Parameter Dimension Reduction for Scientific Computing

by

Andrew Glaws

B.A., Vanderbilt University, 2012

M.S., Virginia Polytechnic Institute and State University, 2014

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

2018

This thesis entitled:
Parameter Dimension Reduction for Scientific Computing
written by Andrew Glaws
has been approved for the Department of Computer Science

Prof. Paul G. Constantine

Prof. Jed Brown

Prof. Claire Monteleoni

Prof. Xiao-Chuan Cai

Date

Glaws, Andrew (Ph.D., Computer Science)

Parameter Dimension Reduction for Scientific Computing

Thesis directed by Prof. Paul G. Constantine

Advances in computational power have enabled the simulation of increasingly complex physical systems. Mathematically, we represent these simulations as a mapping from inputs to outputs. Studying this map—e.g., performing optimization, quantifying uncertainties, etc.—is a critical component of computational science research. Such studies, however, can suffer from the *curse of dimensionality*—i.e., an exponential increase in computational cost resulting from increases in the input dimension. *Dimension reduction* combats this curse by determining relatively important (or unimportant directions) in the input space. The problem is then reformulated to emphasize the important directions while the unimportant directions are ignored. Functions that exhibit this sort of low-dimensional structure through linear transformations of the input space are known as *ridge functions*. Ridge functions appear as the basic components in various approximation and regression techniques such as neural networks, projection pursuit regression, and multivariate Fourier series expansion. This work focuses on how to discover, interpret, and exploit ridge functions to improve scientific computing.

In this thesis, we examine relationships between the ridge recovery technique *active subspaces* and the physically-motivated *Buckingham Pi Theorem* in *magnetohydrodynamics (MHD)* models. We show that active subspaces can recover known unitless quantities from MHD such as the Reynolds and Hartmann numbers through a log transformation of the inputs. We then study the relationship between ridge functions and statistical dimension reduction for regression problem—i.e., *sufficient dimension reduction (SDR)*. We show that a class of SDR methods called *inverse regression methods* provide a gradient-free approach to ridge recovery when applied to deterministic functions. We examine the numerical properties of these methods as well as their failure cases. We also introduce novel algorithms for computing the underlying population matrices of these inverse regression methods using classical iterative methods for generating orthogonal polynomials. Finally, we introduce a new method for cheaply constructing accurate polynomial surrogates on one-dimensional ridge functions by obtaining generalized Gauss-Christoffel quadrature rules with respect to the marginal density on the one-dimensional ridge subspace.

Dedication

In loving memory of my grandfather, Walter Glaws, who always impressed upon me the importance of my studies.

Acknowledgements

There are many people, without whom, this dissertation would not have been possible. Any proper acknowledgement of the support I have received from others must begin with my adviser, Paul Constantine. Paul has provided me with invaluable guidance in developing the research done for this dissertation. He has also given me the freedom to explore ideas, make mistakes, and grow as a researcher. I am forever grateful to have him as a mentor and a friend. I would also like to thank the many other experienced researcher who have all contributed to this work through various conversations and collaborations—Jed Brown, Claire Monteleoni, Xiao-Chuan Cai, Mike Mozer, John Shadid, Tim Wildey, Dennis Cook, and Luis Tenorio.

I would like to acknowledge my colleagues—Zach Grey, Jeff Hokanson, Izzy Aguiar, Paul Diaz, Nick Fisher, Mike Schmidt, Caitlyn Hannum, Nora Stack, and Kai Bartlette—whose discussions and insights have helped to elevate this work. More importantly, their friendship has kept me refreshed and excited to come into the office everyday. I am lucky to have such an amazing group of people in my life.

I am thankful to my family who has supported me over the years. From an early age, my parents have nurtured my passion for math and science, and I am grateful for all of their sacrifices that have enabled me to pursue my dreams. I am proud to be their son. Lastly, I want to thank my incredible wife, Andrea, for her endless love and encouragement during this journey. She has been by my side to celebrate the good times and to lift me up when things got difficult. I am truly fortunate to have her as a partner, and I look forward to our next adventure together.

Contents

Tables

Figures

Chapter 1

Introduction

The rapid growth of computing power has enabled the simulation of complex physical processes by models that more accurately reflect reality. Mathematically, we treat these models as a deterministic mapping from a set of inputs to some output of interest,

$$y = f(\mathbf{x}), \qquad \mathbf{x} \in \mathbb{R}^m, \quad y \in \mathbb{R}, \tag{1.1}$$

where $\mathbf{x}$ contains a vector of continuous physical inputs, y is a scalar-valued output or quantity of interest, and f represents the underlying computational model. It is common to treat f as a deterministic function, since a bug-free computational model produces the same outputs given the same inputs; in other words, there is no random noise in simulation outputs. This perspective is the de facto setup in the statistics subfield of *computer experiments* [100, 76, 102].

