depends on the random variables describing the stochasticity of the diffusivity coefficients a and the solution to the state equation u. Thus, no new randomness is introduced through the adjoint equation. With the solution ,$p(z;\cdot,\cdot)$ to the adjoint equation, the first derivative of $\widehat{J}$ in the direction $w \in Z$ is

$$\begin{aligned}
\langle \widehat{J}'(z), w\rangle_{Z^*,Z} =& \langle e_z(u(z;\cdot,\cdot), z)p(z;\cdot,\cdot) + J_z(u(z;\cdot,\cdot), z), w\rangle_{Z^*,Z} \\
=& -\int_\Gamma \rho(y) \int_D p(z;y,x)w(x)\mathrm{d}x\mathrm{d}y + \alpha \int_D z(x)w(x)\mathrm{d}x.
\end{aligned}$$

The algorithm for first order derivative of $\widehat{J}$ is

Algorithm 5.3.2 (Derivative Computation via Adjoints)

1. *Compute the solution $u(\cdot,\cdot) = u(z;\cdot,\cdot)$ of the state equation*

$$\int_\Gamma \rho(y) \int_D a(y,x)\nabla u(z;y,x) \cdot \nabla v(y,x) = \int_\Gamma \rho(y) \int_D z(x)v(y,x)\mathrm{d}x\mathrm{d}y$$
$$\forall v \in L^2_\rho(\Gamma; H^1_0(D));$$

2. Compute the solution $p(\cdot,\cdot) = p(z;\cdot,\cdot)$ of the adjoint equation

$$\begin{aligned}
&\int_\Gamma \rho(y)\int_D a(y,x)\nabla p(y,x)\cdot\nabla v(y,x)\mathrm{d}x\mathrm{d}y \\
&\quad = -\int_\Gamma \rho(y) f'\Big(\|u(z;y,\cdot)-u_0\|_O^2\Big)\langle u(z;y,\cdot)-u_0, v(y,\cdot)\rangle_O \mathrm{d}y \\
&\quad\quad - g'\Big(\int_\Gamma \rho(y)\|u(z;y,\cdot)-u_0\|_O^2\mathrm{d}y\Big)\int_\Gamma \rho(y)\langle u(z;y,\cdot)-u_0, v(y,\cdot)\rangle_O\mathrm{d}y \\
&\quad \forall v\in L^2_\rho(\Gamma;H^1_0(D)).
\end{aligned}$$

3. Compute the first derivative in the direction $w\in Z$ as

$$\langle \widehat{J}'(z), w\rangle_{Z^*,Z} = -\int_\Gamma \rho(y)\int_D p(z;y,x)w(x)\mathrm{d}x\mathrm{d}y + \alpha\int_D z(x)w(x)\mathrm{d}x.$$

For Hessian computations, one must compute J_{uu}, $J_{uz} = J_{zu}$, and J_{zz}. J depends quadratically on z, thus the mapping $z\mapsto J_z z(u,z)$ is a constant mapping and the derivative in the directions $\delta z_1, \delta z_2 \in Z$ is

$$\langle J_z z(u,z)\delta z_1, \delta z_2\rangle_{Z^*,Z} = \alpha\langle\delta z_1,\delta z_2\rangle_Z.$$

Since the objective function (5.3.1) is a sum of two parts, one which does not explicitly depend on z and the other which does not explicitly depend on u, the mixed derivatives $J_{uz} = J_{zu} = 0$. Finally, to compute J_{uu}, one must employ the chain rule and the product rule. These differentiation rules give for any directions $\delta u_1, \delta u_2\in U$,

$$\begin{aligned}
&\langle J_{uu}(u,z)\delta u_1,\delta u_2\rangle_{U^*,U} \\
&\quad = E\Big[2f''\Big(\|u(y,\cdot)-u_0\|_O^2\Big)\langle u(y,\cdot)-u_0,\delta u_1(y,\cdot)\rangle_O\langle u(y,\cdot)-u_0,\delta u_2(y,\cdot)\rangle_O \\
&\quad\quad + f'\Big(\|u(y,\cdot)-u_0\|_O^2\Big)\langle\delta u_1(y,\cdot),\delta u_2(y,\cdot)\rangle_O\Big] \\
&\quad\quad + 2g''\Big(E\Big[\|u(y,\cdot)-u_0\|_O^2\Big]\Big)E\Big[\langle u(y,\cdot)-u_0,\delta u_1(y,\cdot)\rangle_O\langle u(y,\cdot)-u_0,\delta u_2(y,\cdot)\rangle_O\Big] \\
&\quad\quad + g'\Big(E\Big[\|u(y,\cdot)-u_0\|_O^2\Big]\Big)E\Big[\langle\delta u_1(y,\cdot),\delta u_2(y,\cdot)\rangle_O\Big].
\end{aligned}$$

These computations, along with those from the previous section give the following algorithm for the Hessian times a vector computation.

Algorithm 5.3.3 (Hessian Times a Vector Computation)

1. *Compute the solution* $u(\cdot,\cdot) = u(z;\cdot,\cdot)$ *of the state equation*

$$\int_\Gamma \rho(y) \int_D a(y,x)\nabla u(y,x)\cdot\nabla s(y,x) - z(x)s(y,x)\mathrm{d}x\mathrm{d}y = 0 \quad \forall s \in L^2_\rho(\Gamma; H^1_0(D));$$

2. *Compute the solution* $p(\cdot,\cdot) = p(z;\cdot,\cdot)$ *of the adjoint equation*

$$\begin{aligned}&\int_\Gamma \rho(y) \int_D a(y,x)\nabla p(y,x)\cdot\nabla v(y,x)\mathrm{d}x\mathrm{d}y\\ &\quad= - \int_\Gamma \rho(y) f'\Big(\|u(z;y,\cdot) - u_0\|^2_O\Big)\langle u(z;y,\cdot) - u_0, v(y,\cdot)\rangle_O \mathrm{d}y\\ &\quad\quad - g'\Big(\int_\Gamma \rho(y)\|u(z;y,\cdot)-u_0\|^2_O\mathrm{d}y\Big)\int_\Gamma \rho(y)\langle u(z;y,\cdot) - u_0, v(y,\cdot)\rangle_O\mathrm{d}y\\ &\quad\quad \forall v \in L^2_\rho(\Gamma; H^1_0(D)).\end{aligned}$$

3. *Compute the solution* $w(\cdot,\cdot) = w(z;v;\cdot,\cdot)$ *of*

$$\begin{aligned}&\int_\Gamma \rho(y) \int_D a(y,x)\nabla w(y,x)\cdot\nabla s(y,x)\mathrm{d}x\mathrm{d}y\\ &\quad = - \int_\Gamma \rho(y)\int_D v(x)s(y,x)\mathrm{d}x\mathrm{d}y \quad \forall s \in L^2_\rho(\Gamma; H^1_0(D));\end{aligned}$$

4. *Compute the solution* $q(\cdot,\cdot) = q(z;v;\cdot,\cdot)$ *of*

$$\begin{aligned}&\int_\Gamma \rho(y) \int_D a(y,x)\nabla q(y,x)\cdot\nabla s(y,x)\mathrm{d}x\mathrm{d}y\\ &= E\Big[2f''\Big(\|u(y,\cdot)-u_0\|^2_O\Big)\langle u(y,\cdot)-u_0, w(z;v;y,\cdot)\rangle_O\langle u(y,\cdot)-u_0, s(y,\cdot)\rangle_O\\ &\quad + f'\Big(\|u(y,\cdot)-u_0\|^2_O\Big)\langle w(z;v;y,\cdot), s(y,\cdot)\rangle_O\Big]\\ &\quad + 2g''\Big(E\Big[\|u(y,\cdot)-u_0\|^2_O\Big]\Big)E\Big[\langle u(y,\cdot)-u_0, w(z;v;y,\cdot)\rangle_O\langle u(y,\cdot)-u_0, s(y,\cdot)\rangle_O\Big]\\ &\quad + g'\Big(E\Big[\|u(y,\cdot)-u_0\|^2_O\Big]\Big)E\Big[\langle w(z;v;y,\cdot), s(y,\cdot)\rangle_O\Big] \quad \forall s \in L^2_\rho(\Gamma; H^1_0(D));\end{aligned}$$

5. *Compute the Hessian times a vector* v*, in the direction* $s \in Z$*, as*

$$\langle \widehat{J}''(z)v, s\rangle_{Z^*,Z} = - \int_\Gamma \rho(y)\int_D q(z;v;y,x)s(x)\mathrm{d}x\mathrm{d}y + \alpha\int_D v(x)s(x)\mathrm{d}x.$$

5.4 Discretization

In this section I will present two discretization techniques for the example objective function

$$J(u,z) = \frac{1}{2}f\Big(\Big[\|u(y,\cdot)-u_0\|^2_{L^2(D)}\Big]\Big) + \frac{1}{2}\Big[g\Big(\|u(y,\cdot)-u_0\|^2_{L^2(D)}\Big)\Big] + \frac{\alpha}{2}\|z\|^2_{L^2(D)}.$$

The techniques described in this section are based around the stochastic collocation and stochastic Galerkin method for solving SPDEs. Both methods discretize the solution space as a tensor product of finite dimensional subspaces. Suppose $X_h \subset H^1_0(D)$ is a finite dimensional subspace of dimension N and $Y_h \subset L^2_\rho(\Gamma)$ is a finite dimensional subspace of dimension P, then $X_h \otimes Y_h$ is a finite dimensional subspace of the state space U. Similarly, let $Z_h \subset Z$ be a finite dimensional subspace of the control space (with dimension N).

5.4.1 Stochastic Collocation

For the stochastic collocation method, $Y_h = \mathbb{P}^{P-1}(\Gamma)$ and $X_h = \mathrm{span}\{\phi_1,\ldots,\phi_N\}$ is any finite element space. For this research, X_h is the space of piecewise linear functions built on a given mesh $\mathcal{T}_h$. Furthermore, I choose collocation points $\{y_k\}_{k=1}^P$ as the nodes from a multi-dimensional quadrature formula with corresponding weights $\{\omega_k\}_{k=1}^P$. With these collocation points, one can build a the Lagrange basis for Y_h. Let L_k denote the k^{th} Lagrange polynomial (i.e. $L_k(y_j) = \delta_{kj}$). In the collocation framework the expected value is approximated via the natural quadrature rule built on the collocation points. Let $\{\omega_k\}_{k=1}^P$ denote the quadrature weights associated with the collocation points

$$\omega_k = \int_\Gamma \rho(y) L_k^2(y)\mathrm{d}y \quad \text{for } k = 1,\ldots,P.$$

With this framework, the optimization problem

$$\min_{z\in Z} \widehat{J}(z)$$

where

$$\widehat{J}(z) = \frac{1}{2}f\Big(\Big[\|u(z;y,\cdot) - u_0\|^2_{L^2(D)}\Big]\Big) + \frac{1}{2}\Big[g\Big(\|u(z;y,\cdot) - u_0\|^2_{L^2(D)}\Big)\Big] + \frac{\alpha}{2}\|z\|^2_{L^2(D)}.$$

and $u(z;\cdot,\cdot) = u(\cdot,\cdot)$ is the weak solution of

$$\begin{aligned} -\nabla\cdot\Big(a(y,x)\nabla u(y,x)\Big) &= z(x) && \text{for } (y,x)\in\Gamma\times D \\ u(y,x) &= 0 && \text{for } (y,x)\in\Gamma\times\partial D, \end{aligned}$$

is replaced by the discretized optimization problem

$$\min_{z_h\in Z_h} \widehat{J}_{hP}(z_h)$$

where

$$\begin{aligned} \widehat{J}_{hP}(z_h) =& \frac{1}{2}\sum_{k=1}^{P}\omega_k f\Big(\Big\|\sum_{j=1}^{N}(\vec{u}_k(z_h))_j\ \phi_j - u_0\Big\|^2_{L^2(D)}\Big) \\ &+ \frac{1}{2}g\Big(\sum_{k=1}^{P}\omega_k\Big\|\sum_{j=1}^{N}(\vec{u}_k(z_h))_j\ \phi_j - u_0\Big\|^2_{L^2(D)}\Big) \\ &+ \frac{\alpha}{2}\|z_h\|^2_{L^2(D)} \end{aligned}$$

and $\vec{u}_k(z_h) = \vec{u}_k$, $k = 1,\dots,P$, solves

$$K_k\vec{u}_k = M\vec{z}_h \quad \text{for } k = 1,\dots,P.$$

Here K_k is the k^{th} stochastic collocation FEM stiffness matrix

$$(K_k)_{ij} = \int_D a(y_k,x)\nabla\phi_i(x)\cdot\nabla\phi_j(x)\mathrm{d}x.$$

To compute derivatives of the objective function, one must first compute the adjoint state. In the infinite dimensional formulation, the adjoint state solves

$$\begin{aligned} &\int_\Gamma \rho(y)\int_D a(y,x)\nabla p(y,x)\cdot\nabla v(y,x)\mathrm{d}x\mathrm{d}y \\ &\quad= -\int_\Gamma \rho(y)f'\Big(\|u(z;y,\cdot)-u_0\|^2_O\Big)\langle u(z;y,\cdot)-u_0, v(y,\cdot)\rangle_O\mathrm{d}y \\ &\quad\quad - g'\Big(\int_\Gamma\rho(y)\|u(z;y,\cdot)-u_0\|^2_O\mathrm{d}y\Big)\int_\Gamma\rho(y)\langle u(z;y,\cdot)-u_0, v(y,\cdot)\rangle_O\mathrm{d}y \\ &\quad\quad \forall v\in L^2_\rho(\Gamma; H^1_0(D)). \end{aligned}$$

Thus, as in the stochastic collocation section of the numerical solutions chapter, the block FEM system is

$$\underbrace{\begin{pmatrix} K_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & K_P \end{pmatrix}}_{=K} \underbrace{\begin{pmatrix} \vec{p}_1 \\ \vdots \\ \vec{p}_P \end{pmatrix}}_{=\vec{p}} = \underbrace{\begin{pmatrix} \vec{F}_1 \\ \vdots \\ \vec{F}_P \end{pmatrix}}_{=F}.$$

where F_k is defined as

$$\begin{aligned}
(\vec{F}_k)_i =& -f'\Big(\|u(z;y_k,\cdot)-u_0\|_O^2\Big)\int_D (u(z;y_k,x)-u_0(x))\phi_i(x)\mathrm{d}x \\
&- g'\Big(\sum_{\ell=1}^{P}\omega_\ell\|u(z;y_\ell,\cdot)-u_0\|_{L^2(D)}^2\Big)\int_D (u(z;y_k,x)-u_0(x))\phi_i(x)\mathrm{d}x \\
=& -\left(f'\Big(\|u(z;y_k,\cdot)-u_0\|_{L^2(D)}^2\Big)+g'\Big(\sum_{\ell=1}^{P}\omega_\ell\|u(z;y_\ell,\cdot)-u_0\|_{L^2(D)}^2\Big)\right) \\
&\times \int_D \sum_{j=1}^{N}(\vec{u}_k)_j\phi_j(x)\phi_i(x) - u_0(x)\phi_i(x)\mathrm{d}x \\
=& -\left(f'\Big(\|u(z;y_k,\cdot)-u_0\|_{L^2(D)}^2\Big)+g'\Big(\sum_{\ell=1}^{P}\omega_\ell\|u(z;y_\ell,\cdot)-u_0\|_{L^2(D)}^2\Big)\right) \\
&\times \Big(\sum_{j=1}^{N}(\vec{u}_k)_j\int_D \phi_j(x)\phi_i(x)\mathrm{d}x - \int_D u_0(x)\phi_i(x)\mathrm{d}x\Big) \\
=& -\left(f'\Big(\|u(z;y_k,\cdot)-u_0\|_{L^2(D)}^2\Big)+g'\Big(\sum_{\ell=1}^{P}\omega_\ell\|u(z;y_\ell,\cdot)-u_0\|_{L^2(D)}^2\Big)\right) \\
&\times \Big(\sum_{j=1}^{N}M_{ij}(\vec{u}_k)_j - \vec{b}_i\Big)
\end{aligned}$$

where M is the standard FEM mass matrix

$$M_{ij}=\int_D \phi_i(x)\phi_j(x)\mathrm{d}x \quad \text{and} \quad \vec{b}_i=\int_D u_0(x)\phi_i(x)\mathrm{d}x \quad \forall i,j=1,\ldots,N.$$

Therefore, $\vec{F}_k$ has the matrix-vector representation

$$\vec{F}_k=-\left(f'\Big(\|u(z;y_k,\cdot)-u_0\|_{L^2(D)}^2\Big)+g'\Big(\sum_{\ell=1}^{P}\omega_\ell\|u(z;y_\ell,\cdot)-u_0\|_{L^2(D)}^2\Big)\right)\Big(M\vec{u}_k-\vec{b}\Big).$$

With the discretized adjoint state computed, the derivative of $\widehat{J}(z)$ in the direction ϕ_j for $j = 1, \dots, N$ is approximated by

$$\begin{aligned}\langle \widehat{J}'(z), w\rangle_{Z^*,Z} \approx& \langle \widehat{J}'_{hP}(z_h), \phi_j\rangle_{Z^*,Z} \\ =& -\sum_{k=1}^{P}\omega_k \sum_{i=1}^{N}(\vec{p}_k)_i \int_D \phi_i(x)\phi_j(x)\mathrm{d}x + \alpha\sum_{i=1}^{N}(\vec{z}_h)_i \int_D \phi_i(x)\phi_j(x)\mathrm{d}x \\ =& -\sum_{k=1}^{P}\omega_k \sum_{i=1}^{N} M_{ij}(\vec{p}_k)_i + \alpha \sum_{i=1}^{N} M_{ij}(\vec{z}_h)_i.\end{aligned}$$

The matrix-vector notation for the first derivative is

$$\widehat{J}'(z) \approx M(\alpha\vec{z}_h - \sum_{k=1}^{P}\omega_k\vec{p}_k).$$

This brings us to the following discretized gradient computation algorithm.

Algorithm 5.4.1 Stochastic Collocation Gradient Computation

1. *Compute* $K_1, \dots, K_P, M, A, \vec{u}, \vec{b}$;

2. *Solve*

$$K_k p_k = -\left(f'\left(\|u(z;y_k,\cdot) - u_0\|^2_{L^2(D)}\right) + g'\left(\sum_{\ell=1}^{P}\omega_\ell\|u(z;y_\ell,\cdot) - u_0\|^2_{L^2(D)}\right)\right) \times \left(M\vec{u_k} - \vec{b}\right) \quad \forall k = 1, \dots, P;$$

3. *Compute* $\widehat{J}'_{hP}(z_h) = M(\alpha\vec{z}_h - \sum_{k=1}^{P}\omega_k p_k)$.

Now, suppose to compute Hessian times a vector for the discretized problem. From here on, $v \in X_h$ denotes the projection of any Z function onto the space of piecewise linears. As such, one can write v as

$$v(x) = \sum_{k=1}^{N} v_k\phi_k(x)$$

and let $\vec{v} = (v_1, \dots, v_N)^\top$. With the state and adjoint computed, the next step in Hessian times a vector computations is solving $e_u(u(z), z)w = e_z(u(z), z)v$ for w. As with the adjoint computation, w is the solution to the following linear system

$$K\vec{w} = G$$

where $G = (\vec{G}_1^\top, \ldots, \vec{G}_P^\top)^\top$ and $\vec{G}_k$ for $k = 1, \ldots, P$ is

$$(\vec{G}_k)_i = -\int_D \sum_{j=1}^N v_j \phi_j(x)\phi_i(x)\mathrm{d}x = -\sum_{j=1}^N M_{ij} v_j$$

for all $i = 1, \ldots, N$. Therefore, $\vec{G}_k = M\vec{v}$ for $k = 1, \ldots, P$.

Next one must solve $e_u(u(z), z)q = J_{uu}(u(z), z)w$ for q. As above, this requires the solution to the linear system

$$K\vec{q} = \vec{H}$$

where

$$\begin{aligned}(\vec{H}_k)_i =& 2f''\Big(\|u_k - u_0\|^2_{L^2(D)}\Big)\langle u_k - u_0, w_k\rangle_{L^2(D)}\langle u_k - u_0, \phi_i\rangle_{L^2(D)} \\ &+ f'\Big(\|u_k - u_0\|^2_{L^2(D)}\Big)\langle w_k, \phi_i\rangle_{L^2(D)} \\ &+ 2g''\Big(\sum_{\ell=1}^P \omega_k\|u_k - u_0\|^2_{L^2(D)}\Big)\langle u_k - u_0, w_k\rangle_{L^2(D)}\langle u_k - u_0, \phi_i\rangle_{L^2(D)} \\ &+ g'\Big(\sum_{\ell=1}^P \omega_k\|u_k - u_0\|^2_{L^2(D)}\Big)\langle w_k, \phi_i\rangle_{L^2(D)}.\end{aligned}$$

Plugging in the formulas

$$u_k = \sum_{i=1}^N (\vec{u}_k)_i\phi_i \quad \text{and} \quad w_k = \sum_{i=1}^N (\vec{w}_k)_i\phi_i,$$

the inner products $\langle u_k - u_0, \phi_i\rangle_{L^2(D)}$ and $\langle w_k, \phi_i\rangle_{L^2(D)}$ become

$$\langle u_k - u_0, \phi_i\rangle_{L^2(D)} = \sum_{j=1}^N M_{ij}(\vec{u}_k)_j - b_i$$

and

$$\langle w_k, \phi_i\rangle_{L^2(D)} = \sum_{j=1}^N M_{ij}(\vec{w}_k)_j.$$

The right hand side $\vec{H}_k$ can be rewritten as

$$\begin{aligned}\vec{H}_k =&2f''\Big(\|u_k - u_0\|^2_{L^2(D)}\Big)\langle u_k - u_0, w_k\rangle_{L^2(D)}\Big(M\vec{u}k - \vec{b}\Big)\\ &+ f'\Big(\|u_k - u_0\|^2_{L^2(D)}\Big)M\vec{w}_k\\ &+ 2g''\Big(\sum_{\ell=1}^{P}\omega_k\|u_k - u_0\|^2_{L^2(D)}\Big)\langle u_k - u_0, w_k\rangle_{L^2(D)}\Big(M\vec{u}k - \vec{b}\Big)\\ &+ g'\Big(\sum_{\ell=1}^{P}\omega_k\|u_k - u_0\|^2_{L^2(D)}\Big)M\vec{w}_k \quad \forall k = 1,\dots,P.\end{aligned}$$

Finally, one computes $\widehat{J}''(z)v$ in the direction ϕ_i for $i = 1,\dots,N$ as

$$\begin{aligned}\langle \widehat{J}''(z)v, \phi_i\rangle_{Z^*,Z} \approx&\langle \widehat{J}''_{hP}(z_h)v, \phi_i\rangle_{Z^*,Z}\\ =& -\sum_{k=1}^{P}\omega_k\int_D\sum_{j=1}^{N} q_j\phi_j(x)\phi_i(x)\mathrm{d}x + \alpha\sum_{j=1}^{N}\vec{v}_j\int_D\phi_j(x)\phi_i(x)\mathrm{d}x\\ =& -\sum_{k=1}^{P}\omega_k\sum_{j=1}^{N}M_{ij}(\vec{q}_k)_j + \alpha\sum_{j=1}^{N}M_{ij}\vec{v}_j\end{aligned}$$

or equivalently

$$\widehat{J}''(z)v \approx M(\alpha\vec{v} - \sum_{k=1}^{P}\omega_k\vec{q}_k).$$

Algorithm 5.4.2 Stochastic Collocation Hessian-Vector Product

1. *Compute* $K_1,\dots,K_P, M, A$;

2. *Solve* $K_k\vec{w}_k = -M\vec{v} \quad \forall k = 1,\dots,P$;

3. *Solve*

$$\begin{aligned}K_k\vec{q}_k =&2\Big(f''\Big(\|u_k - u_0\|^2_{L^2(D)}\Big) + g''\Big(\sum_{\ell=1}^{P}\omega_k\|u_k - u_0\|^2_{L^2(D)}\Big)\Big)\\ &\times \langle u_k - u_0, w_k\rangle_{L^2(D)}\Big(M\vec{u}k - \vec{b}\Big)\\ &+ \Big(f'\Big(\|u_k - u_0\|^2_{L^2(D)}\Big) + g'\Big(\sum_{\ell=1}^{P}\omega_k\|u_k - u_0\|^2_{L^2(D)}\Big)\Big)\\ &\times M\vec{w}_k \quad \forall k = 1,\dots,P;\end{aligned}$$

4. Compute $\widehat{J}''_{hP}(z_h)v = M(\alpha\vec{v} - \sum_{k=1}^{P} \omega_k \vec{q}_k)$.

Notice that these algorithms will work for any sample based SPDE solution technique; hence, this technique will also work for the Monte Carlo finite element method.

5.4.2 Stochastic Galerkin

For general objective functions, the derivative using the stochastic Galerkin method become very tedious and rather messy to compute. This subsection will parallel the previous subsection of stochastic collocation, but I will restrict my attention to the objective function

$$J(u,z) = \frac{1}{2}E\Big[\|u(y,\cdot) - u_0\|^2_{L^2(D)}\Big] + \frac{\alpha}{2}\|z\|^2_{L^2(D)}.$$

In the same fashion as above, I will derive the discrete versions of the objective function, gradient, and Hessian times a vector calculation corresponding to the stochastic Galerkin solution technique for SPDEs. For the stochastic Galerkin method, $Y_h = \mathbb{P}^{P-1}(\Gamma)$ and X_h is any finite element space. For this research, X_h is the space of piecewise linear functions built on a given mesh $\mathcal{T}_h$. As a basis of Y_h, I choose the system of polynomials that are ρ-orthonormal. That is, $Y_h = \operatorname{span}(\Psi_1(y),\ldots,\Psi_P(y))$ where $\int_\Gamma \rho(y)\Psi_i(y)\Psi_j(y)\mathrm{d}y = \delta_{ij}$.

The discretized optimization problem in the stochastic Galerkin framework is

$$\min_{z_h\in Z_h} \widehat{J}_{hP}(z_h) \equiv \frac{1}{2}E\left[\left\|\sum_{k=1}^{P}\sum_{j=1}^{N}(\vec{u}_k(z_h))_j\ \Psi_k(y)\phi_j - u_0\right\|^2_{L^2(D)}\right] + \frac{\alpha}{2}\|z_h\|^2_{L^2(D)}$$

where

$$\vec{u}(z_h) = (u_1(z_h)^{\mathsf{T}},\ldots,u_P(z_h)^{\mathsf{T}})^{\mathsf{T}} = (u_1^{\mathsf{T}},\ldots,u_P^{\mathsf{T}})^{\mathsf{T}} = \vec{u}$$

is the stochastic Galerkin solution to the state equation

$$\underbrace{\begin{pmatrix} K_{11} & \cdots & K_{1P} \\ \vdots & \ddots & \vdots \\ K_{P1} & \cdots & K_{PP} \end{pmatrix}}_{=K} \underbrace{\begin{pmatrix} \vec{u}_1 \\ \vdots \\ \vec{u}_P \end{pmatrix}}_{=\vec{u}} = \begin{pmatrix} E[\Psi_1]M\vec{z}_h \\ \vdots \\ E[\Psi_P]M\vec{z}_h \end{pmatrix} = \begin{pmatrix} \delta_{11}M\vec{z}_h \\ \vdots \\ \delta_{1P}M\vec{z}_h \end{pmatrix}$$

since $E[\Psi_k] = E[1\Psi_k] = E[\Psi_1\Psi_k] = \delta_{1k}$. The blocks of the above K matrix have the form

$$(K_{kl})_{ij} = \int_\Gamma \rho(y)\Psi_k(y)\Psi_l(y)\int_D a(y,x)\nabla\phi_i(x)\cdot\nabla\phi_j(x)\mathrm{d}x\mathrm{d}y.$$

To compute derivatives of the objective function, one must first compute the adjoint state. In the infinite dimensional formulation, the adjoint state solves

$$\begin{aligned} -\nabla\cdot\Big(a(y,x)\nabla p(y,x)\Big) &= -\big(u(y,x)-u_0(x)\big) && \text{for } (y,x)\in\Gamma\times D \\ p(y,x) &= 0 && \text{for } (y,x)\in\Gamma\times\partial D. \end{aligned}$$

Thus, as in the stochastic Galerkin section of the numerical solutions chapter, the block FEM system is

$$K\vec{p} = F$$

where $F = (F_1^\top,\ldots,F_P^\top)^\top$ and F_k is defined as

$$\begin{aligned} (\vec{F}_k)_i &= -\int_\Gamma \rho(y)\Psi_k(y)\int_D (u(y,x)-u_0(x))\phi_i(x)\mathrm{d}x\mathrm{d}y \\ &= -\int_\Gamma \rho(y)\Psi_k(y)\int_D \sum_{l=1}^P\sum_{j=1}^N (\vec{u}_l)_j\Psi_l(y)\phi_j(x)\phi_i(x) - u_0(x)\phi_i(x)\mathrm{d}x\mathrm{d}y \\ &= -\sum_{l=1}^P\sum_{j=1}^N (\vec{u}_l)_j\left(\int_\Gamma \rho(y)\Psi_k(y)\Psi_l(y)\mathrm{d}y\right)\left(\int_D \phi_j(x)\phi_i(x)\mathrm{d}x\right) \\ &\quad+\left(\int_\Gamma \rho(y)\Psi_k(y)\mathrm{d}y\right)\left(\int_D u_0(x)\phi_i(x)\mathrm{d}x\right) \\ &= \vec{b}_i\delta_{1k} - \sum_{l=1}^P\sum_{j=1}^N M_{ij}(\vec{u}_l)_j\delta_{kl} \\ &= \vec{b}_i\delta_{1k} - \sum_{j=1}^N M_{ij}(\vec{u}_k)_j \end{aligned}$$

where $M_{ij} = \int_D \phi_i(x)\phi_j(x)\mathrm{d}x$ and $\vec{b}_i = \int_D u_0(x)\phi_i(x)\mathrm{d}x$. Therefore,

$$\vec{F}_k = \begin{cases} \vec{b} - M\vec{u}_1 & \text{if } k = 1 \\ -M\vec{u}_k & \text{if } k = 2,\ldots,P. \end{cases}$$

With the adjoint state computed, the derivative of $\widehat{J}(z)$ in the direction $\phi_j(x)$ for all $j = 1, \dots, N$ is

$$\begin{aligned}
\langle \widehat{J}'(z), w\rangle_{Z^*,Z} =& \langle e_z(u(z;\cdot,\cdot),z)p + J_z(u(z;\cdot,\cdot),z), w\rangle_{Z^*,Z} \\
=& -\int_\Gamma \rho(y) \int_D p(z;y,x)\phi_j(x)\mathrm{d}x\mathrm{d}y + \int_D z(x)\phi_j(x)\mathrm{d}x \\
\approx& -\sum_{k=1}^{P}\sum_{i=1}^{N}(p_k)_i \int_\Gamma \rho(y)\Psi_k(y)\mathrm{d}y \int_D \phi_i(x)\phi_j(x)\mathrm{d}x \\
&+\sum_{i=1}^{N} \vec{z}_i \int_D \phi_i(x)\phi_j(x)\mathrm{d}x \\
=& -\sum_{k=1}^{P}\sum_{i=1}^{N}(\vec{p}_k)_i M_{ij}\delta_{1k} + \sum_{i=1}^{N}\vec{z}_i M_{ij} \\
=& -\sum_{i=1}^{N}(\vec{p}_1)_i M_{ij} + \sum_{i=1}^{N}\vec{z}_i M_{ij}
\end{aligned}$$

Thus, the gradient is given by $\widehat{J}'(z_h) = M(\vec{z}_h - \vec{p}_1)$ and is computed via the following algorithm.

Algorithm 5.4.3 Stochastic Galerkin Gradient Computation

1. *Compute* z_h,K,M,$\vec{u}$, *and* $\vec{b}$;
2. *Compute* $\vec{F}_k = \vec{b}\delta_{1k} - M\vec{u}_k$ *for* $k = 1, \dots, P$;
3. *Solve* $Kp = F = (\vec{F}_1^\top, \dots, \vec{F}_P^\top)^\top$ *for* $k = 1, \dots, P$;
4. *Compute* $\widehat{J}'(z_h) = M(\vec{z}_h - \vec{p}_1)$.

Now, to multiply the Hessian of $\widehat{J}$ with a some vector $v \in Z$, consider $v \in X_h$ to be the projection of any Z function onto the space of piecewise linear polynomials. As such, one can write v as

$$v(x) = \sum_{k=1}^{N} v_k\phi_k(x).$$

The first step in Hessian times a vector computations is to solve $e_u(u(z),z)w = e_z(u(z),z)v$ for w. As with the adjoint computation, w is the solution to the linear system

$$K\vec{w} = G$$

where $\vec{G} = (\vec{G}_1^\top, \dots, \vec{G}_P^\top)^\top$ and $\vec{G}_k$ for $k = 1, \dots, P$ is

$$(\vec{G}_k)_i = \int_\Gamma \rho(y)\Psi_k(y)\mathrm{d}y \int_D \sum_{j=1}^N v_j\phi_j(x)\phi_i(x)\mathrm{d}x = \sum_{j=1}^N M_{ij}v_j\delta_{1k}$$

for all $i = 1, \dots, N$. Therefore,

$$\vec{G}_k = \begin{cases} M\vec{v} & \text{if } k = 1 \\ 0 & \text{if } k = 2, \dots, P. \end{cases}$$

Next, one must solve $e_u(u(z),z)q = J_{uu}(u(z),z)w$ for q. As above, this requires the solution to the linear system $K\vec{q} = H$ where

$$(\vec{H}_k)_i = \sum_{l=1}^P \sum_{j=1}^N (\vec{w}_l)_j \int_\Gamma \rho(y)\Psi_k(y)\Psi_l(y)\mathrm{d}y \int_D \phi_j(x)\phi_i(x)\mathrm{d}x = \sum_{j=1}^N M_{ij}(\vec{w}_k)_j$$

or equivalently, $\vec{H}_k = M\vec{w}_k$ for all $k = 1, \dots, P$. Finally, $\widehat{J}''(z)v$ in the direction ϕ_i for $i = 1, \dots, N$ is approximated by

$$\begin{aligned}
\langle \widehat{J}''(z)v, \phi_i\rangle_{Z^*,Z} \approx& \langle \widehat{J}''_{hP}(z_h)v, \phi_i\rangle_{Z^*,Z} \\
=& -\int_\Gamma \rho(y) \int_D \sum_{k=1}^P \sum_{j=1}^N (\vec{q}_k)_j\phi_j(x)\Psi_k(y)\phi_i(x)\mathrm{d}x\mathrm{d}y + \alpha \int_D v(x)\phi_i(x)\mathrm{d}x \\
=& -\sum_{k=1}^P \sum_{j=1}^N M_{ij}(\vec{q}_k)_j\delta_{1k} + \alpha \sum_{j=1}^N \vec{v}_j M_{ij} \\
=& \sum_{j=1}^N M_{ij}(\alpha\vec{v}_j - (\vec{q}_1)_j)
\end{aligned}$$

or equivalently $\widehat{J}''(z)v \approx M(\alpha\vec{v} - \vec{q}_1)$. The following algorithm describes this procedure.

Algorithm 5.4.4 Stochastic Galerkin Hessian-Vector Product

1. *Compute* K, M, A;

2. *Compute* $\vec{G}_1 = M\vec{v}$ *and* $\vec{G}_k = 0$ *for* $k = 2, \ldots, P$;

3. *Solve* $K\vec{w} = G = (\vec{G}_1^\top, \ldots, \vec{G}_P^\top)^\top$;

4. *Compute* $\vec{H}_k = M\vec{w}_k$ *for* $k = 1, \ldots, P$;

5. *Solve* $K\vec{q} = H = (\vec{H}_1^\top, \ldots, \vec{H}_P^\top)^\top$;

6. *Compute* $\widehat{J}''_{hP}(z_h)v = M(\alpha\vec{v} - \vec{q}_1)$.

5.5 Stochastic Collocation and the Inexact Newton's Method

In this section, I will develop a nearly parallel algorithm for solving optimization problems governed by SPDEs using collocation techniques. In the previous sections, I developed gradient and Hessian times vector computations for the objective function $\widehat{J}(z) = J(u(z;\cdot,\cdot), z)$ where $u(z;\cdot,\cdot)$ is the implicit solution to $e(u,z) = 0$. Under the collocation framework, some parallelizations may be made to these computations. Also, Hessian approximations may be used to achieve a form of inexact Newton's method with inexact Hessian information. Two key ideas are central to the results in the section:

- Since the solution to the state equation $e(u,z)$ depends on a finite number of random variables, the solution to the adjoint equation also depends on a finite number of random variables and these variables are equal to those of the state equation;

- Newton's method converges q-linearly with approximate Hessian information H, as long as $\|H^{-1}(H - \widehat{J}''(z))\| < \zeta \in (0,1)$.

To solidify the first observation, recall that the Doobs-Dynkins lemma shows that if the state equation depends on a finite number of random variables, $e(u, z; y)$ for $y \in \Gamma$, then the solution to the state equation also depends on y. Now, the adjoint equation is given by

$$e_u(u(z; \cdot, \cdot), z; y)^* p = -J_u(u(z; \cdot, \cdot), z).$$

Clearly $e_u(u, z; y)^*$ depends on y and $J_u(u(z; \cdot, \cdot), z)$ depends on y implicitly through its dependence on u. Therefore, the Doobs-Dynkins lemma also implies that $p(z; \cdot, \cdot)$ depends on y. The second observation is solidified in lemma 5.1.7.

First, I will present the parallelized computations to solve the state and adjoint equations. With these computations, one can compute the objective function and its gradient in one shot. Consider the optimization problem

$$\min_{z \in Z} \widehat{J}(z) = J(u(z; \cdot, \cdot), z)$$

where $u(z; \cdot, \cdot)$ is the implicit solution to $e(u(z; \cdot, \cdot), z) = 0$. Suppose the dimension of the collocation space is P_1. That is, there are P_1 collocation points $\{y_k\}_{k=1}^{P_1}$ with nodes $\{\omega_k\}_{k=1}^{P_1}$ and the basis functions are the Lagrange polynomials built on the collocation points. Under the stochastic collocation framework, the state equation decouples into P_1 deterministic equations, $e(u, z; y_k) = 0$ for $k = 1, \ldots, P_1$. Furthermore, consider objective functions of the form

$$J(u, z) = \frac{1}{2} E\Big[f\Big(\|u(y, \cdot) - u_0\|_O^2\Big)\Big] + \frac{1}{2} g\Big(E\Big[\|u(y, \cdot) - u_0\|_O^2\Big]\Big) + \frac{\alpha}{2}\|z\|_{L^2(D)}^2$$

where $u_0 \in H^1(D)$ is some desired distribution and

$$f : [0, \infty) \to [0, \infty),$$

$$g : [0, \infty) \to [0, \infty),$$

are twice continuously differentiable functions. Then, in the collocation framework, the objective function is rewritten as

$$J_{hP_1}(u, z) = \frac{1}{2} \sum_{k=1}^{P_1} \omega_k f\Big(\|u(y_k, \cdot) - u_0\|_O^2\Big) + \frac{1}{2} g\Big(\sum_{k=1}^{P_1} \omega_k \|u(y_k, \cdot) - u_0\|_O^2\Big) + \frac{\alpha}{2}\|z\|_{L^2(D)}^2.$$

Let $e_k(u,z) = e(u,z;y_k)$ be the state equation evaluated at the k collocation point y_k. Once the state, u_k for $k = 1,\dots,P_1$, is computed, the objective function is readily available and, furthermore, the adjoint equation becomes

$$\begin{aligned}
&\langle (e_k)_u(u_{hP}(z;y_k,\cdot),z)^* p_k, v\rangle_{Z^*,Z}\\
&\quad = - f'\Big(\|u(z;y_k,\cdot)-u_0\|_O^2\Big)\langle u(z;y_k,\cdot)-u_0, v\rangle_O\\
&\quad\quad - g'\Big(\sum_{\ell=1}^{P_1}\omega_\ell\|u(z;y_\ell,\cdot)-u_0\|_O^2\Big)\langle u(z;y_k,\cdot)-u_0, v\rangle_O.
\end{aligned}$$

This leads to the following algorithm for computing state, objective function, and adjoint state in one shot.

Algorithm 5.5.1 (Parallel State/Objective Function/Adjoint Computation)

```
1. Given $z_h$, $\{y_k\}_{k=1}^{P_1}$ and $\{\omega_k\}_{k=1}^{P_1}$. Set $F = 0$ and $G = 0$.
2. for $k = 1,\dots,P_1$
    (a) Compute $u_k(\cdot) = u_k(z_h;\cdot)$ which solves the $k^{th}$ state equation,
            $e_k(u, z_h) = 0;$
    (b) Set $F \leftarrow F + \omega_k f\Big(\|u(y_k,\cdot)-u_0\|_O^2\Big)$;
    (c) Compute $\bar{p}_k(\cdot) = \bar{p}_k(z_h;\cdot)$ which solves
            $\langle (e_k)_u(u_k(z;\cdot),z_h)^*\bar{p}_k, v\rangle_{Z^*,Z} = -\langle u(z_h;y_k,\cdot)-u_0, v\rangle_O \quad \forall v \in U;$
    (d) Set $\hat{p}_k = f'\Big(\|u(y_k,\cdot)-u_0\|_O^2\Big)\bar{p}_k$;
    (e) Set $G \leftarrow G + \omega_k\|u(y_k,\cdot)-u_0\|_O^2$;
3. end
4. Compute $\widehat{J}_{hP_1}(z_h) = \frac{1}{2}(F + g(G) + \alpha\|z_h\|_{L^2(D)}^2)$;
```

5. *Compute the adjoint as* $p = \{\hat{p}_k + g'(G)\bar{p}_k\}_{k=1}^{P_1}$.

In the study of linear SPDE constrained optimization, this is a vast improvement to the originally stated algorithm. Solving the state equation alone requires the solution of P_1 linear system solves where P_1 is the number of collocation points. The manner in which the original algorithm is written requires the computation and storage of P_1 stiffness matrices at each optimization iteration, or the computation of $2P_1$ stiffness matrices at each optimization iteration if one recomputes the stiffness matrices for the state and adjoint computations. This could not be possible due to storage limitations or limitations on the speed of computating each stiffness matrix. The above algorithm requires the computation of P_1 stiffness matrices, but only stores one stiffness matrix at each iteration. One observation should be noted: if P_1 is "small" then it may be more efficient to store the P_1 stiffness matrices instead of computing them on the fly. If the matrices are stored, then they are readily available for the Hessian times a vector computations as well. In the scheme I have just described, the stiffness matrices need to be recomputed.