We assume (1.1) is accompanied by a known input probability measure $\pi_{\mathbf{x}}$. This measure treats the model inputs as random variables to represent aleatory uncertainty. One common choice is to assume a uniform weighting over a valid set of parameter ranges for each of the inputs. In general, we assume that the inputs are uncorrelated and standardized such that

$$\mathbb{E}\left[\mathbf{x}\right] = \mathbf{0} \quad \text{and} \quad \mathrm{Cov}\left[\mathbf{x}\right] = \boldsymbol{I}. \tag{1.2}$$

We can apply relatively straightforward transformations of the inputs to ensure (1.2) holds if needed. We revisit and justify this idea later in the thesis. This is the standard problem setup in the field of *uncertainty quantification* [109, 117, 55].

One means by which we can increase the complexity of a given model is by increasing the dimension of the model—i.e., the number of inputs m. This increases the size of the input space, potentially leading to better understanding of the physical system, improved optimal solutions, and other benefits. However, this increase in dimension can also lead to strange and unintuitive behavior. This idea is generally referred to as the *curse of dimensionality* and can have different interpretations in different contexts.

1.1 The curse of dimensionality

The curse of dimensionality was first introduced 1961 by Richard Bellman in reference to combinatorial optimization in high dimensions [7]. This phrase has been used to describe a wide range of issues arising from the nonintuitive behaviors of high-dimensional spaces. In general, the curse of dimensionality refers to an exponential increase in the difficulty of applying various machine learning techniques as the dimension of the input space increases [43]. Despite the significant strides made in computational power in recent years, we continue to find ourselves dealing with issues related to dimensionality. Pedro Domingos states that "after overfitting, the biggest problem in machine learning is the curse of dimensionality" [42].

In statistics, the curse of dimensionality can make adequately sampling high-dimensional spaces very difficult, even in seemingly simple cases [103, Chap. 7]. Consider a standard multivariate Gaussian distribution in m dimensions, $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Typically, we think of such distributions as being highly intuitive, with the mean in the center of the density and an exponential decay as we move outward. When visualizing the Gaussian density function (note that the process of visualization implies $m = 1$ or $m = 2$), we see much of the function's "mass" is concentrated around the mean. This would seem to suggest that a high percentage of random samples drawn from this distribution should be near the mean. However, this does not hold true as we increase the dimension. Figure 1.1 shows the percentage of random samples that fall within a unit ball around the mean. There is an exponential decay in the number of points within a radius $r = 1$ of the center of the distribution.

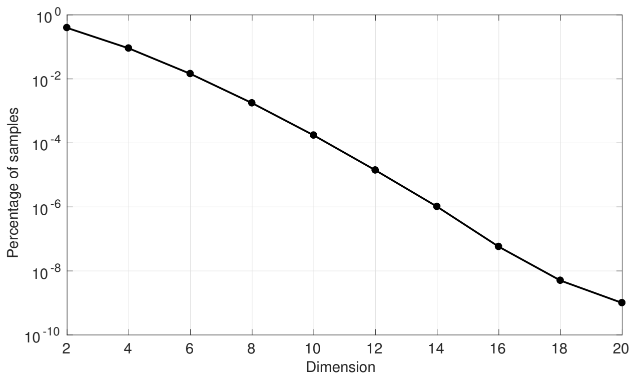


Figure 1.1: A study of the percentage of random samples that fall with the unit ball around the mean of an m-dimensional multivariate Gaussian distribution.

High-dimensions can also inhibit clustering-based algorithms for unsupervised machine learning, such as the k-nearest neighbor algorithm [69, Chap. 2]. This algorithm infers some characteristics regarding a particular point of interest by considering the k closest points in m-dimensional space to that point. The fundamental assumption of this algorithm is that nearby points are more likely to share similar characteristics than far away points. Such an assumption seems reasonable when considered for $m = 2$ or $m = 3$ dimensions. However, this breaks down as the dimension increases [8]. Consider a collection of N points uniformly sampled from the $[-1, 1]^m$ hypercube. For each point, we consider the ratio of the distance to its farthest neighbor over the distance to its nearest neighbor,

$$\frac{\max_{i \neq j} ||\mathbf{x}_i - \mathbf{x}_j||}{\min_{i \neq j} ||\mathbf{x}_i - \mathbf{x}_j||}. \tag{1.3}$$

Figure 1.2 contains the average of this ratio for fixed N with increasing dimension. Notice that this ratio tends towards one as m increases. This implies that the difference between nearby points and far-away points vanishes in high dimensions. This undermines the fundamental assumption behind the k-nearest neighbor algorithm.

In computational science modeling, the curse of dimensionality typically appears in the context of high-dimensional function approximation. These models can be expensive to evaluate, often requiring us to numerically approximate a highly-coupled system of partial differential equations.