The same parallelism is natually available in the Hessian times a vector computation. The two equations (other than the state and adjoint equations) that must be solved are

$$e_u(u(z;\cdot,\cdot),z)w = e_z(u(z;\cdot,\cdot),z)v$$

where v is the vector to be multiplied and

$$e_u(u(z;\cdot,\cdot),z)^*q = L_{uu}(u(z;\cdot,\cdot),z,p(z;\cdot,\cdot))w - L_{uz}(u(z;\cdot,\cdot),z,p(z;\cdot,\cdot))v.$$

Defining the stochastic collocation Laplacian as

$$L_k(u,z,p) = j_k(u,z) + \langle p_k, e_k(u,z)\rangle_{U^*,U}$$

then the stochastic collocation framework requires

$$(e_k)_u(u_k(z),z)w_k = (e_k)_z(u_k(z),z)v$$

and

$$(e_k)_u(u_k(z), z)^* q_k = (L_k)_{uu}(u_k(z), z, p_k(z))w_k - (L_k)_{uz}(u_k(z), z, p_k(z))v.$$

This formulation gives rise to the analogous stochastic collocation Hessian times a vector computation:

Algorithm 5.5.2 Parallel Hessian Times a Vector Vector, Steps 3 and 4

```
1. Given $v$,$G$,$\{u_k\}_{k=1}^{P_1}$, $\{p_k\}_{k=1}^{P_1}$, $\{y_k\}_{k=1}^{P_1}$ and $\{\omega_k\}_{k=1}^{P_1}$;
2. for $k = 1, \dots, P_1$
    (a) Compute $w_k(\cdot) = w_k(z_h, \cdot)$ which solves
            $(e_k)_u(u_k(z_h; \cdot, \cdot), z_h)w_k = (e_k)_z(u_k(z_h; \cdot, \cdot), z_h)v;$
    (b) Compute $\bar{q}_k^1(\cdot) = \bar{q}_k^2(z; \cdot)$ which solves
            $\langle (e_k)_u(u_k(z), z)^* q_k, v\rangle_{Z^*,Z} = \langle u_k - u_0, v\rangle_O \quad \forall v \in U$
    (c) Compute $\bar{q}_k^2(\cdot) = \bar{q}_k^1(z; \cdot)$ which solves
            $\langle (e_k)_u(u_k(z), z)^* q_k, v\rangle_{Z^*,Z} = \langle w_k, v\rangle_O \quad \forall v \in U$
    (d) Set $\hat{q}_k^1 = 2f''\Big(\|u_k - u_0\|_O^2\Big)\langle u_k - u_0, w_k\rangle_O \bar{q}_k^1;$
    (e) Set $\hat{q}_k^2 = f'\Big(\|u_k - u_0\|_O^2\Big)\bar{q}_k^2;$
    (f) Compute $q_k = \hat{q}_k^1 + \hat{q}_k^2 + 2g''(G)\bar{q}_k^1 + g'(G)\bar{q}_k^2.$
3. end
```

The fact that the above algorithm does not store stiffness matrices brings me to the next topic. From here on, I will assume P_1 is too large to store P_1 stiffness matrices. Note that this does not take much. If $\Gamma \subset \mathbb{R}^{10}$, a first order tensor product quadrature rule will have $P_1 = 2^{10}$ nodes! As stated, the above algorithm does not store stiffness matrices and hence they must be recomputed for the Hessian vector multiplication routine. Since P_1 is large, stiffness matrix computation is a costly process. One way to circumvent this cost is to choose a lower order quadrature rule, requiring $P_2 \ll P_1$ nodes. This will result in an approximate Hessian times a vector computation (taking the Hessian computed with P_2 nodes as the exact Hessian). Note that for quadratic control problems governed by linear SPDEs, this method may be applied as is because the first order derivatives and second order derivatives of e do not depend on u_k and the second order derivatives of J also do not depend on u_k or p_k. In the general setting, one may need to interpolate the state and adjoint in order to evaluate them at the new quadrature points.

The approximation of the Hessian vector products using lower order quadrature rules leads to a version of Newton's method with inexact derivative information. As lemma (5.1.7) pointed out, as long as

$$\|\widehat{J}''_{hP_2}(z)^{-1}(\widehat{J}''_{hP_2}(z) - \widehat{J}''_{hP_1}(z))\|_{Z^*} = \|I - \widehat{J}''_{hP_2}(z)^{-1}\widehat{J}''_{hP_1}(z)\|_{Z^*} < \eta \in (0,1),$$

inexact Newton's method will converge q-linearly. This is a slow down in the convergence rate of Newton's method (from q-quadratic to q-linear), but may result in a more computationally tractable method since each iteration is computationally cheaper.

5.6 Numerical Example

This example illustrates the quality of the algorithms I just presented and demonstrates the need for such routines. When faced with an SPDE constrained optimization problem, one might ask: why can't I take the expected value of each of the uncertain coefficient functions and solve the resulting deterministic PDE? This example is meant to clarify this question by presenting a quadratic control problem governed by a stochastic advection-diffusion equation where the solution to the stochastic problem does not agree with the solution to the mean value problem. The stochastic advection-diffusion equation is of the form

$$\begin{aligned} -\nabla \cdot (a(y,x)\nabla u(y,x)) + c(y,x) \cdot \nabla u(y,x) &= z(x) && \text{for } (y,x) \in \Gamma \times D \quad (5.6.1)\\ u(y,x) &= d(y,x) && \text{for } (y,x) \in \Gamma \times \partial D_D\\ \frac{\partial u}{\partial n}(y,x) &= 0 && \text{for } (y,x) \in \Gamma \times \partial D_N. \end{aligned}$$

The corresponding mean value problem is the deterministic problem resulting from taking the expected value of each random coefficient is given by

$$\begin{aligned} -\nabla \cdot (E[a](x)\nabla u(x)) + E[c](x) \cdot \nabla u(x) &= z(x) && \text{for } x \in D \quad (5.6.2)\\ u(x) &= E[d](x) && \text{for } x \in \partial D_D\\ \frac{\partial u}{\partial n}(x) &= 0 && \text{for } x \in \partial D_N. \end{aligned}$$

In the following examples the domain is the unit square $D = (0,1)^2$. The Neumann boundary is $\partial D_N = \{1\} \times (0,1)$ and $\partial D_D = \partial D \setminus \partial D_N$. See Figure 5.1 for a sketch of the domain and the boundary conditions.

For this example, I chose the following random fields as the coefficient functions: The Dirichlet boundary conditions are

$$d(y,x) = 0 \text{ for } x \in (0,1) \times \{0,1\},$$

and on $\{x \in \partial D_D \ : \ x \in \{0\} \times (0,1)\}$

$$d(y,x) = \begin{cases} 0 & \text{if } x_2 \notin (0.25, 0.75),\\ \sin(2\pi(x_2 - 0.25)) & \text{if } x_2 \in (0.25,0.75) \text{ and } y_1 = 0,\\ \sin\left(\pi\dfrac{\exp(4y_1(x_2-0.25)) - 1}{\exp(2y_1) - 1}\right) & \text{if } x_2 \in (0.25,0.75) \text{ and } y_1 \neq 0. \end{cases}$$

The advection coefficients are

$$c(y,x) = \frac{1}{2}(y_3+1)\begin{pmatrix} \cos\left(\frac{\pi}{4}y_2\right)\\ \sin\left(\frac{\pi}{4}y_2\right) \end{pmatrix}.$$

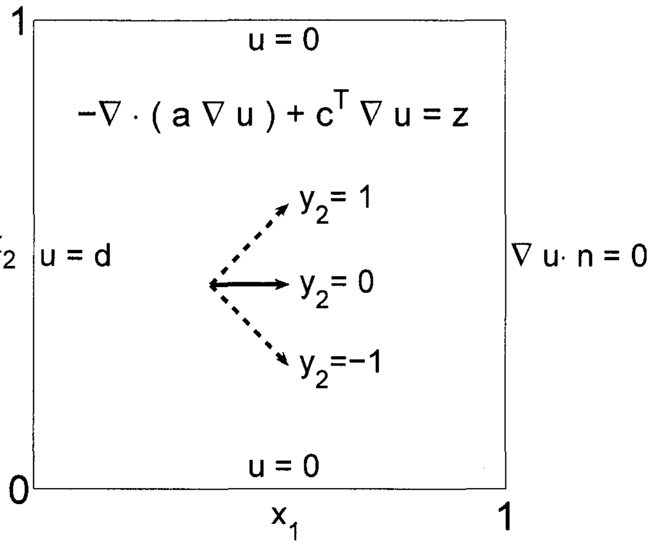


Figure 5.1: The domain and boundary conditions used for the numerical experiments

The diffusivity coefficient is

$$a(y,x) = 10^{-2}\Big(1 + y_4\frac{x_1}{2}\Big).$$

Each random variable y_i is independently and identically distributed according to a uniform distribution on $[-1, 1]$; therefore,

$$\Gamma = [-1, 1]^4$$

and the Lebesgue density is

$$\rho(y) = 2^{-4}\chi_\Gamma(y).$$

Taking expected values of the random fields gives

$$\begin{aligned} E[a](x) &= 10^{-2} \\ E[c](x) &= \frac{2\sqrt{2}}{\pi}\begin{pmatrix}1\\0\end{pmatrix} \\ E[d](x) &= \begin{cases} 0 & \text{for } x \in \partial D_D \setminus \big(\{0\} \times (0.25, 0.75)\big) \\ \sin(2\pi(x_2 - 0.25)) & \text{for } x \in \{0\} \times (0.25, 0.75). \end{cases} \end{aligned}$$

Therefore, the mean value problem is

$$-10^{-2}\Delta u(x) + \frac{2\sqrt{2}}{\pi}\frac{\partial}{\partial x_1}u(x) = z(x) \qquad \text{for } x \in D$$

$$\begin{aligned} u(x) &= 0 && \text{for } x \in \partial D_D \setminus \big(\{0\} \times (0.25, 0.75)\big) \\ u(x) &= \sin(2\pi(x_2 - 0.25)) && \text{for } x \in \{0\} \times (0.25, 0.75) \\ \frac{\partial u}{\partial n}(x) &= 0 && \text{for } x \in \partial D_N. \end{aligned}$$

For the remainder of this section I use a fixed spatial finite element discretization. The spatial domain D is subdivided into $1/32 \times 1/32$ squares, which are then each subdivided into two triangles. This leads to a uniform mesh of triangles with a total of $N = 1089$ vertices. For the finite element basis, I use piecewise linear polynomial basis functions. Furthermore, when Smolyak quadrature is applied, the Smolyak rules are built on one dimensional Clenshaw-Curtis quadrature rules. The Smolyak formula uses the subsequence of 1D rules corresponding to one dimensional polynomial exactness $m_i = 2^{i-1} + 1$ in order to build the multidimensional sparse grid. This subsequence gives an embedding of the 1D Clenshaw-Curtis quadrature nodes and hence an embedding of the Smolyak quadrature nodes (see figure 4.1). Furthermore, [29] provides a proof showing that Smolyak of level $M + \ell$ (i.e. $\mathcal{A}(M + \ell, M)$) built on such nodes is exact for all polynomials of total degree ℓ.

5.6.1 Properties of the Hessian

In this section, I will consider two optimal control problems governed by the SPDE described in the above introduction. The first control problems is

$$\min_{z \in Z} \widehat{J}(z) \equiv \frac{1}{2}E\Big[\|u(z;\cdot,\cdot) - u_0\|^2_{L^2(D)}\Big] + \frac{\alpha}{2}\|z\|^2_{L^2(D)}$$

where $u(z;\cdot,\cdot) = u(\cdot,\cdot)$ is the weak solution to

$$\begin{aligned}
-\nabla \cdot (a(y,x)\nabla u(y,x)) + c(y,x) \cdot \nabla u(y,x) &= z(x) && \text{for } (y,x) \in \Gamma \times D \\
u(y,x) &= d(y,x) && \text{for } (y,x) \in \Gamma \times \partial D_D \\
\frac{\partial u}{\partial n}(y,x) &= 0 && \text{for } (y,x) \in \Gamma \times \partial D_N.
\end{aligned}$$

The second control problem is

$$\min_{z\in Z} \widehat{J}(z) \equiv \frac{1}{2}\mathrm{Var}\Big[\|u(z;\cdot,\cdot) - u_0\|^2_{L^2(D)}\Big] + \frac{\alpha}{2}\|z\|^2_{L^2(D)}$$

again where $u(z;\cdot,\cdot)$ weakly solves the above SPDE. All optimization problems are discretized using stochastic collocation finite elements. For more details concerning the discretization, see the following subsection on optimization results. For the following computations, $\alpha = 10^{-4}$.

In order for any optimization problem to have a solution, there must exist a point at which the gradient of $\widehat{J}(z)$ is zero and the Hessian of $\widehat{J}(z)$ is positive semidefinite. For both expected value and variance example optimization problems, the discretized Hessian, $\widehat{J}''_{hP}(z)$, is associated with the matrix

$$A(z) + \alpha M$$

where $A(z)$ is some matrix that may depend on the control z and M is the standard FEM mass matrix. If $(\lambda, v) \in \mathbb{R} \times Z$ is an eigenpair of $\widehat{J}''_{hP}(z)$, then

$$\langle \widehat{J}''_{hP}(z)v, v\rangle_{Z^*,Z} = \lambda\langle v, v\rangle_Z.$$

Discretely, the $Z = L^2(D)$ norm is represented as the quadratic form

$$\langle v, v\rangle_Z \approx \vec{v}^{\top} M\vec{v}$$

where $\vec{v}$ is the vector containing the coefficients of the piecewise linear expansion of v. In the above matrix notation, this eigenvalue problem becomes: find $(\lambda, \vec{v}) \in \mathbb{R} \times \mathbb{R}^n$ such that

$$(A(z) + \alpha M)\vec{v} = \lambda M\vec{v}.$$

Therefore, the pair $(\eta := \lambda - \alpha, \vec{v}) \in \mathbb{R} \times \mathbb{R}^n$ is a generalized eigenpair for the matrix pair $(A(z), M)$. In order to compute the eigenvalues of the Hessian, one can solve the generalized eigenvalue problem: find $(\eta, \vec{v}) \in \mathbb{R} \times \mathbb{R}^n$ such that

$$A(z)\vec{v} = \eta M \vec{v}$$

and shift the resulting generalized eigenvalues by a factor of α. For the Hessian to be positive semidefinite, the generalized eigenvalues of the pair $(A(z), M)$ must all be greater than or equal to α. For the problem of minimizing the expected value, the objective function is quadratic and, hence, the mapping $z \mapsto \widehat{J}''_{hP}(z)$ is constant. Figure 5.2 depicts the generalized eigenvalues for $(A(z), M)$ computed using multiple tensor product, Smolyak, and Stroud rules. Notice that all eigenvalues are positive or zero and, thus, the optimization problem has a solution. In the problem of minimizing the variance, the mapping $z \mapsto \widehat{J}''_{hP}(z)$ is not constant. Since I am interested in the eigenvalues at a solution z^*, I first solve the optimal control problem discretized in the stochastic domain Γ with Smolyak quadrature of total degree 4 for z^*, then computing the generalized eigenvalues of the pair $(A(z^*), M)$. As seen in figures 5.3, 5.4, and 5.5, the matrix pair $(A(z^*), M)$ has some negative eigenvalues. Without a large enough regularization parameter, α, this would be a problem. Notice that all of the negative eigenvalues have magnitude less than or equal to 10^{-4}, therefore, all of the eigenvalues of the Hessian are greater than zero. Figures 5.3, 5.4, and 5.5 also show that varying degrees of quadrature could give vastly different Hessians.

Table 5.1 contains the number of positive, negative, and zero eigenvalues corresponding to the different quadrature rules. These numbers reinforce the fact that low order quadrature rules, specifically the Stroud rules, may not be sufficient to solve these optimization problems governed by SPDEs. The Stroud rules may perform particularly poorly because they require very few quadrature nodes. This sparsity of nodes leads to a vary sparse sampling of the stochastic space Γ. Also, table 5.1 shows that the Hessian for each quadrature rule has 97 zero eigenvalues. These eigenvalues correspond to the 97 vertices in the mesh that are constrained (i.e. have Dirichlet

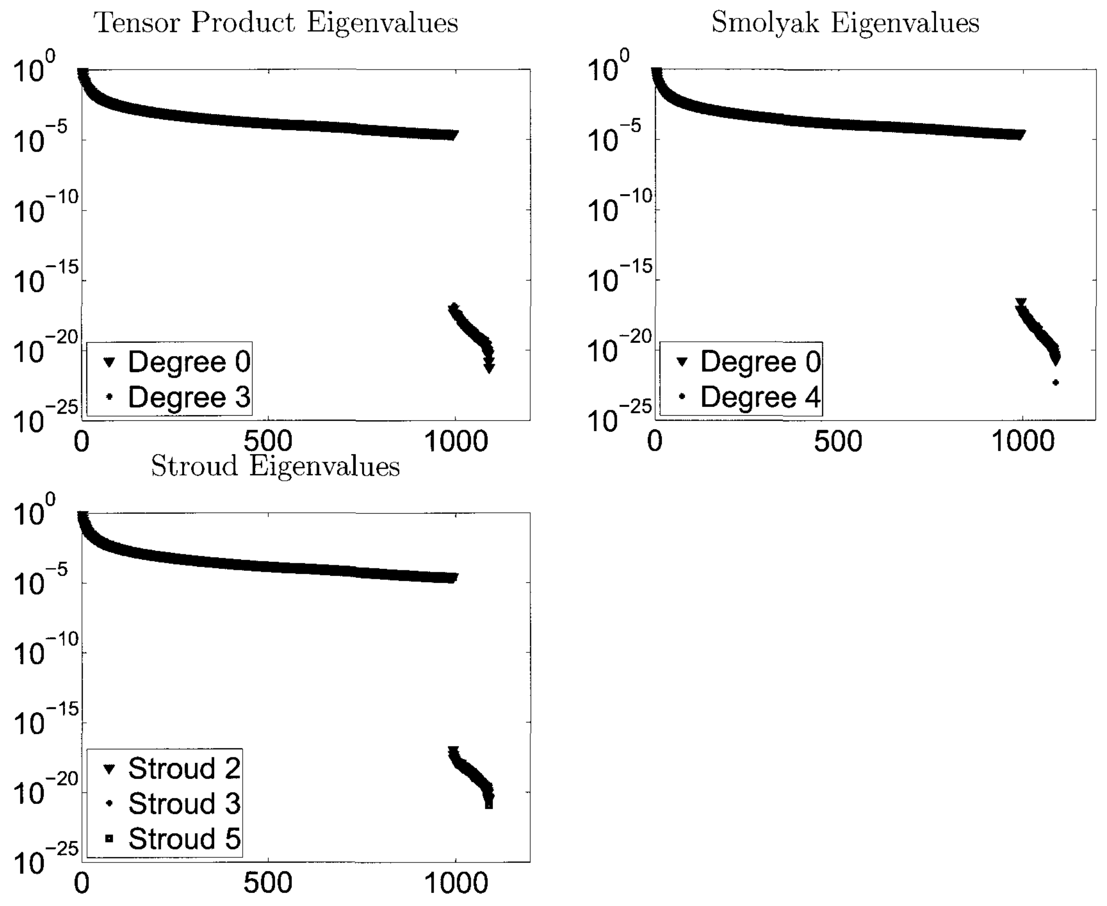


Figure 5.2: Eigenvalues of the Hessian of the expected value, discretized using tensor product (top left), Smolyak (top right), and Stroud (bottom left) quadrature rules.

boundary conditions).

5.6.2 Optimization Results

In this section, the objective function is the quadratic, given by

$$J(u,z) = \frac{1}{2}\|u-u_0\|^2_{L^2_\rho(\Gamma;L^2(D))} + \frac{\alpha}{2}\|z\|^2_{L^2(D)} = \frac{1}{2}E\Big[\|u(y,\cdot)-u_0\|^2_{L^2(D)}\Big] + \frac{\alpha}{2}\|z\|^2_{L^2(D)}$$

where the desired distribution is $u_0 = 1$ and the regularization parameter is $\alpha = 10^{-4}$.

First I solve the deterministic problem, that is $u(z;\cdot) = u(\cdot)$ is the solution of (5.6.2). The optimization problem is solved using Newton-CG. The solutions to this

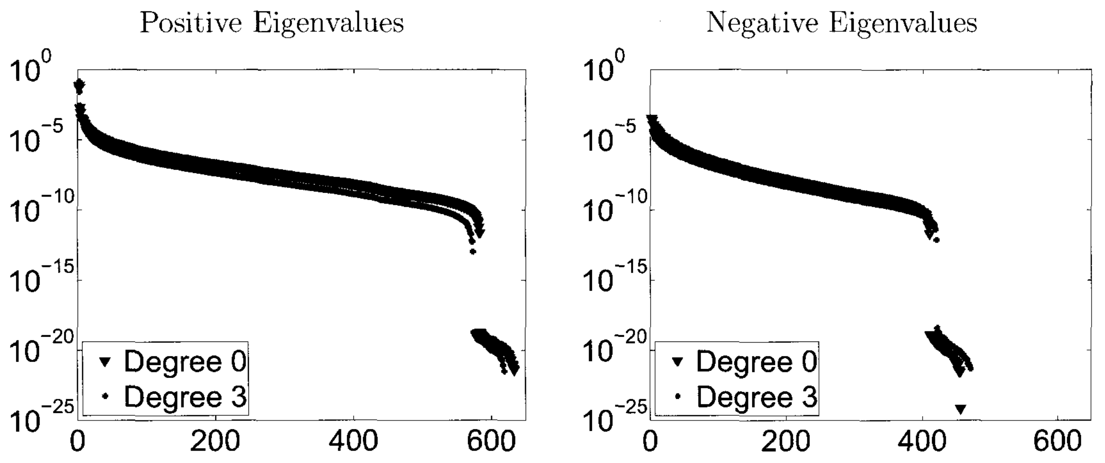


Figure 5.3: Eigenvalues of the Hessian of the variance, discretized using tensor product quadrature.

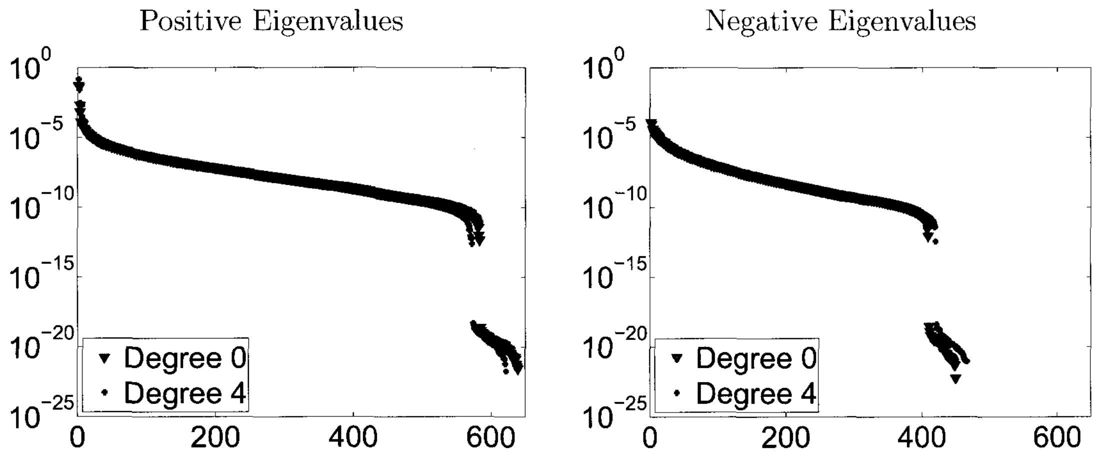


Figure 5.4: Eigenvalues of the Hessian of the variance, discretized using Smolyak quadrature.

deterministic problem are illustrated in figure 5.6.2.

To solve the stochastic problem, I use the stochastic collocation framework described in the previous section. Using the spatial discretization scheme described above, the resulting stiffness matrices are $K_k \in \mathbb{R}^{1089 \times 1089}$. I use a warm started Newton-CG framework in which I begin with Smolyak nodes built on 1D Clenshaw-Curtis quadrature nodes. The initial quadrature nodes are exact for polynomials of total degree 1. I solve the optimal control problem on these nodes starting with an

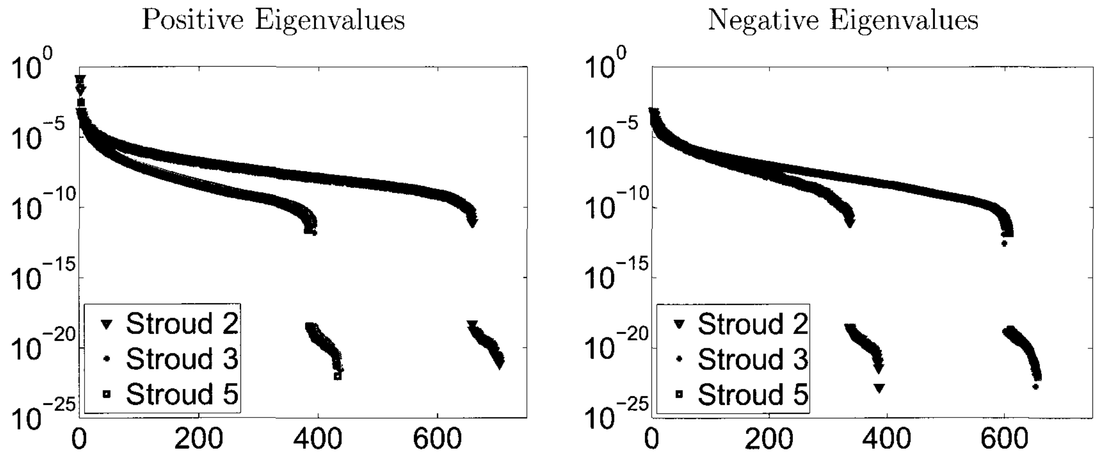
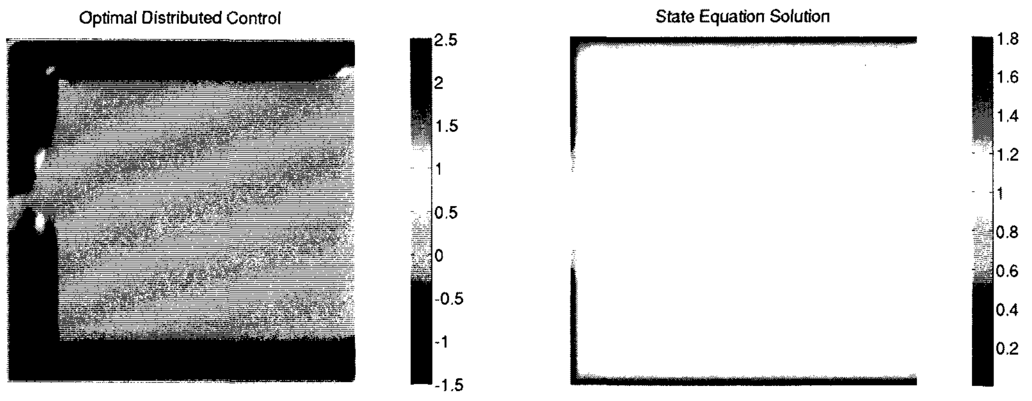


Figure 5.5: Eigenvalues of the Hessian of the variance, discretized using Stroud quadrature.



initial guess of $z = 0$. Once the degree 1 problem converges, I construct Smolyak nodes, again built on 1D Clenshaw-Curtis quadrature nodes, which integrate polynomials of total degree 2 exactly. The initial guess for the degree 2 Smolyak rule is the optimal Smolyak degree 1 solution. Finally, once the degree 2 problem has converged, I build Smolyak nodes, again built on 1D Clenshaw-Curtis quadrature nodes, which integrate polynomials of total degree 3 exactly and solve the optimal control problem with initial guess as the optimal Smolyak degree 2 solution. The Smolyak degree 1 formula requires $P = 401$ quadrature nodes, Smolyak degree 2 requires $P = 1105$ quadrature nodes, and Smolyak degree 3 requires $P = 2929$ quadrature nodes. I use

Quadrature Rule	Degree	Positive	Negative	Zero
Tensor Product	0	582	507	97
	1	503	586	97
	2	560	529	97
	3	572	517	97
Smolyak	0	583	506	97
	1	560	529	97
	2	547	542	97
	3	566	523	97
	4	572	517	97
Stroud	2	657	432	97
	3	393	696	97
	5	384	705	97

Table 5.1: The table contains the total number of positive, negative, and zero eigenvalues for the Hessian of the variance when tensor product, Smolyak, and Stroud quadrature rules of varying degrees are used. Note that the Smolyak degree refers to the total degree of polynomial exactness, while the tensor product and Stroud degrees refer to the degree of polynomial exactness in every direction.

the same collocation points for the gradient and Hessian computations. To solve each optimization problem, I used an inexact Newton method with backtracking linesearch. I solve the Newton system using the conjugate gradient method. The convergence history for this problem is displayed in Table 5.6.2. Note that since the objective function is quadratic, Newton's method should converge in one iteration. Also, due to the quadratic nature, one could solve this problem directly using the conjugate gradient method. I have developed a framework centered around Newton's method, which is general enough to handle any objective function. Thus, for this example, I apply the Newton's framework described in this chapter.

Smolyak	Iteration	$\hat{J}(z)$	$\lVert\nabla\hat{J}(z)\rVert_2$	$\lVert s\rVert_2$	Step Size	# CG Iterations
1	0	6.428522e-01	6.943939e-02	1.440176e+02	1.000000e+00	15
	1	6.639377e-02	6.254207e-04	3.130008e+02	1.000000e+00	691
	2	4.587229e-02	3.860550e-07			
2	0	4.540477e-02	1.658924e-04	1.091528e+01	1.000000e+00	737
	1	4.536547e-02	2.323872e-08			
3	0	4.543999e-02	9.652338e-05	1.819926e+00	1.000000e+00	357
	1	4.543617e-02	8.943427e-09			

Table 5.2: This table displays the convergence history of the warm started Newton-CG method for the expected value control problem. s denotes the Newton step and step size refers to the linesearch step size.

It takes the warm started Newton-CG method two iterations for Smolyak degree 1 and one iterate for Smolyak degree 2 and 3. Figure 5.6 illustrates the optimal distributed control to the stochastic problem. The left panel of figure 5.7 depicts

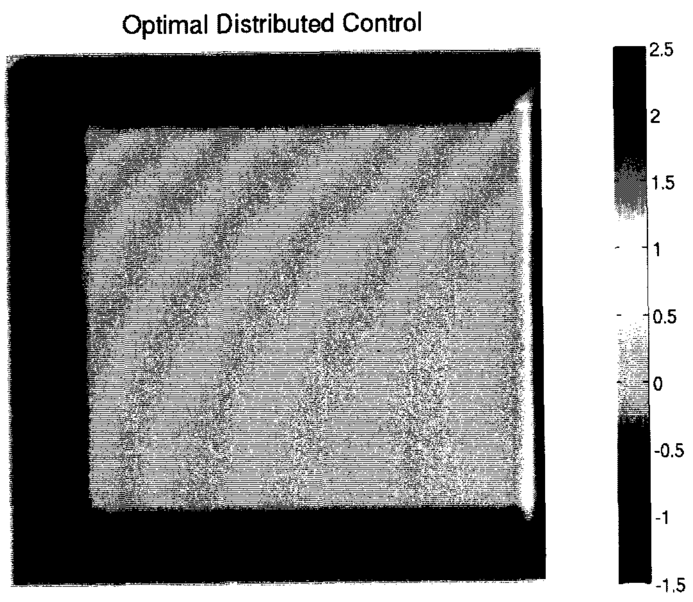


Figure 5.6: Optimal distributed control for the expected value control problem.

the expected value of the solution to the SPDE state equation corresponding to the computed optimal distributed control. The right panel of figure 5.7 depicts the

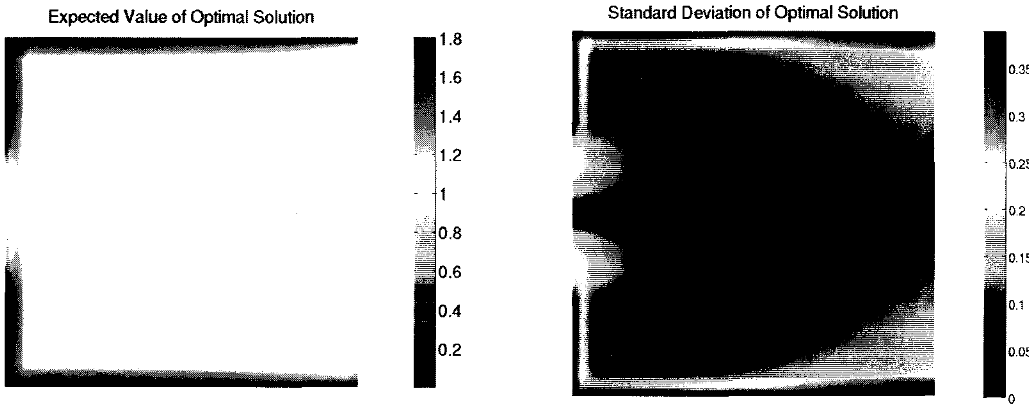


Figure 5.7: The left figure depicts the expected value of the state equation solution corresponding to the optimal distributed control. The right figure depicts the standard deviation of the solution to the state equation.

standard deviation of the solution to the stochastic state equation corresponding to the computed optimal distributed control. As one would expect, the variance is high near the Dirichlet boundary conditions.

Figures 5.8 and 5.9 display, side by side, the optimal controls and states computed for the deterministic (5.6.2)(with random field coefficients replaced by their expected values) and the stochastic problems (5.6.1). Upon comparing the controls, one notices that it is unnecessary to control the deterministic problem near the nonzero Dirichlet conditions and the Neumann conditions. This difference is due to the fact that both the advection and the Dirichlet conditions are fixed in the deterministic problem. Although the controls do vary, the solution to the state equation for the deterministic problem and the expected value of the solution to the state equation for the stochastic problem are similar. With this said, much information in the control is lost when substituting the deterministic problem for the stochastic problem.

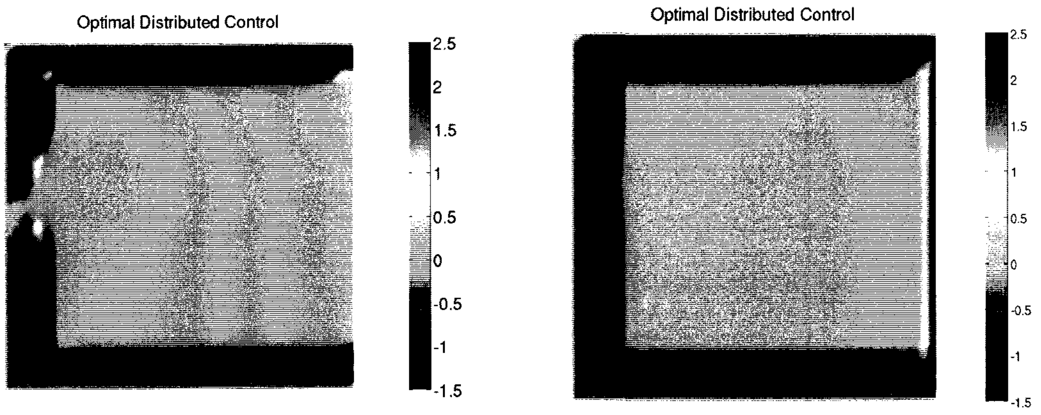


Figure 5.8: The left figure depicts the optimal control corresponding to the deterministic problem, with random field coefficients replaced by their expected values. The right figure depicts the optimal control corresponding to the stochastic problem.

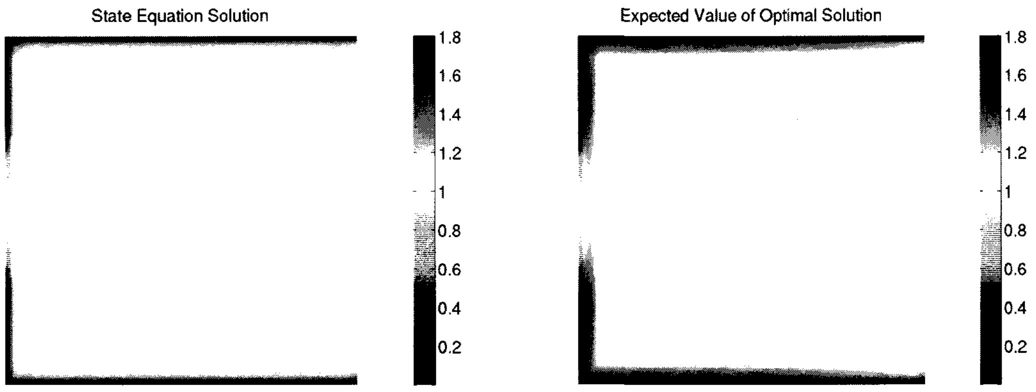


Figure 5.9: The left figure depicts the solution to the state equation corresponding to the deterministic control problem, with random field coefficients replaced by their expected values. The left figure depicts the expected value of the state equation solution corresponding to the stochastic problem.

5.6.2.1 Newton with Inexact Hessian Information

As seen in lemma 5.1.7, in order for Newton's method with inexact Hessian information to converge q-linearly, there must exists a $\zeta \in (0,1)$ such that the error term

$$\|H^{-1}(H - \widehat{J}''(z))\| < \zeta$$

Quadrature Rule	$\|H^{-1}(H - \widehat{J}''(z))\|_2$
1 Point	21.7061
Smolyak 0	7.1017
Smolyak 1	2.6986
Smolyak 2	**0.6038**
Stroud 2	1.4022
Stroud 3	2.2973
Stroud 5	**0.5663**

Table 5.3: In order for Newton's method with inexact Hessian information to converge q-linearly, the error $\|H^{-1}(H - \widehat{J}''(z))\|_2$ must be less than 1. From this table, approximating the Hessian using degree 2 Smolyak and degree 5 Stroud will yield q-linear convergence.

where H is an approximation to $\widehat{J}''(z)$. For the following results, the exact Hessian is taken to be the degree 3 Smolyak approximation using stochastic collocation FEM. Here the degree 3 Smolyak rule is built on one dimensional Clenshaw-Curtis quadrature nodes. I investigate approximating the Hessian using a one point quadrature rule, Smolyak quadrature of degree 0, 1, and 2, and Stroud quadrature of degree 2, 3, and 5. Again, the Smolyak rules are built on one dimensional Clenshaw-Curtis quadrature rules. Table 5.3 summarizes the error bounds for the different quadrature rules and shows that the degree 2 Smolyak and degree 5 Stroud rules will achieve q-linear convergence. The degree 2 Smolyak rule requires $P = 1105$ collocation points while the Stroud 5 rule requires $P = 61$ collocation points. The exact Hessian (using degree 3 Smolyak requires $P = 2929$. Therefore, using the Stroud 5 rule to approximate the Hessian will give q-linear convergence and requires significantly fewer collocation points than either the degree 2 or 3 Smolyak rules. In the Newton-CG framework, each Newton iteration applies CG to solve the Newton system

$$Hs = -\widehat{J}'(z)$$

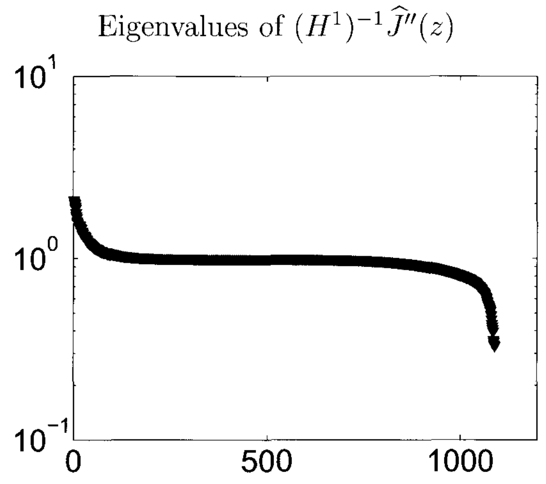
and each CG iteration requires a Hessian times a vector computation. The main computation cost of the Hessian times a vector routine is the $2P$ linear system solves of size 1089 by 1089. Thus, using the Stroud 5 Hessian approximation drastically reduces the number of linear system solves required at each CG iteration.

5.6.2.2 Preconditioned Newton-CG

Along with the Newton's method with inexact Hessian information, I have also developed a preconditioned Newton-CG algorithm where the preconditioner is the one point Hessian approximation, H^1, applied using CG. This application of the preconditioner is no longer a linear process and requires the flexible conjugate gradient algorithm. In order for H^1, to be a good preconditioner for the Newton-PCG method, the eigenvalues of $(H^1)^{-1}\widehat{J}''(z_k)$ should be clustered close to one. Since the eigenvalues of $(H^1)^{-1}\widehat{J}''(z)$ do exhibit this behavior (see figure 5.10), H^1 may work well as a preconditioner for Newton-PCG. Table 5.4 demonstrates the convergence history for Newton-PCG applied to the example problem.

Smolyak	Iteration	$\widehat{J}(z)$	$\Vert\nabla\widehat{J}(z)\Vert_2$	$\Vert s\Vert_2$	Outer CG	Inner CG
1	0	6.428522e-01	6.943939e-02	3.732302e+02	17	1627
	1	4.587335e-02	3.297836e-06	1.705401e+00	40	5161
	2	4.587204e-02	2.098836e-13			
2	0	4.540451e-02	1.659602e-04	1.058052e+01	27	2967
	1	4.536547e-02	1.241542e-10			
3	0	4.543998e-02	9.652572e-05	1.813187e+00	25	2659
	1	4.543616e-02	1.777759e-10			

Table 5.4: Convergence history for the Newton-PCG method. Outer CG refers to the number of CG iterations to solve the preconditioned linear system. Inner CG refers to the number of iterations required to apply the preconditioner.



Interval	Percent (%)
1 ± 0.1	77.8
1 ± 0.2	88.4
1 ± 0.3	93.8
1 ± 0.4	96.3
1 ± 0.5	97.6
1 ± 1.0	99.7

Figure 5.10: The figure depicts the eigenvalues of the preconditioned Newton system. For H^1 to be a good preconditioner, the eigenvalues of $(H^1)^{-1}\widehat{J}''(z)$ should be sufficiently close to 1. The table demonstrates this clustering. For a given interval, the right hand column lists what percentage of the total number of eigenvalues is in that interval.

Notice that the warm started Newton-CG scheme (no preconditioner) required 706 CG iterations for Smolyak degree 1, 737 for Smolyak degree 2, and 357 for Smolyak degree 3. Each CG iteration requires one Hessian times a vector computation. The main computational work required by the Hessian times a vector routine is solving $2P$ linear system of size 1089 by 1089 (i.e. to compute the derivative of the state and the derivative of the adjoint), where the number of quadrature nodes required for degree 1 Smolyak is $P = 401$, for degree 2 Smolyak is $P = 1105$, and for degree 3 Smolyak is $P = 2929$. Using H^1 as a preconditioner requires only two linear system solves per inner iteration. Also, using the preconditioner drastically reduced the number CG iterations required: Smolyak degree 1 required 57 CG iterations, Smolyak degree 2 required 27, and Smolyak degree 3 required 25. Table 5.5 contains the total number of linear system solves required by the Hessian times a vector routine in Newton-CG and Newton-PCG. As seen in the table, Newton-PCG with H^1 preconditioner drastically reduces the number of linear system solves required to converge. In fact,

Smolyak	Newton-CG	Newton-PCG
1	566212	59290
2	1628770	65604
3	2091306	151768
Total	4286288	276662

Table 5.5: Total number of linear system solves of size 1089 by 1089 required to solve the optimal distributed control problem using warm started Newton-CG (center column) and warm started Newton-PCG with preconditioner H^1 (right column).

Newton-PCG requires about 15 times fewer linear system solves than Newton-CG and thus is an advantageous scheme for accelerating Newton's method.

5.6.3 Objective Functions Based on the Expected Value and the Variance

Now, consider an objective function which is a linear combination of the expected value of $\|u - u_0\|^2_{L^2(D)}$ and the variance of $\|u - u_0\|^2_{L^2(D)}$. That is, for any $c_1, c_2 \in \mathbb{R}$, consider

$$J(u, z) = c_1 E\Big[\|u - u_0\|^2_{L^2(D)}\Big] + c_2 \mathrm{Var}\Big[\|u - u_0\|^2_{L^2(D)}\Big] + \frac{\alpha}{2}\|z\|^2_{L^2(D)}.$$

Disregarding the regularization term, this optimization problem could be thought of as a way to approximate the Pareto curve for the bi-objective problem with objectives give by the expected value

$$E\Big[\|u(z;\cdot,\cdot) - u_0\|^2_{L^2(D)}\Big]$$

and the variance

$$\mathrm{Var}\Big[\|u(z;\cdot,\cdot) - u_0\|^2_{L^2(D)}\Big],$$

respectively. The Pareto curve is the curve generated by changing the parameters $c_1, c_2 \in \mathbb{R}$, solving the resulting optimization problem, and plotting the variance versus the expected value. The Pareto curve for this SPDE constrained optimization problem is depicted in figure (5.11). In order to generate this curve, I employed the Normal-Boundary Intersection (NBI) method developed by Das and Dennis [6].

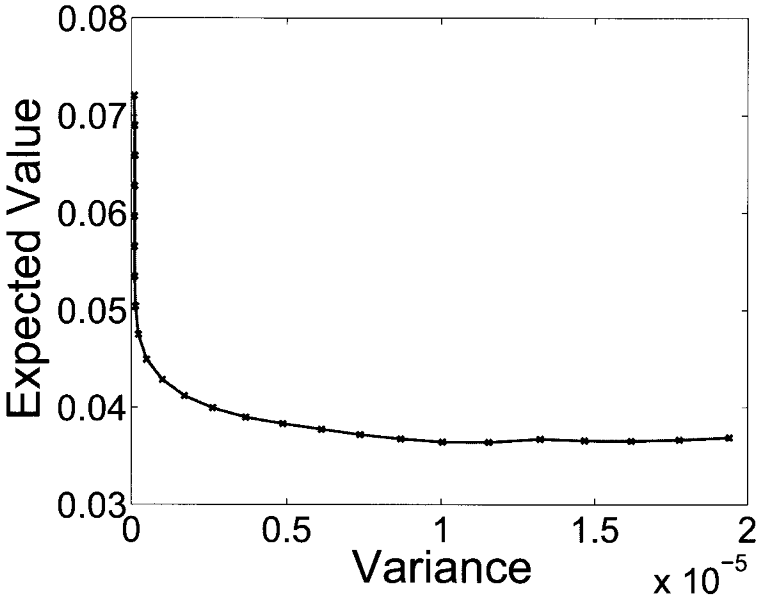


Figure 5.11: Pareto curve for the variance of $\|u-u_0\|^2_{L^2(D)}$ versus the expected value of $\|u-u_0\|^2_{L^2(D)}$ using NBI.

5.6.4 The Effect of Quadrature on the Variance Problem

Many complications may arise when applying the stochastic collocation method to solve the optimization problem

$$\min_{z\in Z}\frac{1}{2}\mathrm{Var}\left[\|u(z;\cdot,\cdot)-u_0\|^2_{L^2(D)}\right]+\frac{\alpha}{2}\|z\|^2_{L^2(D)}$$

where the regularization parameter $\alpha = 10^{-4}$. The variance of $\|u(z;\cdot,\cdot)-u_0\|^2_{L^2(D)}$ is quartic in u; thus, the Hessian need not always be positive definite. Since the conjugate gradient method requires positive definiteness, Newton-CG may only be applied when the current iterate z_c is sufficiently close to the optimal solution (if it exists). On the other hand, many quadrature rules, including Smolyak quadrature rules, have negative weights. With the stochastic collocation, the variance of a random variable X is approximated as

$$\mathrm{Var}[X] = E\big[(X - E[X])^2\big] \approx \sum_{k=1}^{P} \omega_k (X_k - E[X])^2.$$

Thus, even though $(X_k - E[X])^2 \geq 0$, the stochastic collocation approximation of the variance may be negative. Having negative weights may cause some problems for optimization. In some cases, it may be possible to make the objective function arbitrarily small by finding a control z that maximizes

$$\Big(\|u_k - u_0\|^2_{L^2(D)} - E[\|u(y,\cdot) - u_0\|^2_{L^2(D)}]\Big)^2$$

for the indices k that correspond to the negative quadrature weights. In this case, the resulting control may give little to no information on the true solution to the problem. To illustrate this problem, let

$$J(u,z) = \frac{1}{2}\mathrm{Var}\Big[\|u(y,\cdot) - u_0\|^2_{L^2(D)}\Big] + \frac{\alpha}{2}\|z\|^2_{L^2(D)}$$

and consider the optimization problem

$$\min_{z \in Z} \widehat{J}(z) \equiv J(u(z;\cdot,\cdot),z)$$

where $u(z;\cdot,\cdot) = u(\cdot,\cdot)$ solve the state equation

$$\begin{aligned} -10^{-1}\Delta u(y,x) + c(y,x)\cdot\nabla u(y,x) &= z(x) && \text{for } x \in D \\ u(x) &= d(y,x) && \text{for } x \in (\partial D)_D \\ \tfrac{\partial u}{\partial n}(x) &= 0 && \text{for } x \in (\partial D)_N. \end{aligned}$$

The Dirichlet boundary conditions are

$$d(y,x) = 0 \text{ for } x \in (0,1)\times\{0,1\},$$

and on $\{x \in \partial D_D \ : \ x \in \{0\}\times(0,1)\}$

$$d(y,x) = \begin{cases} 0 & \text{if } x_2 \notin (0.25, 0.75), \\ \sin(2\pi(x_2 - 0.25)) & \text{if } x_2 \in (0.25,0.75) \text{ and } y_1 = 0, \\ \sin\big(\pi\dfrac{\exp(4y_1(x_2-0.25)) - 1}{\exp(2y_1) - 1}\big) & \text{if } x_2 \in (0.25,0.75) \text{ and } y_1 \neq 0 \end{cases}$$

and the advection coefficients are

$$c(y,x) = \begin{pmatrix} \cos\left(\frac{\pi}{4}y_2\right) \\ \sin\left(\frac{\pi}{4}y_2\right) \end{pmatrix}.$$

Figure 5.12 depicts the computed optimal controls for this problem. The Smolyak rule was built on 1D Clenshaw-Curtis quadrature nodes and weights. The Smolyak rule integrates polynomials of total degree 1 exactly and contains negative weights. The tensor product rule was built on the same 1D Clenshaw-Curtis rule and integrates polynomials of degree 1 in each direction exactly. Furthermore, the tensor product rule has all positive weights. Figure 5.13 depicts the difference between the

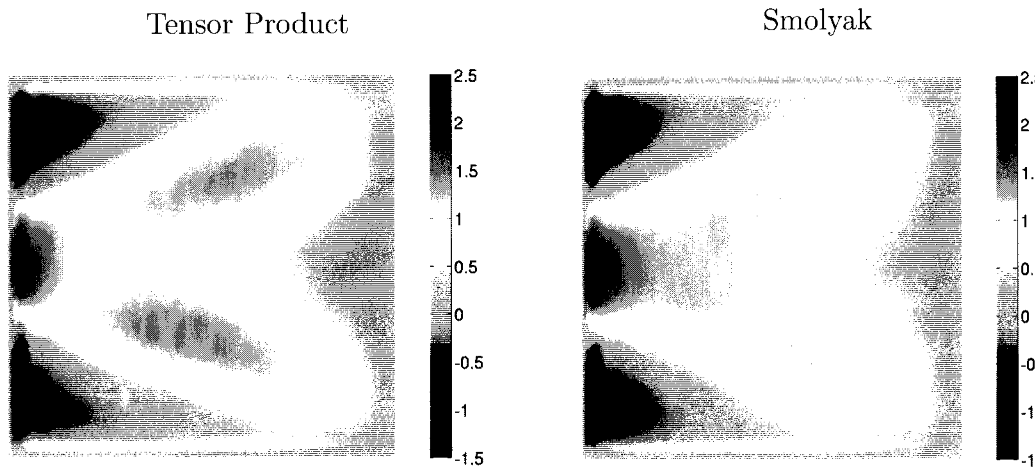


Figure 5.12: Optimal distributed controls for 2 random variable example problem. The left figure depicts the control computed using tensor product quadrature built on 1D Clenshaw Curtis quadrature nodes. The tensor product rule is exact for polynomials of degree 1 in each direction. The right figure depicts the control computed using Smolyak quadrature nodes built on 1D Clenshaw Curtis quadrature nodes. The Smolyak rule is exact for polynomials of total degree 1.

control computed using the tensor product quadrature nodes and the control computed using the Smolyak quadrature nodes. Although, to the naked eye, the controls look rather similar, the maximum magnitude of the differences between the two is

0.4. Furthermore, when using the low order Stroud rules, the characteristics of the

Difference between Tensor Product and Smolyak Controls

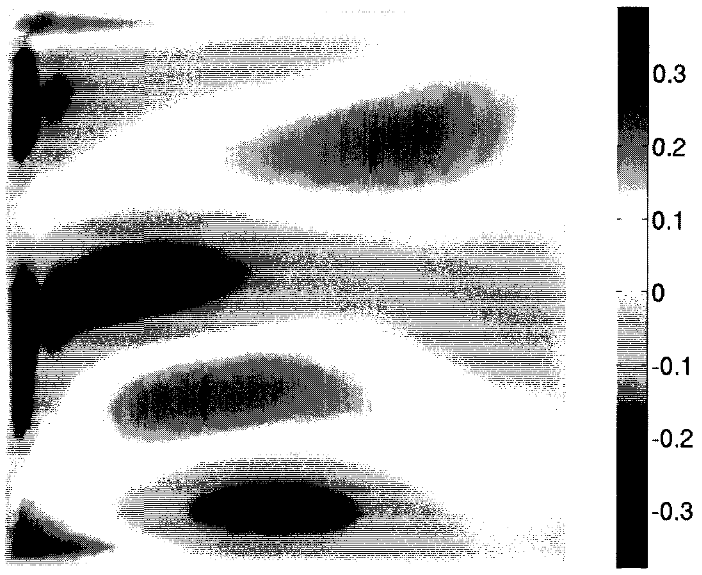
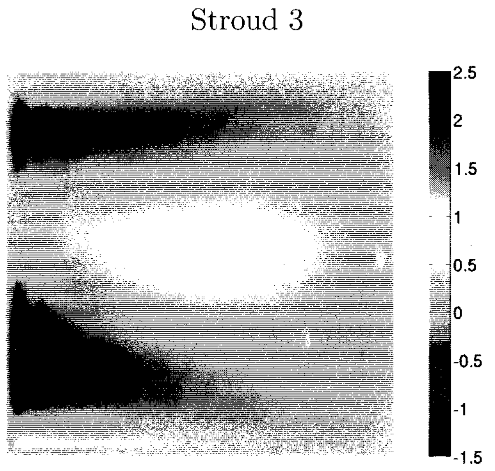


Figure 5.13: Difference between the control computed using the tensor product rule and the control computed using the Smolyak rule.

controls vary greatly from either the control computed using the tensor product rule or the control computed using the Smolyak rule. This variation could be do to the fact that the Stroud rules do not sample the stochastic domain Γ very thoroughly. That is, the Smolyak rule of total degree 3 requires $P = 29$ nodes whereas the degree 3 Stroud rule only requires $P = 4$. The controls computed using the Stroud 3 and 5 rules are plotted in figure 5.14.

Another problem arises when approximating the Hessian with the one point quadrature rule (used in the Newton-PCG and Newton with inexact Hessian information algorithms). In order to compute the Hessian, one must compute the derivative



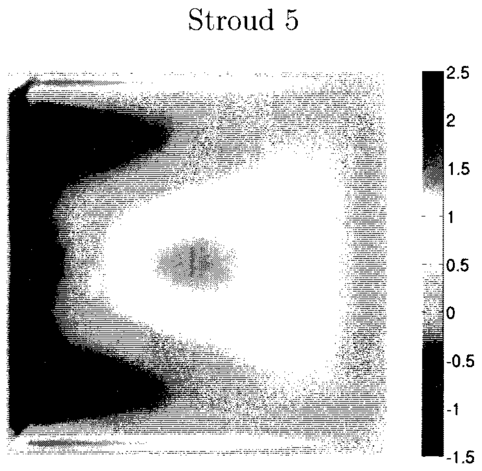


Figure 5.14: Optimal distributed controls for 2 random variable example problem. The left figure depicts the control computed using Stroud 3 quadrature. The Stroud 3 rule exact for polynomials of degree 3 in each direction. The right figure depicts the control computed using Stroud 5 quadrature. The Stroud 5 rule is exact for polynomials of degree 5 in each direction.

of the adjoint which requires forming the product

$$\begin{aligned}\langle J_{uu}(u,z)w,s\rangle_{U^*,U} =& 2E\Big[\langle u(y,\cdot)-u_0, w(y,\cdot)\rangle_{L^2(D)}\langle u(y,\cdot)-u_0, s(y,\cdot)\rangle_{L^2(D)}\Big] \\ &+ E\Big[\|u(y,\cdot)-u_0\|^2_{L^2(D)}\langle w(y,\cdot), s(y,\cdot)\rangle_{L^2(D)}\Big] \\ &- 2E\Big[\langle u(y,\cdot)-u_0, w(y,\cdot)\rangle_{L^2(D)}\Big]E\Big[\langle u(y,\cdot)-u_0, s(y,\cdot)\rangle_{L^2(D)}\Big] \\ &- E\Big[\|u(y,\cdot)-u_0\|^2_{L^2(D)}\Big]E\Big[\langle w(y,\cdot), s(y,\cdot)\rangle_{L^2(D)}\Big]\end{aligned}$$

for any direction $s \in L^2_\rho(\Gamma; H^1_0(D))$. Let $\bar{y}$ denote the quadrature point for the one point quadrature rule, $\bar{u}(\cdot) = u(\bar{y},\cdot)$, $\bar{w}(\cdot) = w(\bar{y},\cdot)$, and $\bar{s}(\cdot) = s(\bar{y},\cdot)$. Approximating the directional derivative $\langle J_{uu}(u,z)w, s\rangle_{U^*,U}$ with the one point quadrature rule yields

$$\begin{aligned}\langle J_{uu}(u,z)w, s\rangle_{U^*,U} \approx & 2E\Big[\langle u(y,\cdot) - u_0, w(y,\cdot)\rangle_{L^2(D)}\Big]\langle \bar{u} - u_0, \bar{s}\rangle_{L^2(D)} \\ & + \|\bar{u} - u_0\|^2_{L^2(D)}\langle \bar{w}, \bar{s}\rangle_{L^2(D)} \\ & - 2E\Big[\langle u(y,\cdot) - u_0, w(y,\cdot)\rangle_{L^2(D)}\Big]\langle \bar{u} - u_0, \bar{s}\rangle_{L^2(D)} \\ & - E\Big[\|u - u_0\|^2_{L^2(D)}\Big]\langle \bar{w}, \bar{s}\rangle_{L^2(D)} \\ = & \Big(\|\bar{u} - u_0\|^2_{L^2(D)} - E\Big[\|u - u_0\|^2_{L^2(D)}\Big]\Big)\langle \bar{w}, \bar{s}\rangle_{L^2(D)}.\end{aligned}$$

This approximation is merely a scaling of the right hand side associated with the derivative of the adjoint equation for the problem of minimizing the expected value. Furthermore, if the term $E\Big[\|u - u_0\|^2_{L^2(D)}\Big]$ is also approximated with the one point rule, then the approximation of J_{uu} is identically equal to zero. Since the product $\langle J_{uu}(u,z)w, s\rangle_{U^*,U}$ corresponds to the right hand side of the derivative of the adjoint equation,

$$\langle e_u(u(z;\cdot,\cdot),z)^*q, s\rangle_{U^*,U} = \langle J_{uu}(u(z;\cdot,\cdot),z)w, s\rangle_{U^*,U} \quad \forall s \in L^2_\rho(\Gamma; H^1_0(D)),$$

and the boundary conditions for the adjoint equation are homogeneous Dirichlet and Neumann, the solution to the derivative of the adjoint equation is the constant function $q = 0$. This leads to a constant Hessian

$$\langle \widehat{J}''(z)v, s\rangle_{Z^*,Z} = \alpha \int_D v(x)s(x)\mathrm{d}x$$

for any vectors $v, s \in Z$. Therefore, approximating the Hessian with a one point quadrature rule may destroy some or all important geometric features of the original problem. As such, care must be taken in using the one point quadrature rule to approximate the Hessian for the Newton-CG with inexact Hessian information and the Newton-PCG methods.

Chapter 6

Conclusions

This thesis focuses on optimization problems governed by stochastic partial differential equations. In order to develop and further the study of such problems, I have covered the existence and uniqueness of linear elliptic SPDEs as well as their numerical solution. Furthermore, I have developed an adjoint based approach to computing derivatives for SPDE constrained optimization problems. When applied with the stochastic collocation method, the adjoint approach for computing gradients parallelizes. Similarly, with the stochastic collocation approach, the Hessian times a vector computation parallelizes. Finally, I have formulated two algorithms: an inexact Newton's method with inexact Hessian information and a preconditioned Newton-CG algorithm. The novelty of these approaches is their ability to handle large problems due to their parallel structure, reduced storage requirements, and reduced computational expense.

Even with these contributions to the field of SPDE constrained optimization, much work still needs to be done. Of utmost importance is a rigorous analysis concerning the convergence of the discretized SPDE constrained optimization problem to the original infinite dimensional problem as discretizations are refined. By applying convergence results for the numerical solution of SPDEs with convergence analysis techniques for deterministic control problems, one can produce such an analysis. With this analysis, the optimization framework I have presented in this thesis will prove its merit. On the other hand, Xiu's paper [41] develops a method of approximating objective functions for stochastic optimization problems by polynomial chaos expansion. Although this seems to be a reasonable approach to solve such problems, the paper only gives numerical convergence results from either 1D or seemingly simple examples. Furthermore, Xiu does not mention derivative computations for such expansions. This method seems to have merit, but many theoretical aspects need to be pinned down before this method can prove its worth. Aside from this approximation technique, there is much work to be done concerning nonlinear or time dependent SPDEs. Although the framework that I have developed is general enough to accommodate for nonlinear and time dependent SPDEs, some interesting computational complexities may arise in their implementation. Finally, as with many deterministic PDE constrained optimization methods, reduced order modeling techniques and adaptive finite element solvers are critical for quick and accurate solutions. One may be able to adapt such model reduction and adaptive FEM schemes to work for SPDEs.

In conclusion, developing the field of optimization problems governed by SPDEs is essential for the accurate mathematical understanding of many physical systems. This thesis gives a rigorous framework for such problems and develops a parallel method for computing derivatives of the objective function. This thesis also develops two forms of the inexact Newton's method which are well suited to solve large problems due to their parallel nature and minimal memory requirements. Aside from these achievements, there are many more aspects of SPDE constrained optimization to be studied and many open questions to be answered.

Appendix A

Function Spaces

This thesis develops an adjoint based optimization scheme for problems governed by stochastic partial differential equations. In order to solve optimization problems governed by SPDEs, one needs a concrete functional analytic framework. This framework should include the function (Banach) spaces for which the optimization, state, and constraint variables live and some notion of convergence of sequences in these spaces. Also, one needs to solidify what it means to be measurable and integrable in these general function spaces. Such a functional analytic framework will make solving the SPDE and the corresponding optimization problems well-defined. Since the SPDE constrained optimization is truly the interface to two fields of study (SPDEs and optimization), I will focus on the two topics separately.

Suppose $D \subset \mathbb{R}^d$ $(d = 1, 2, 3)$ is our physical domain and $(\Omega, \mathcal{F}, P)$ is a complete probability space. Here, Ω is the set of events, $\mathcal{F}$ is a $\sigma-$algebra, and P is a measure such that $P(\Omega) = 1$. A function $u : \Omega \times D \to \mathbb{R}$ is a random field if $u(\cdot, x)$ is a $P-$measurable function (i.e. random variable) for almost every $x \in D$. These functions may act intrinsically different with respect to $\omega \in \Omega$ than with respect to $x \in D$. In the study of SPDEs, the solution u, as well as the input data, are random fields. Suppose u solves an elliptic SPDE. As such, u must satisfy the property that the mapping $x \mapsto u(\cdot, x)$ is weakly differentiable in $L^2(D)$. One may also require that the mapping $\omega \mapsto u(\omega, \cdot)$ have finite p^{th} order moments. This characterization leads to

$$x \mapsto u(\cdot, x) \in H^1(D)$$

and

$$\omega \mapsto u(\omega, \cdot) \in L^p_P(\Omega)$$

where $L^p_P(\Omega) = \left\{ v \; : \; \int_\Omega v(\omega)^p \mathrm{d}P(\omega) < \infty \right\}$. This chapter provides the functional analytic framework for dealing with abstract function spaces as well as the measurability and integrability property of functions that output into general Banach spaces.

A.1 Notions of Convergence

In optimization theory, an overwhelming majority of algorithms are iterative. These algorithms generate sequences of the optimization variables with the goal that the sequence converges to a minimizer of the objective function. Similarly, when attempting to solve SPDEs numerically, one discretizes the spatial and stochastic domains, then solves the resulting finite dimensional problem. As the level of discretization increases, the resulting solution should converge to the solution to the SPDE. Thus, it is critical to understand how these sequences converge. Since this thesis is concerned with SPDEs, I will present convergence definitions that are useful in general Banach spaces and in general measure spaces. For the rest of this section, X will denote a normed vector space, X^* the dual space of X, and $(\Omega, \mathcal{F}, \mu)$ will denote an arbitrary measure space.

First, I will present definitions concerning convergence in general normed vector spaces. These definitions do not necessarily deal with sequences of functions, but may be applied to functions spaces which are also Banach spaces.

Definition A.1.1 *Suppose $\{x_n\}_{n=1}^\infty \subset X$ and $x \in X$.*

1. *x_n converges to x strongly if for all $\epsilon > 0$, there exists an integer $N = N(\epsilon) > 0$ such that $\|x_n - x\|_X < \epsilon$ whenever $n \geq N$.*

2. *x_n converges to x weakly if for all $f \in X^*$, $\lim_{n \to \infty} f(x_n) = f(x)$.*

These definitions are ubiquitous throughout the fields of mathematical and functional analysis. Strong convergence is exactly convergence in norm while weak convergence refers to the idea that for any bounded linear functional, $f : X \to \mathbb{R}$, (i.e. $f \in X^*$), the sequence $\{f(x_n)\}$ converges strongly. The optimization problems with which this thesis deals have optimizers in some Banach space and thus, these definitions are pivotal to understanding the convergence of any sequence generated by an optimization algorithm.

The next few definitions deal with convergence in an arbitrary measure space. These definitions consider sequences of functions that take $(\Omega, \mathcal{F}, P)$ as their domain and X as their range.

Definition A.1.2 *Suppose $f_n : \Omega \to X$ is a sequence of functions and $f : \Omega \to X$.*

1. *f_n converges to f pointwise if for all $\omega \in \Omega$ and for every $\epsilon > 0$, there exists an integer $N = N(\omega, \epsilon) > 0$ such that $\|f_n(\omega) - f(\omega)\|_X < \epsilon$ whenever $n \geq N$.*

2. *f_n converges to f almost everywhere (a.e.) if there exists a set $E_0 \subset \Omega$ with $\mu(E_0) = 0$ such that for all $\omega \in \Omega \backslash E_0$ and for every $\epsilon > 0$, there exists an integer $N = N(\epsilon) > 0$ such that $\|f_n(\omega) - f(\omega)\|_X < \epsilon$ whenever $n \geq N$. This definition is equivalent to pointwise convergence almost everywhere. In probability theory, this is referred to as almost sure (a.s.) convergence.*

3. *f_n converges to f uniformly if for every $\epsilon > 0$, there exists an integer $N = N(\epsilon) > 0$ such that $\|f_n(\omega) - f(\omega)\|_X < \epsilon$ for all $\omega \in \Omega$ whenever $n \geq N$.*

4. *f_n converges to f almost uniformly if for every $\epsilon > 0$ there exists a set $E_\epsilon \subset \Omega$ such that $\mu(E_\epsilon) < \epsilon$ and for every $\delta > 0$ there exists an integer $N = N(\delta) > 0$ such that $\|f_n(\omega) - f(\omega)\|_X < \delta$ for all $\omega \in \Omega \setminus E_\epsilon$ and for all $n \geq N$.*

5. *f_n converges to f in measure if for every $\epsilon > 0$*

$$\mu(\{\omega \in \Omega : \|f_n(\omega) - f(\omega)\|_X > \epsilon\}) \to 0 \text{ as } n \to \infty.$$

Relating these definitions to the definitions of convergence in Banach spaces, a sequence of functions, $f_n : \Omega \to X$, converge pointwise (almost everywhere) if and only if for all $\omega \in \Omega$ (for a.e. $\omega \in \Omega$), the sequence $\{f_n(\omega)\}$ converges strongly to some limit. Uniform convergence and almost uniform convergence are special cases of pointwise and almost everywhere convergence in which the positive integer N does not depend on ω. A simple consequence of these definitions is that uniform convergence implies pointwise convergence which implies almost everywhere convergence, but the other direction is generally not true.

A.2 Weak and Strong (Bochner) Measurability

As stated, the solutions of SPDEs are mappings from a measure space to a Banach space and must satisfy some generalized notion of measurability and integrability. The reader will note that the definitions and results to come are completely analogous to standard measurability and integrability results for functions that map a measure space into $\mathbb{R}$ or $\mathbb{C}$. This section concentrates on the measurability aspects while the next section covers integrability. For proofs of these results, see Einar Hille's book *Functional Analysis and Semi-Groups* [15] or Kôsaku Yosida's book *Functional Analysis* [44].

First, I present definitions concerning any non-separable space X and the ideas of weak and Bochner measurability. The notions of non-separability and weak measurability define a useful characterization of Bochner measurability and will be used later. Throughout this thesis, a simple function always refers to a finitely valued function. To clarify, if $u : \Omega \to X$ is a simple function, then can be written as $u = \sum_{n=1}^{N} u_n \chi_{A_n}$ where $A_n \in \mathcal{F}$, $u_n \in X$, and χ_A denotes the characteristic function of A

$$\chi_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{otherwise.} \end{cases}$$

Definition A.2.1 *Suppose* $u : \Omega \to X$.

1. *u is separably valued if $u(\Omega)$ is separable. It is almost separable valued if there exists a set $E_0 \subset \Omega$ such that $\mu(E_0) = 0$ and $u(\Omega \setminus E_0)$ is separable.*

2. *u is weakly measurable if for all $f \in X^*$, the mapping $\omega \mapsto \langle f, u(\omega)\rangle_{X^*,X}$ is Lebesgue measurable. Here $\langle \cdot, \cdot \rangle_{X^*,X}$ denotes the duality product on X.*

3. *u is Bochner measurable (or strongly measurable) if there exists a sequence of simple functions, $u_n : \Omega \longrightarrow X$, converging to u almost everywhere.*

As stated above, there is a connection between almost separably valued, weakly measurable functions with Bochner measurable functions. These first two properties give a useful characterization of Bochner measurable which allows for easy characterization of some Bochner measurable functions.

Theorem A.2.2 *A function $u : \Omega \longrightarrow X$ is Bochner measurable if and only if it is weakly measurable and almost separably valued.*

Now, suppose X is separable, then a function with range in X is separably valued which implies it is also almost separably valued. Therefore, an easy corollary of the above theorem is that weak measurability and Bochner measurability are equivalent when X is separable.

Corollary A.2.3 *If X is separable, then $u : \Omega \longrightarrow X$ is Bochner measurable if and only if u is weakly measurable.*

The next result is another useful characterization of Bochner measurable functions and demonstrates the connection between Bochner measurability and Borel measurability. Borel measurability refers to functions that outputs into real or complex vector spaces. Therefore, this result yields a tangible result concerning Bochner measurability.

Lemma A.2.4 *If $u : \Omega \longrightarrow X$ is Bochner measurable, then the mapping $t \mapsto \|u(t)\| : \Omega \longrightarrow \mathbb{R}$ is Borel measurable.*

Many numerical schemes generate sequences of functions and, in numerous applications, it is essential to know that the limit function and the sequence functions satisfy the same properties. The following theorem ensures that measurability properties do transfer from sequence functions to limit functions when the sequence converges weakly.

Theorem A.2.5 *If X is separable and $u_n : \Omega \longrightarrow X$ is a sequence of Bochner measurable functions that converges weakly to $u : \Omega \longrightarrow X$, then u is Bochner measurable.*

A.3 Bochner Integrability

The solution to SPDEs must satisfy some desired statistical properties, such as finite p^{th} moments. Since these solutions are thought of as mappings from the probability space $(\Omega, \mathcal{F}, P)$ to some Banach space X ($H^1(D)$ for instance), computing p^{th} order moments requires the idea of integrating functions that output into general Banach spaces (not just $\mathbb{R}$). The definitions and theorems concerning Bochner measurability are instrumental in defining such concepts of integration. As with Lebesgue integration, I will first define integration for simple functions, then extract the integral of a general Bochner measurable function as the limit of the integrals of simple functions. For proofs and more results concerning Bochner integration, consult [9], [32].

Definition A.3.1 *A simple function $u : \Omega \longrightarrow X$ (i.e. $u = \sum_{n=1}^{N} u_n \chi_{A_n}$) is Bochner integrable if and only if the mapping $\omega \longrightarrow \|u(\omega)\|_X$ is Lebesgue integrable. The Bochner integral of u is given by*

$$\int_\Omega u(\omega)\mathrm{d}\mu(\omega) = \sum_{n=1}^{N} u_n \mu(A_n)$$

Since any Bochner measurable function is the a.e. limit of simple functions, the definition of Bochner integrable is readily extended to general Bochner measurable functions. The following definition clarifies this idea.

Definition A.3.2 *A function $u : \Omega \longrightarrow X$ is Bochner Integrable if and only if there exists a sequence of simple functions $u_n : \Omega \longrightarrow X$ converging almost everywhere to u, such that*

$$\lim_{n,m\to\infty} \int_\Omega \|u_m(\omega) - u_n(\omega)\|_X \mathrm{d}\mu(\omega) = 0.$$

The Bochner integral is defined as

$$\int_\Omega u(\omega)\mathrm{d}\mu(\omega) = \lim_{n\to\infty} \int_\Omega u_n(\omega)\mathrm{d}\mu(\omega).$$

Now that Bochner integrability is defined, I will present a useful characterization of Bochner integrability. This characterization is a necessary and sufficient condition for a function that outputs into an arbitrary Banach space to be Bochner integrable.

Theorem A.3.3 *A function $u : \Omega \longrightarrow X$ is Bochner integrable if and only if u is Bochner measurable and $\int_\Omega \|u(\omega)\|_X \mathrm{d}\mu(\omega) < \infty$.*

As defined, the Bochner integral satisfies all the properties of the Lebesgue integral. Namely, the Bochner integral is linear and for any function $u : \Omega \longrightarrow X$, the Bochner integral satisfies

$$\left\| \int_B u(\omega)\mathrm{d}\mu(\omega) \right\|_X \leq \int_B \|u(\omega)\|_X \mathrm{d}\mu(\omega). \tag{A.3.1}$$

Furthermore, one can also define L^p spaces in the same fashion as for Lebesgue integration.

Definition A.3.4 *For $1 \leq p < \infty$, the linear space $L^p(\Omega; X)$ is defined as the space of Bochner integrable functions $u : \Omega \longrightarrow X$ such that*

$$\int_\Omega \|u(\omega)\|_X^p \mathrm{d}\mu(\omega) < \infty.$$

If $p = \infty$, then $L^\infty(\Omega; X)$ is the space of Bochner integrable functions $u : \Omega \longrightarrow X$ such that

$$\operatorname*{ess\,sup}_{\omega\in\Omega} \|u(\omega)\|_X < \infty.$$

All desirable properties of the Lebesgue spaces transfer to the Bochner spaces. The following theorem sums up many of these properties.

Theorem A.3.5 *1. $L^p(\Omega; X)$ with norm $\|v\|_{L^p(\Omega;X)} = (\int_\Omega \|v(\omega)\|_X^p \mathrm{d}\mu(\omega))^{1/p}$ for $1 \le p < \infty$ and norm $\|v\|_{L^\infty(\Omega;X)} = \operatorname{ess\,sup}_{\omega\in\Omega} \|v(\omega)\|_X$ for $p = \infty$ is a Banach space.*

2. For $1 \le p < \infty$, $L^p(\Omega; X)$ is separable if X is.

3. (Hölder's Inequality) Suppose $u \in L^p(\Omega; X)$ and $f \in L^q(\Omega; X^*)$ with $1/p+1/q = 1$. Then the mapping $\omega \mapsto \langle f(\omega), u(\omega)\rangle_{X^*,X} \in L^1(\Omega, X)$ and

$$\int_\Omega \langle f(\omega), u(\omega)\rangle_{X^*,X} \mathrm{d}\mu(\omega) \le \|f\|_{L^q(\Omega;X^*)}\|u\|_{L^p(\Omega;X)}.$$

4. If H is a Hilbert space with inner product $< \cdot, \cdot >_H$, then so is $L^2(\Omega; H)$ with inner product

$$\langle u, v\rangle_{L^2(\Omega;H)} := \int_\Omega \langle u(\omega), v(\omega)\rangle_H \mathrm{d}\mu(\omega).$$

5. If X and Y are Banach Spaces such that $X \hookrightarrow Y$. Then $L^p(\Omega; X) \hookrightarrow L^q(\Omega; Y)$ for $1 \le p \le q \le \infty$.

In the definition of Bochner L^p spaces, X is an arbitrary Banach space. In this research, X has a particular structure. Namely, X is a Sobolev space characterizing the regularity of the solution to an SPDE. Since Sobolev spaces are subspaces of L^p spaces, it is natural to study the tensor product space $L^p(\Omega) \otimes L^q(D)$. Considering Bochner spaces of the form $L^p(\Omega; L^q(D))$ will give an essential isomorphic relationship between the tensor space $L^2(\Omega) \otimes H^1(D)$ and the Bochner space $L^p(\Omega; H^1(D))$; thus classifying the tensor product spaces of interested, as previously mentioned.

Theorem A.3.6 *The space $L^p(\Omega_1; L^q(\Omega_2))$ for $p, q \in [1, \infty)$ is isomorphic to*

$$V = \left\{ v : \Omega_1 \times \Omega_2 \to \mathbb{R}^d \ : \ \int_{\Omega_1} \left(\int_{\Omega_2} |v(y, x)|^q \mathrm{d}x \right)^{p/q} \mathrm{d}y < \infty \right\}.$$

Proof: Consider the map $\tilde{u} \mapsto u$ defined by $[u(y)](x) := \tilde{u}(y,x)$. Thus, pick any $u \in L^q(\Omega_1; L^q(\Omega_2))$, then there exists $\tilde{u} \in V$ such that $\tilde{u}(y,x) = [u(y)](x)$. Similarly, for any $\tilde{u} \in V$, for almost every $\tilde{y} \in \Omega_1$, we have that

$$\int_{\Omega_1} \left(\int_{\Omega_2} |v(\tilde{y},x)|^q \mathrm{d}x \right)^{p/q} \mathrm{d}y = \mu_1(\Omega_1) \int_{\Omega_2} |v(\tilde{y},x)|^q \mathrm{d}x < \infty.$$

Hence, for almost all $y \in \Omega_1$, $\tilde{u}(y,x) \in L^q(\Omega_2)$. Therefore, the mapping $\tilde{u} \mapsto u$ defines an isomorphism. □

As an easy corollary to the previous theorem, consider the case when $p = q$.

Corollary A.3.7 *Suppose* $1 \le p < \infty$. *Then* $L^p(\Omega_1 \times \Omega_2) \cong L^p(\Omega_1; L^p(\Omega_2))$.

An analogous result may be applied to spaces of the form $L^p(\Omega_1; W^{1,q}(\Omega_2))$, which will be pivotal in the following chapters. The resulting isomorphism does not hold for $p = q = \infty$. In this case, one can only prove the following embedding:

Theorem A.3.8 $L^\infty(\Omega_1; L^\infty(\Omega_2)) \subset L^\infty(\Omega_1 \times \Omega_2)$, *but, in general,* $L^\infty(\Omega_1; L^\infty(\Omega_2)) \neq L^\infty(\Omega_1 \times \Omega_2)$.

Proof: First suppose that $u \in L^\infty(\Omega_1; L^\infty(\Omega_2))$, then

$$\|u\|_{L^\infty(\Omega_1 \times \Omega_2)} = \sup_{(y,x) \in \Omega_1 \times \Omega_2} |u(y,x)| \le \sup_{y \in \Omega_1} \sup_{x \in \Omega_2} |u(y,x)| = \|u\|_{L^\infty(\Omega_1; L^\infty(\Omega_2))} < \infty.$$

Therefore, $L^\infty(\Omega_1; L^\infty(\Omega_2)) \subset L^\infty(\Omega_1 \times \Omega_2)$.

On the other hand, assume $\Omega_1 = \Omega_2 = [0, T]$. Then applying the mapping from the previous theorems to the function $u(y) = \chi_{[0,y]}$ yields $\tilde{u}(y,x) = \chi_{[0,y]}(x) \in L^\infty$. This function is not Bochner measurable and hence $\tilde{u}(y,x) \notin L^\infty(\Omega_1; L^\infty(\Omega_2))$. □

In the situation of partial differential equations with uncertain input data, solutions must satisfy regularity properties almost surely and must exhibit statistical properties such as finite variance. This gives rise to stochastic Sobolev spaces: let $(\Omega, \mathcal{F}, P)$ is a probability space and $D \subset \mathbb{R}^d$ then

$$\begin{aligned} &L^q_P(\Omega; W^{s,q}(D)) \\ &\quad = \left\{ v : \Omega \longrightarrow W^{s,q}(D) \ : \ v \text{ Bochner measurable, } \int_\Omega \|v(\omega,\cdot)\|^q_{W^{s,q}(D)} \mathrm{d}P(\omega) < \infty \right\}. \end{aligned}$$

Similar results as stated above for Bochner spaces also hold for the stochastic Sobolev spaces. For example, the following theorem solidifies the isomorphic relationship between

$$L^2(\Omega_1) \otimes H^k(\Omega_2;) \cong L^2(\Omega_1; H^k(\Omega_2)) \cong H^k(\Omega_2; L^2(\Omega_1)).$$

Theorem A.3.9 *The space $L^p(\Omega_1; W^{1,q}(\Omega_2))$ for $p, q \in [1, \infty)$ is isomorphic to*

$$V = \left\{ v : \Omega_1 \times \Omega_2 \to \mathbb{R}^d \ : \ \int_{\Omega_1} \left(\int_{\Omega_2} |v(y,x)|^q + |\nabla v(y,x)|^q \mathrm{d}x \right)^{p/q} \mathrm{d}y < \infty \right\}.$$

Proof: This result uses the exact same isomorphism from above, just changing the range space of the Bochner space to $W^{1,q}(\Omega_2)$. $\square$

Similar definitions and results hold for $H^k(D; L^2(\Omega))$, but more care has to be taken when computing derivatives of functions that output in a general Banach space [4]. With these tools, we can now properly formulate the strong and weak forms of our SPDEs.

Appendix B

Karhunen-Loève Expansion of Random Fields

In general, the random fields do not satisfy the finite dimensional assumption. In the case of stochastic partial differential equations, the finite dimensional noise assumption is absolutely crucial to be able to solve the SPDE numerically. In order to overcome this problem, the standard technique is to decompose the random fields as infinite sums involving the eigenpairs of some linear operator. In particular, this linear operator will be compact and self adjoint. This will require a little background in functional analysis. This chapter will develop probably the most common such expansion. This expansion is the Karhunen-Lòeve expansion (KL expansion). I will begin with some basic functional analysis results. Most of the results in this first section are from Peter Lax's book [21]. Then, I will derive the KL expansion. Finally, I will give a few convergence results concerning the expansion and present a numerical example.

B.1 Compact, Self Adjoint Operators

The goal of this section is to understand how to decompose a linear operator into a spectral decomposition. In order to do this, notions of compactness, self adjoint, and positivity must be defined in the case of linear operators. To define the notion of a compact operator, I require the notion of precompactness.

Definition B.1.1 *Suppose S is a subset of a complete metric space, then S is precompact if its closure is compact.*

There are many alternative characterizations of precompactness. The following theorem gives two general characterizations.

Theorem B.1.2 *Suppose S is a subset of a complete metric space.*

1. *S is precompact if and only if every sequence in S has a Cauchy subsequence.*
2. *S is precompact if and only if S can be covered by a finite number of balls of a fixed, arbitrary radius.*

With an understanding of precompactness, one can define what it is to be a compact operator. Note that for the remainder of this chapter, X and Y will denote Banach spaces.

Definition B.1.3 *A linear operator $T : X \to Y$ is compact if the image TB where B denotes the unit ball in X is precompact in Y.*

In many applications, T is an integral operator. If X is a Banach space of functions that map D_1 to $\mathbb{R}$ and Y is a Banach spaces of functions that D_2 to $\mathbb{R}$, then one can define the operator $T : X \to Y$ with respect to some kernel $K : D_1 \times D_2 \to \mathbb{R}$ as:

$$Tu(\cdot) = \int_{D_1} K(x, \cdot)u(x)\mathrm{d}x. \tag{B.1.1}$$

It is then useful to determine whether or not such an operator is compact. The following theorem gives compactness results for three different pairs of Banach spaces X and Y.

Theorem B.1.4 *Consider the integral operator (B.1.1).*

1. *If $X = L^1(D_1)$ and $Y = C^0(D_2)$, then (B.1.1) is compact if the kernel is continuous on $D_1 \times D_2$.*

2. *If $X = C^0(D_1)$ and $Y = C^0(D_2)$, then (B.1.1) is compact if the kernel is a continuous function on D_1 in the L^1 norm on D_2.*

3. *If the kernel of (B.1.1) is $L^2(D_1 \times D_2)$, then (B.1.1) is a compact operator from $X = L^2(D_1)$ to $Y = L^2(D_1)$.*

Aside from compact operators, the derivation of the KL expansion requires the idea of self adjoint operators. Much of the theory of self adjoint matrices can be generalized for self adjoint operators and this theory lays the framework for the KL expansion.

Definition B.1.5 *Suppose A is an operator mapping a Hilbert space H with inner product $(\cdot, \cdot)$ into itself. Then A is self adjoint if for all $u, v \in H$,*

$$(Au, v) = (u, Av).$$

Another useful definition concerns the positivity of such operators.

Definition B.1.6 *Suppose A is a self adjoint operator on the Hilbert space H. Then A is positive if the quadratic form (Au, u) is nonnegative for all $u \in H$.*

With these definitions, one can investigate the spectrums and eigenspaces of such operators. As eluded to at the beginning of this discussion, self adjoint matrix theory can be generalized to self adjoint operators. Since self adjoint matrices have real eigenvalues and eigenvectors that form an orthonormal basis, one would hope that the same can be said for self adjoint operators. The following theorem clarifies this idea.

Theorem B.1.7 *Suppose A is a self adjoint operator on the Hilbert space H. Then there is an orthonormal basis $\{\phi_n\}$ for H consisting of eigenfunctions of A. Furthermore, the corresponding eigenvalues are real and their only point of accumulation is 0.*

The connections between matrix theory and operator theory can be taken one step further by noting the following result.

Theorem B.1.8 *Suppose the self adjoint operator A is positive, then the spectrum of A is a subset of the nonnegative real line.*

As a simple consequence of these results, one can derive the KL expansion. The only task is to determine which operator is appropriate to expand the random fields a and f. In the following sections, I will choose an operator and form the KL expansion. I will then present results that state that the KL expansion is, in some sense, optimal.

B.2 KL Expansion

Suppose $a \in L^2(\Omega \times D)$ is a random field. The covariance function of a is:

$$K(x,x') = \int_\Omega \left(a(\omega,x) - \int_\Omega a(\omega',x)\mathrm{d}P(\omega') \right) \left(a(\omega,x') - \int_\Omega a(\omega',x)\mathrm{d}P(\omega') \right) \mathrm{d}P(\omega)$$

or equivalently:

$$K(x,x') = E\Big[(a(\cdot,x) - E[a(\cdot,x)])(a(\cdot,x') - E[a(\cdot,x')])\Big]. \tag{B.2.1}$$

Now, using this as the kernel, define the linear operator, $A : L^2(D) \to L^2(D)$ such that for all $u \in L^2(D)$,

$$Au(x') = \int_D K(x,x')u(x)\mathrm{d}x. \tag{B.2.2}$$

The following theorem classifies this operator.

Theorem B.2.1 *The linear operator A defined by (B.2.2) is compact, self adjoint, and positive. Therefore, the eigenfunctions of A form an orthonormal basis of $L^2(D)$*

and the eigenvalues are all nonnegative real numbers with only one accumulation point, namely 0.

Proof: First, notice the Cauchy-Schwarz and Jensen's inequalities (applied to the concave function $f(x) = \sqrt{x}$ and to the convex function $g(x) = x^2$), imply that

$$\begin{aligned}
\int_{D\times D} K(x,x')^2 \mathrm{d}m^2(x,x') =& \int_{D\times D} E\Big[(a(\cdot,x) - E[a(\cdot,x)])(a(\cdot,x') - E[a(\cdot,x')])\Big]^2 \mathrm{d}x\mathrm{d}x' \\
\leq& \int_{D\times D} E\Big[(a(\cdot,x) - E[a(\cdot,x)])^2\Big]^{1/2} \\
&\times E\Big[(a(\cdot,x') - E[a(\cdot,x')])^2\Big]^{1/2} \mathrm{d}x\mathrm{d}x' \\
=& \left(\int_D E[(a(\cdot,x) - E[a(\cdot,x)])^2]^{1/2} \mathrm{d}x\right) \\
&\times \left(\int_D E[(a(\cdot,x') - E[a(\cdot,x')])^2]^{1/2} \mathrm{d}x'\right) \\
=& \left(\int_D E[(a(\cdot,x) - E[a(\cdot,x)])^2]^{1/2} \mathrm{d}x\right)^2 \\
\leq& \left(\int_D E[(a(\cdot,x) - E[a(\cdot,x)])^2] \mathrm{d}x^{1/2}\right)^2 \\
=& \int_D E[(a(\cdot,x) - E[a(\cdot,x)])^2]^{1/2} \mathrm{d}x \\
=& \Big\|a(\omega,x) - E[a(\cdot,x)]\Big\|^2_{L^2(\Omega\times D)} \\
\leq& \Big(\|a\|_{L^2(\Omega\times D)} + \|E[a(\cdot,x)]\|_{L^2(\Omega\times D)}\Big)^2 \\
\leq& 4\|a\|^2_{L^2(\Omega\times D)}.
\end{aligned}$$

Since the random field a is in $L^2(\Omega \times D)$, the integral operator $K \in L^2(D \times D)$. Therefore, theorem B.1.4, implies that A is a compact operator from $L^2(D)$ to $L^2(D)$.

Now, by definition, K is symmetric (i.e. $K(x,x') = K(x',x)$). Using this fact, notice that for all $u, v \in L^2(D)$

$$\begin{aligned}
\int_D (Au(x'))v(x')\mathrm{d}x' &= \int_D \left(\int_D K(x,x')u(x)\mathrm{d}x \right) v(x')\mathrm{d}x' \\
&= \int_D \left(\int_D K(x',x)u(x)\mathrm{d}x \right) v(x')\mathrm{d}x' \\
&= \int_D u(x) \left(\int_D K(x',x)v(x')\mathrm{d}x' \right) \mathrm{d}x \\
&= \int_D u(x)(Av(x))\mathrm{d}x
\end{aligned}$$

and A is self adjoint.

Finally, by Fubini's theorem, the following holds true: for all $u \in L^2(D)$

$$\begin{aligned}
\int_D (Au(x'))u(x')\mathrm{d}x' &= \int_D \left(\int_D E[(a(\cdot,x) - E[a(\cdot,x)])(a(\cdot,x') - E[a(\cdot,x')])]u(x)\mathrm{d}x \right) u(x')\mathrm{d}x' \\
&= E\left[\int_D \int_D (a(\cdot,x) - E[a(\cdot,x)])(a(\cdot,x') - E[a(\cdot,x')])u(x)u(x')\mathrm{d}x\mathrm{d}x' \right] \\
&= E\left[\left(\int_D (a(\cdot,x) - E[a(\cdot,x)])u(x)\mathrm{d}x \right) \left(\int_D (a(\cdot,x') - E[a(\cdot,x')])u(x')\mathrm{d}x' \right) \right] \\
&= E\left[\left(\int_D (a(\cdot,x) - E[a(\cdot,x)])u(x)\mathrm{d}x \right)^2 \right] \\
&\geq 0.
\end{aligned}$$

Hence, A is positive.

With these three properties, theorems B.1.7 and B.1.8 ensures that the eigenfunctions of A form an orthonormal basis of $L^2(D)$ and the eigenvalues of A are positive real numbers that have a single accumulation point at 0. □

Notice that, without lose of generality, $E[a(\cdot,x)] = 0$. If $E[a] \neq 0$, then consider the random field $a(\omega,x) - E[a(\cdot,x)]$ and define the equivalent kernel and integral operator. Now, let (λ_n, ψ_n) denote the eigenvalues and eigenfunctions of the operator A. Since $\{\psi_n\}$ is an orthonormal basis of $L^2(D)$ the random field a has the following spectral decomposition

$$a(\omega, x) = \sum_{n=1}^{\infty} a_n(\omega)\psi_n(x).$$

Now, multiplying through by ψ_m, integrating, and applying the Lebesgue Dominated convergence theorem,

$$\begin{aligned}
\int_D \psi_x a(\omega, x)\mathrm{d}x &= \int_D \psi_x \sum_{n=1}^{\infty} a_n(\omega)\psi_n(x)\mathrm{d}x \\
&= \int_D \sum_{n=1}^{\infty} a_n(\omega)\psi_x\psi_n(x)\mathrm{d}x \\
&= \sum_{n=1}^{\infty} \int_D a_n(\omega)\psi_x\psi_n(x)\mathrm{d}x \\
&= a_m(\omega).
\end{aligned}$$

The above formula thus defines the coefficients of the expansion of a. These coefficients are random variables. First notice that these coefficients are orthogonal: for $m \neq n$

$$\begin{aligned}
E[a_m a_n] &= E\left[\left(\int_D \psi_x a(\cdot, x)\mathrm{d}x\right)\left(\int_D \psi_n(x')a(\cdot, x')\mathrm{d}x'\right)\right] \\
&= \int_D \left(\int_D E[a(\cdot, x)a(\cdot, x')]\psi_x \mathrm{d}x\right)\psi_n(x')\mathrm{d}x' \\
&= \int_D \lambda_m \psi'_x \psi_n(x')\mathrm{d}x' \\
&= 0
\end{aligned}$$

since the $\{\psi_n\}$ are an orthonormal basis of $L^2(D)$. Also notice that these coefficients have expected value equal to zero.

$$\begin{aligned}
E[a_m] &= E\left[\int_D a(\cdot, x)\psi_x \mathrm{d}x\right] \\
&= \int_D E[a(\cdot, x)]\psi_x \mathrm{d}x \\
&= 0
\end{aligned}$$

since $E[a(\cdot,x)] = 0$ by assumption. Now, to normalize the random coefficients $\{a_n\}$, notice

$$\begin{aligned}
E[a_m^2] &= E\left[\left(\int_D a(\cdot,x)\psi_x \mathrm{d}x\right)\left(\int_D a(\cdot,x')\psi'_x \mathrm{d}x'\right)\right] \\
&= E\left[\int_D\int_D a(\cdot,x)a(\cdot,x')\psi_x \mathrm{d}x \psi'_x \mathrm{d}x'\right] \\
&= \int_D\int_D E[a(\cdot,x)a(\cdot,x')\psi_x \mathrm{d}x \psi'_x \mathrm{d}x' \\
&= \int_D (A\phi'_x)\psi'_x \mathrm{d}x' \\
&= \int_D \lambda_m \psi'^2_x \mathrm{d}x' \\
&= \lambda_m.
\end{aligned}$$

Thus, defining the random variables $Y_m = \frac{1}{\sqrt{\lambda_m}} a_m$, $\{Y_m\}$ is an orthonormal set in $L^2(\Omega)$ with zero mean. The expansion of a is now given by

$$a(\omega,x) = \sum_{n=1}^{\infty} \sqrt{\lambda_n}\psi_n(x)Y_n(\omega).$$

Now, suppose $E[a(\cdot,x)]$ is not identically zero, computing the ψ_n and λ_n using $\tilde{a}(\omega,x) = a(\omega,x) - E[a(\cdot,x)]$. This gives the following expansion

$$a(\omega,x) - E[a(\cdot,x)] = \sum_{n=1}^{\infty} \sqrt{\lambda_n}\psi_n(x)Y_n(\omega).$$

In order to define Y_n, notice that

$$\begin{aligned}
Y_n(\omega) &= \frac{1}{\sqrt{\lambda_n}}\int_D \psi_n(x)\tilde{a}(\omega,x)\mathrm{d}x \\
&= \frac{1}{\sqrt{\lambda_n}}\int_D \psi_n(x)(a(\omega,x) - E[a(\cdot,x)])\mathrm{d}x.
\end{aligned}$$

This construction yields the KL expansion of the random field a.

Definition B.2.2 *For a random field $a \in L^2(\Omega \times D)$, the Karhunen-Loève expansion of a is given by:*

$$a(\omega,x) = E[a(\cdot,x)] + \sum_{n=1}^{\infty} \sqrt{\lambda_n}\psi_n(x)Y_n(\omega) \qquad \text{(B.2.3)}$$

where ψ_n and λ_n are the eigenfunctions and eigenvalues of the compact, self adjoint, positive operator (B.2.2) (i.e. $\{\psi_n\}$ are an orthonormal basis of $L^2(D)$ and $\{\lambda_n\}$ are positive and real), and Y_n are given by

$$Y_n(\omega) = \frac{1}{\sqrt{\lambda_n}} \int_D \psi_n(x)(a(\omega, x) - E[a(\cdot, x)])\mathrm{d}x.$$

Furthermore, $\{Y_n\}$ have the following three properties:

1. *$E[Y_n] = 0$ for all n*
2. *$E[Y_n Y_m] = 0$ for all $n \neq m$*
3. *$E[Y_n^2] = 1$ for all n*

Thus, $\{Y_n\}$ are uncorrelated random variables with zero mean and unit variance.

B.3 Optimality of the KL Expansion

The KL expansion is a clever way of approximating any random field a. If one can determine the eigenvalues and eigenfuctions of the covariance of a, then a can be approximated by the partial sums of the KL expansion. Since such a truncation yields an approximation of a, one would like to be able to say that the approximation is optimal in some sense. I will now present results showing that the partial sums of the KL expansion minimize the error $\|a - P_N a\|^2_{L^2(\Omega\times D)}$, where $P_N a$ denotes a projection of a onto some finite dimensional subspace of dimension N. The following results from Schwab and Todor's paper *Karhunen-Loève approximation of random fields by generalized fast multipole methods* [34] do exactly that.

Theorem B.3.1 *Suppose $a \in L^2(\Omega \times D)$ has KL expansion (B.2.3), then for any positive integer N,*

$$\inf\left\{\|a - P_N a\|^2_{L^2(\Omega\times D)} \ : \ P_N \textit{ projects onto } U \subset L^2(\Omega) \textit{ with } \mathit{dim}(U) = N\right\} = \sum_{n=N+1}^{\infty} \lambda_n$$

and the infimum is attained with $U = span\{\phi_1, ..., \phi_N\}$ (i.e. P_N is the N^{th} partial sum of the KL expansion).

Furthermore, they present sufficient condition for the KL expansion of a to converge.

Theorem B.3.2 *The KL expansion of a converges P-a.s. in $L^\infty(D)$ if*

$$\sum_{n=1}^{\infty} \lambda_n (\log n)^2 \|\phi_n\|^2_{L^\infty(D)} \, Var(Y_n) < \infty.$$

The final result of this section will determine how quickly the partial sums of the KL expansion converge to the random field a. By theorem B.3.1, we know that the truncation error is the sum of the remaining eigenvalues of the integral operator (B.2.2). Thus, one needs to know how fast the eigenvalues decay in order to be able to judge convergence rate. The following theorems displays some decay rates given that the kernel of the integral operator (B.2.2) satisfies certain properties.

Theorem B.3.3 *Let $D \subset \mathbb{R}^d$ be a bounded domain. Suppose $K \in L^2(D \times D)$ is the kernel defined in equation (B.2.1) and A is the corresponding integral operator (B.2.2) with eigenvalues $\{\lambda_n\}$.*

1. *If K is piecewise analytic on $D \times D$, then there exists constants $C_1, C_2 > 0$ depending on K only such that*

$$0 \leq \lambda_n \leq C_1 e^{-C_2 n^{1/d}}.$$

2. *If K is piecewise $H^k \otimes L^2$ on $D \times D$, then there exists a constant $C_3 > 0$ depending on K only such that*

$$0 \leq \lambda_n \leq C_3 n^{-k/d}.$$

3. *If K is piecewise smooth on $D \times D$, then for any $s > 0$ there exists a constant $C_4 > 0$ depending on s and K such that*

$$0 \leq \lambda_n \leq C_4 n^{-s}.$$

B.4 Numerical Computation of the KL Expansion

When required to compute the partial sums of the KL expansion, in general, one does not know the analytic forms of the eigenfunctions and eigenvalues of the integral operator A. Thus some approximation is necessary. That is, suppose V is a Hilbert space, then the goal is to find the pairs $(\lambda, \psi) \in \mathbb{R} \times V$ such that

$$A\psi = \int_D K(x, x')\psi(x')\mathrm{d}x' = \lambda\psi.$$

As is standard in the theory of PDEs, one can write down the variational formulation of this problem and use it as a starting block to approximate the eigenvalues and eigenfunctions. The variational form is as follows: find $(\lambda, \psi) \in \mathbb{R} \times V$ such that

$$\int_D w(x) \int_D K(x, x')\psi(x')\mathrm{d}x'\mathrm{d}x = \lambda \int_D \psi(x)w(x)\mathrm{d}x$$

for all $w \in V$.

I will discuss two methods of approximating these eigenvalues and eigenfunctions. Now, define $V_h \subset V$ to be a finite dimensional subspace of V with basis $\{\phi_i\}_{i=1}^N$. The first approach approximates the random field a with a linear combination of the basis functions:

$$a_h(\omega, x) = \sum_{i=1}^{N} a_i(\omega)\phi_i(x).$$

Note that if one choose $\{\phi_i\}$ to be a nodal basis (i.e. $\phi_i(x_j) = \delta_{ij}$ where x_j is a node of a mesh element), then the random variables $a_i(\omega) = a(\omega, x_i)$. Assume that $\{\phi_i\}$ is nodal. Then, the approximated covariance function can be written as follows:

$$\begin{aligned}
K_h(x,x') &= E[a_h(x)a_h(x')] - E[a_h(x)]E[a_h(x')] \\
&= E[(\sum_{i=1}^{N} a_i(\cdot)\phi_i(x))(\sum_{j=1}^{N} a_j(\cdot)\phi_j(x'))] - E[\sum_{i=1}^{N} a_i(\cdot)\phi_i(x)]E[\sum_{j=1}^{N} a_j(\cdot)\phi_j(x')] \\
&= E[\sum_{i,j=1}^{N} \phi_i(x)\phi_j(x')a_i(\cdot)a_j(\cdot)]] - E[\sum_{i=1}^{N} a_i(\cdot)\phi_i(x)]E[\sum_{j=1}^{N} a_j(\cdot)\phi_j(x')] \\
&= \sum_{i,j=1}^{N} \phi_i(x)\phi_j(x')E[a_i(\cdot)a_j(\cdot)] - \sum_{i,j=1}^{N} \phi_i(x)\phi_j(x')E[a_i(\cdot)]E[a_j(\cdot)] \\
&= \sum_{i,j=1}^{N} \phi_i(x)\phi_j(x')(E[a_i(\cdot)a_j(\cdot)] - E[a_i(\cdot)]E[a_j(\cdot)]) \\
&= \sum_{i,j=1}^{N} \phi_i(x)\phi_j(x')\Sigma_{i,j}
\end{aligned}$$

where $\Sigma_{i,j} = E[a_i(\cdot)a_j(\cdot)] - E[a_i(\cdot)]E[a_j(\cdot)])$. Now, constructing the discretized operator A_h,

$$A_h v(x) = \int_D K_h(x,x')v(x')\mathrm{d}x'$$

and the associated weak (finite element) formulation is: find $(\lambda, \psi_h) \in \mathbb{R} \times V_h$ such that

$$\int_D \phi_i(x) \int_D K_h(x,x')\psi_h(x')\mathrm{d}x'\mathrm{d}x = \lambda \int_D \psi_h(x)\phi_i(x)\mathrm{d}x \tag{B.4.1}$$

for all $i = 1, ..., N$. Writing

$$\psi_h(x) = \sum_{i=1}^{N} v_i\phi_i(x),$$

equation (B.4.1) becomes:

$$\int_D \phi_i(x) \int_D K_h(x,x') \sum_{j=1}^{N} v_j\phi_j(x')\mathrm{d}x'\mathrm{d}x = \lambda \int_D \sum_{j=1}^{N} v_j\phi_j(x)\phi_i(x)\mathrm{d}x$$

Plugging in the expansion of K_h,

$$\int_D \phi_i(x) \int_D \sum_{k,l=1}^{N} \phi_k(x)\phi_l(x')\Sigma_{k,l} \sum_{j=1}^{N} v_j\phi_j(x')\mathrm{d}x'\mathrm{d}x = \lambda \int_D \sum_{j=1}^{N} v_j\phi_j(x)\phi_i(x)\mathrm{d}x.$$

Rearranging the terms, yields

$$\sum_{j,k,l=1}^{N} \Sigma_{k,l} v_j \int_D \phi_i(x)\phi_k(x)\mathrm{d}x \int_D \phi_l(x')\phi_j(x')\mathrm{d}x' = \lambda \sum_{j=1}^{N} v_j \int_D \phi_j(x)\phi_i(x)\mathrm{d}x.$$

Thus, let M denote the standard mass matrix from the finite element method. That is,

$$M_{ij} = \int_D \phi_i(x)\phi_j(x)\mathrm{d}x,$$

then the reformulated weak form can be rewritten as a generalized eigenvalue problem of the following form: find $(\lambda, v) \in \mathbb{R}^{N+1}$ such that

$$M\Sigma M v = \lambda M v \tag{B.4.2}$$

where $v = [v_i]$ and $\Sigma = [\Sigma_{i,j}]$.

The second method of approximating the eigenvalues and eigenfunctions does not first discretize the random field a, rather, it assumes that the eigenfunctions live in the finite dimension space V_h and solve the weak problem: find $(\lambda, \psi) \in \mathbb{R} \times V_h$ such that

$$\int_D \phi_i(x) \int_D K(x,x')\psi_h(x')\mathrm{d}x'\mathrm{d}x = \lambda \int_D \psi_h(x)\phi_i(x)\mathrm{d}x$$

for all $i = 1, ..., N$. Notice plugging in the basis expansion of ψ_h, we get

$$\int_D \phi_i(x) \int_D K(x,x') \sum_{j=1}^{N} v_j\phi_j(x')\mathrm{d}x'\mathrm{d}x = \lambda \int_D \sum_{j=1}^{N} v_j\phi_j(x)\phi_i(x)\mathrm{d}x.$$

Rearranging terms,

$$\sum_{j=1}^{N} v_j \int_D \phi_i(x) \int_D K(x,x')\phi_j(x')\mathrm{d}x'\mathrm{d}x = \lambda \sum_{j=1}^{N} v_j \int_D \phi_j(x)\phi_i(x)\mathrm{d}x.$$

Thus, the weak form again turns into a generalized eigenvalue problem. This time, the problem is: find $(\lambda, \psi) \in \mathbb{R} \times V_h$ such that

$$Sv = \lambda M v \tag{B.4.3}$$

where $S = [S_{i,j}]$ and

$$S_{i,j} = \int_D \phi_i(x) \int_D K(x,x')\phi_j(x')\mathrm{d}x'\mathrm{d}x.$$

In most cases, one cannot compute this integral analytically, thus needs to employ some quadrature rule. Notice that if one applies a Newton-Coates quadrature rule using the nodes $\{x_i\}$ from the mesh, the integral equation reduces to:

$$S_{i,j} = w_i w_j K(x_i, x_j)$$

where $w = [w_i]$ are the quadrature weights. This is due to the fact that $\{\phi_i\}$ was chosen to be a nodal basis. Other methods can be generated using Gaussian quadrature on each element separately.

B.5 Examples

Gaussian random fields arise in many applications for multiple reasons: asymptotically due to the Central Limit Theorem and by approximation since the normal distribution is the maximum entropy distribution when the first and second moments are known. Thus, a Gaussian random field a is completely determined by it's expectation and it's covariance function. In this example, I investigate a Gaussian random field with covariance function given by the Green's function for the two point boundary value problem:

$$-\frac{\mathrm{d}^2u}{\mathrm{d}x^2} = f \text{ in } (0,1) \quad \text{(B.5.1)}$$

$$u(0) = 0 \quad \text{(B.5.2)}$$

$$u(1) = 1. \quad \text{(B.5.3)}$$

$$\text{(B.5.4)}$$

The Green's function is the solution to (B.5.1) for $f = \delta_0(x-y)$ and is given by the following formula:

$$G(x,y) = (1-y)x\chi_{[0,y)}(x) + y(1-x)\chi_{(y,1]}(x). \quad \text{(B.5.5)}$$

By definition, Green's functions are always symmetric. Also, the Green's function is positive on the set of interest, $(0,1)$, and is $L^2((0,1)^2)$. Therefore, the integral operator with Kernel $G(x,y)$ satisfies the conditions in the previous section, and the corresponding eigenvalues are positive, real numbers and the eigenfunctions form an orthonormal basis. Moreover, since the kernel is the Green's function of (B.5.1), the eigenfunctions are the eigenfunctions of the negative second derivative operator with 0 boundary conditions and the eigenvalues are the reciprocals the the eigenvalues of the negative second derivative operator with 0 boundary conditions. Thus, the eigenfunctions and eigenvalues are:

$$\psi_n(x) = \sin(n\pi x) \tag{B.5.6}$$

$$\lambda_n = \frac{1}{(n\pi)^2}. \tag{B.5.7}$$

This leads to the following KL expansion:

$$a(\omega,x) = E[a(\cdot,x)] + \sum_{n=1}^{\infty} \frac{1}{n\pi} \sin(n\pi x) Y_n(\omega)$$

where Y_n are random variables with mean zero and unit variance.

The purpose of this example is to determine how to generate realizations of the a and the partial sums of the KL expansion of a as well as to compute numerical approximations to the truncation error generated by the partial sums.

Let $x_i = i\Delta x$ for $i = 0, 1, ..., N$, $\Delta x = \frac{1}{N}$, and define the covariance matrix of a on the mesh $\{x_i\}$ as the $(N+1) \times (N+1)$ matrix with entries:

$$\Sigma_{ij} = G(x_i, x_j).$$

Figure B.1 depicts the covariance matrix generated using the aforementioned covariance function.

In order to generate realizations of a, one can draw samples from a multivariate normal distribution with mean vector $\mu_i = E[a(\cdot, x_i)]$ and covariance matrix Σ (see figure B.2).

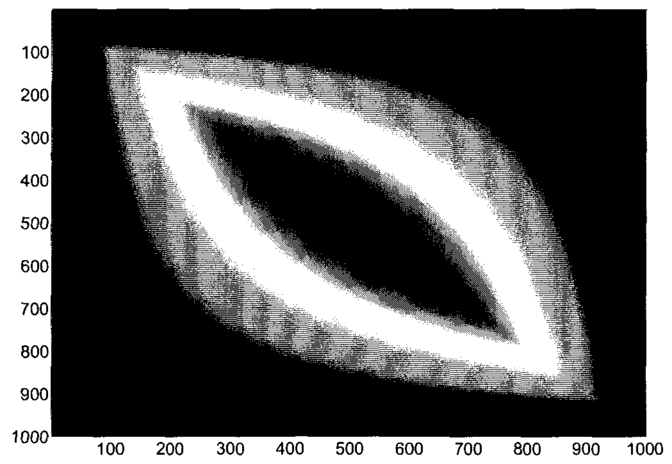


Figure B.1: Covariance function for the 1D Green's function.

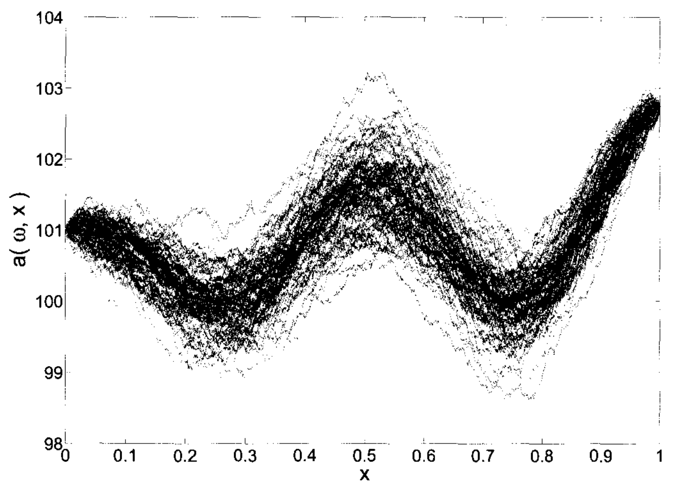


Figure B.2: 100 realizations of the Gaussian random field generated by the 1D Green's function.

Figure B.3 shows that indeed these realizations were drawn from a Gaussian random field. That is, if one chooses any finite number of spatial locations $\{z_i\}$, then the random variables $a(\omega, z_i)$ are distributed according to a multivariate normal distribution.

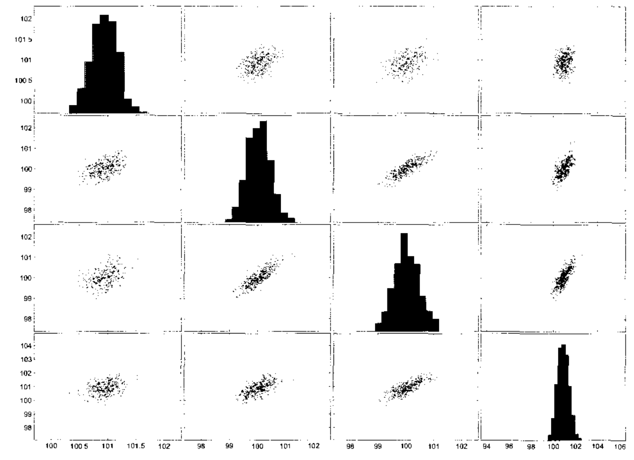


Figure B.3: Distribution of the random field for at 4 different spatial locations.

An important issue to consider when computing approximations to the KL expansion is how do the eigenvalues and eigenvectors of the discretized covariance function relate to the eigenvalues and eigenfunctions of the continuous covariance function? Figure B.4 demonstrates that the eigenvalues of the discretized operator compare rather well with those of the continuous operator.

In order to compute realizations of the partial sums of the KL expansion of the random field a, one must compute the zero mean, unit variance random variables

$$Y_i(\omega) = \frac{1}{\sqrt{\lambda_i}} \int_0^1 (a(\omega, x) - E[a(\cdot, x)])\psi_i(x)\mathrm{d}x.$$

Given realizations of a at the grid points, $a(\omega_k, x_i)$, one can compute approximate realizations of the Y_i using the following formula:

$$Y_i^{(k)} = \Delta x \sum_{j=0}^{N} (a(\omega_k, x_j) - \mu_j)\psi_i(x_j).$$

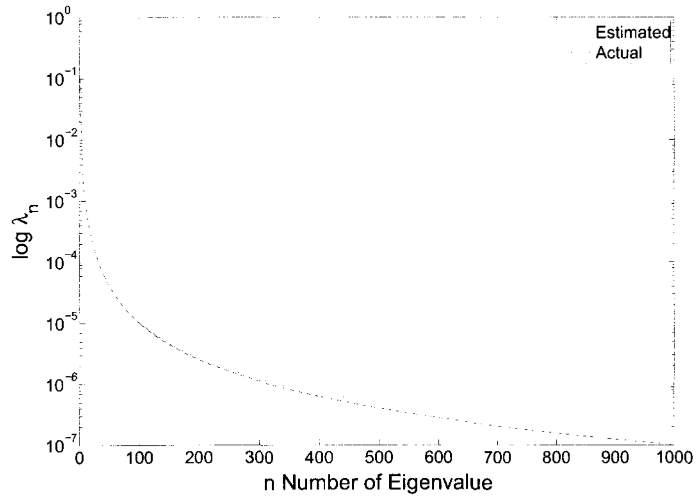


Figure B.4: Eigenvalues for the discretized and actual integral operator.

Since the resulting series are the sums of sines and cosines, the partial sums with few terms are rather smooth, but as more and more terms are added, the stochasticity of the actual random field (see figure B.5).

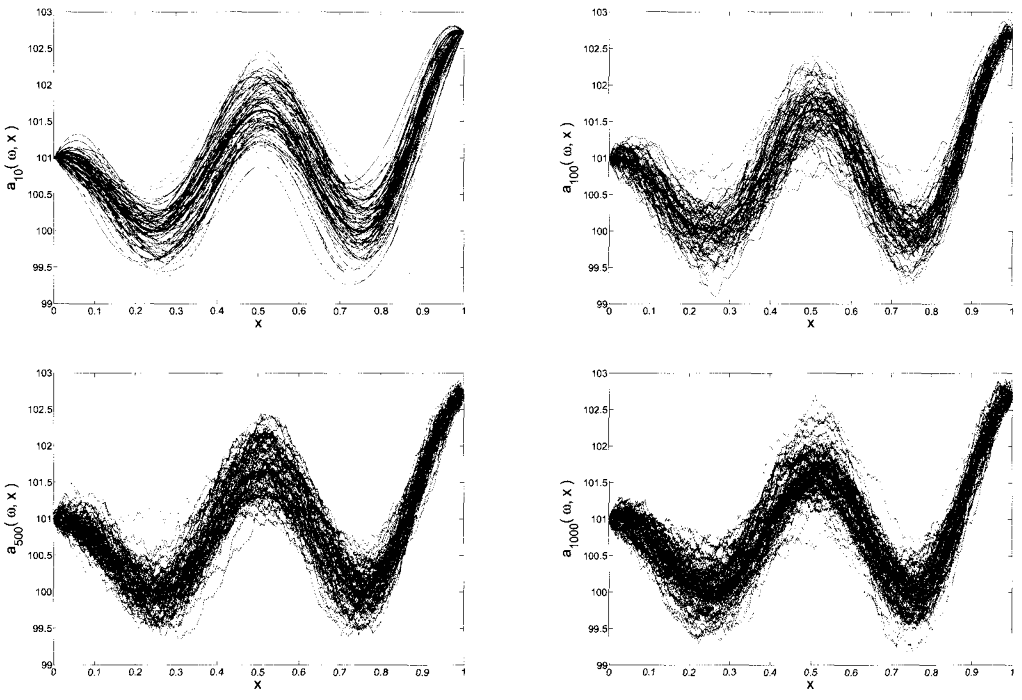


Figure B.5: 100 realizations of the partial sums of the KL expansion.

Bibliography

[1] M. Anitescu. Spectral finite-element methods for parametric constrained optimization problems. *SIAM J. Numer. Anal.*, 47(3):1739–1759, 2009.

[2] I. Babuška, F. Nobile, and R. Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM J. Numer. Anal.*, 45(3):1005–1034 (electronic), 2007.

[3] I. Babuška, R. Tempone, and G. E. Zouraris. Galerkin finite element approximations of stochastic elliptic partial differential equations. *SIAM J. Numer. Anal.*, 42(2):800–825 (electronic), 2004.

[4] I. Babuška, R. Tempone, and G. E. Zouraris. Solving elliptic boundary value problems with uncertain coefficients by the finite element method: the stochastic formulation. *Comput. Methods Appl. Mech. Engrg.*, 194(12-16):1251–1294, 2005.

[5] V. Barthelmann, E. Novak, and K. Ritter. High dimensional polynomial interpolation on sparse grids. *Adv. Comput. Math.*, 12(4):273–288, 2000. Multivariate polynomial interpolation.

[6] I. Das and J.E. Dennis. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM J. on Optimization*, 8(3):631–657, 1998.

[7] J. E. Dennis, Jr. and R. B. Schnabel. *Numerical Methods for Nonlinear Equations and Unconstrained Optimization.* Prentice-Hall, Englewood Cliffs, N. J, 1983. Republished as [8].

[8] J. E. Dennis, Jr. and R. B. Schnabel. *Numerical Methods for Nonlinear Equations and Unconstrained Optimization.* SIAM, Philadelphia, 1996.

[9] E. Emmerich. *Gewöhnliche und Operator-Differentialgleichungen Eine integrierte Einführung in Randwertprobleme und Evolutionsgleichungen für Studierende.* Vieweg Verlag, Braunschweig, 2004.

[10] G. B. Folland. *Real analysis.* Pure and Applied Mathematics (New York). John Wiley & Sons Inc., New York, second edition, 1999. Modern techniques and their applications, A Wiley-Interscience Publication.

[11] B. Ganapathysubramanian and N. Zabaras. Sparse grid collocation schemes for stochastic natural convection problems. *J. Comput. Phys.*, 225(1):652–685, 2007.

[12] T. Gerstner and M. Griebel. Numerical integration using sparse grids. *Numer. Algorithms*, 18(3-4):209–232, 1998.

[13] R. G. Ghanem and P. D. Spanos. *Stochastic finite elements: a spectral approach.* Springer-Verlag, New York, 1991.

[14] M. Heinkenschloss. Numerical solution of implicitly constrained optimization problems. Technical Report TR08–05, Department of Computational and Applied Mathematics, Rice University, Houston, TX 77005–1892, 2008.

[15] E. Hille. *Functional Analysis and Semi-Groups.* American Mathematical Society Colloquium Publications, vol. 31. American Mathematical Society, New York, 1948.

[16] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich. *Optimization with Partial Differential Equations*, volume 23 of *Mathematical Modelling, Theory and Applications.* Springer Verlag, Heidelberg, New York, Berlin, 2009.

[17] iB. Øksendal. *Stochastic differential equations.* Universitext. Springer-Verlag, Berlin, sixth edition, 2003. An introduction with applications.

[18] K. Karhunen. Über lineare Methoden in der Wahrscheinlichkeitsrechnung. *Ann. Acad. Sci. Fennicae. Ser. A. I. Math.-Phys.*, 1947(37):79, 1947.

[19] G. E. Karniadakis, C.-H. Su, D. Xiu, D. Lucor, C. Schwab, and R. A. Todor. Generalized polynomial chaos solution for differential equations with random inputs. Technical Report 2005-01, Seminar for Applied Mathematics, ETH Zurich, Zurich, Switzerland, 2005.

[20] C. T. Kelley. *Iterative Methods for Optimization.* SIAM, Philadelphia, 1999.

[21] P. D. Lax. *Functional Analysis.* John Wiley & Sons, New-York, Chicester, Brisbane, Toronto, 2002.

[22] M. Loève. Fonctions aléatoires de second ordre. *Revue Sci.*, 84:195–206, 1946.

[23] D. Lucor and G. E. Karniadakis. Adaptive generalized polynomial chaos for nonlinear random oscillators. *SIAM J. Sci. Comput.*, 26(2):720–735 (electronic), 2004.

[24] K. Marti. Stochastic optimization methods in optimal engineering design under stochastic uncertainty. *ZAMM Z. Angew. Math. Mech.*, 83(12):795–811, 2003.

[25] K. Marti. *Stochastic optimization methods.* Springer-Verlag, Berlin, 2005.

[26] F. Nobile, R. Tempone, and C. G. Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 46(5):2309–2345, 2008.

[27] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer Verlag, Berlin, Heidelberg, New York, second edition, 2006.

[28] Y. Notay. Flexible conjugate gradients. *SIAM J. Sci. Comput.*, 22(4):1444–1460 (electronic), 2000.

[29] E. Novak and K. Ritter. High-dimensional integration of smooth functions over cubes. *Numer. Math.*, 75(1):79–97, 1996.

[30] E. Novak and K. Ritter. Simple cubature formulas with high polynomial exactness. *Constr. Approx.*, 15(4):499–522, 1999.

[31] R. T. Rockafellar. Coherent approaches to risk in optimization under uncertainty. *Tutorials in Operations Research INFORMS*, pages 38–61, 2007.

[32] T. Roubíček. *Nonlinear partial differential equations with applications*, volume 153 of *International Series of Numerical Mathematics.* Birkhäuser Verlag, Basel, 2005.

[33] V. Schulz and C. Schillings. On the nature and treatment of uncertainties in aerodynamic design. *AIAA Journal*, 47(3):646–654, 2009.

[34] C. Schwab and R. A. Todor. Karhunen-Loève approximation of random fields by generalized fast multipole methods. *J. Comput. Phys.*, 217(1):100–122, 2006.

[35] S. A. Smoljak. Quadrature and interpolation formulae on tensor products of certain function classes. *Soviet Math. Dokl.*, 4:240–243, 1963.

[36] A. H. Stroud. Remarks on the disposition of points in numerical integration formulas. *Mathematical Tables and Other Aids to Computation*, 11(60):257–261, 1957.

[37] R. A. Todor and C. Schwab. Convergence rates for sparse chaos approximations of elliptic problems with stochastic coefficients. *IMA J. Numer. Anal.*, 27(2):232–261, 2007.

[38] G. W. Wasilkowski and H. Woźniakowski. Explicit cost bounds of algorithms for multivariate tensor product problems. *J. Complexity*, 11(1):1–56, 1995.

[39] C. Webster. *Sparse Grid Stochastic Collocation Techniques for the Numerical Solution of Partial Differential Equations with Random Input Data.* PhD thesis, Department of Mathematics and School of Computational Science, Florida State University, Tallahassee, FL, 2007.

[40] N. Wiener. The homogeneous chaos. *Amer. J. Math.*, 60:897–938, 1938.

[41] D. Xiu. Fast numerical methods for robust optimal design. *Engineering Optimization*, 40(6):489 – 504, 2008.

[42] D. Xiu and J. S. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM J. Sci. Comput.*, 27(3):1118–1139 (electronic), 2005.

[43] D. Xiu and G. E. Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. *J. Comput. Phys.*, 187(1):137–167, 2003.

[44] K. Yosida. *Functional Analysis.* Springer Verlag, Berlin, Heidelberg, New-York, sixth edition, 1980.