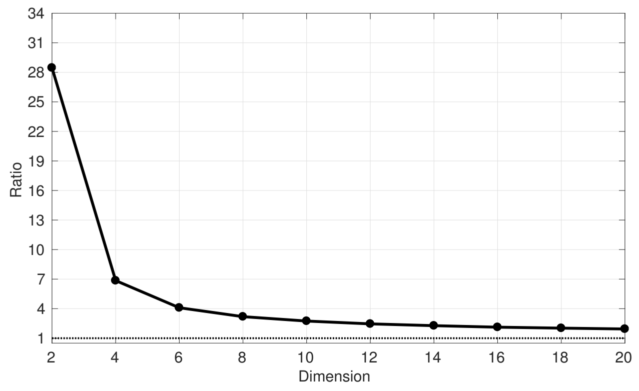


Figure 1.2: A study of the average ratio of distance to the farthest point over the distance to the nearest point for a fixed number of samples as dimension increases.

Studying these models by performing uncertainty quantification or optimization may require multiple evaluations at various inputs. Instead, we may choose to construct a cheap approximation of the computational model. These approximations appear under several names: response surfaces [90, 75], surrogate models [98, 5], metamodels [126], and emulators [12]. Methods for building such surrogate models include radial basis functions [47], splines [125], and polynomial approximations (referred to as polynomial chaos expansions in the uncertainty quantification literature) [132]. Once we have constructed the surrogate model, we can perform the study on it much more cheaply than on the original function.

Unfortunately, if we are restricted to only classical smoothness assumptions—e.g., Lipschitz continuity, continuous differentiability, etc.—then building surrogate approximations suffers from the curse of dimensionality [119, 129]. Roughly speaking, the number of function evaluations required to construct a surrogate of fixed accuracy grows exponentially with the dimension—i.e., the number of inputs to the function. David Donoho acknowledges this fact when he refers to the curse of dimensionality as "the apparent intractability of accurately approximating a general high-dimensional function" [43]. This perspective on the curse of dimensionality is most closely aligned with the work presented in this thesis.

To demonstrate the type of intractability Donoho mentions, we consider the following illustrative example. We wish to sufficiently sample an m-dimensional input space in order to build an accurate surrogate of a computational model. To do this, we must evaluate the function at $N = 6$ unique input locations per dimension and each model evaluation takes $T = 1$ second to execute. Figure 1.3 shows the exponential growth in the computational time required to run the required number of the simulations as the dimension increases. We rapidly approach infeasible computation times. Suppose we develop a better method for searching the input space that only requires $N = 3$ model evaluations per dimension. Or alternatively, suppose we obtain a more powerful computer that can evaluate the model in $T = 1$ milliseconds. The computational times for these studies as a function of dimension are shown in red and blue, respectively. Note that these improvements reduce the overall computation time; however, the costs are still growing exponentially with dimension and quickly become infeasible. Thus, the best approach for combating the curse of dimensionality is to reduce the dimension of the input space.

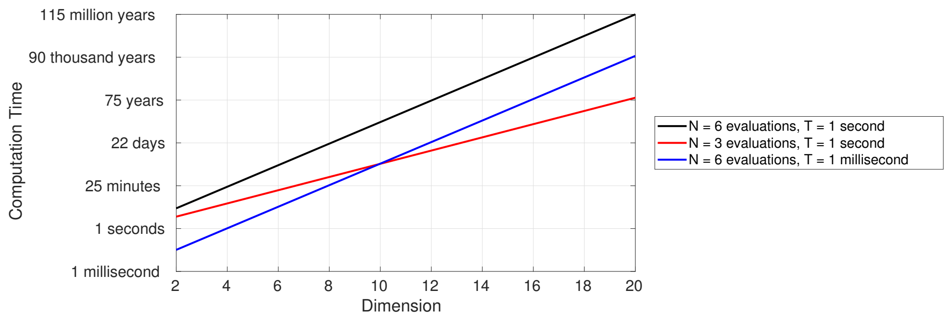


Figure 1.3: A study of the computation time to perform a study of an m-dimensional model requiring N model evaluations per dimension with each evaluation taking time T to execute.

1.2 Dimension reduction

Dimension reduction methods attempt to reduce the dimension of some space or set of data while retaining various important characteristics. There exist a wide range of both linear [37] and nonlinear [78] techniques for performing dimension reduction with the main differences between them being the specific characteristics they deem important. Additionally, we can consider dimension reduction in the context of unsupervised and supervised problems.

Unsupervised methods for dimension reduction consider relationships among the various parameters. For example, principle component analysis (PCA) seeks out linear transformations of the parameters that maximizes the variance [74]. This can be expressed in terms of the optimization problem

$$\begin{aligned} &\underset{\mathbf{a}\in\mathbb{R}^m}{\operatorname{argmax}}\ \mathbf{a}^\top \operatorname{Cov}[\mathbf{x}]\,\mathbf{a}, \\ &\text{subject to}\ \ \mathbf{a}^\top\mathbf{a} = 1. \end{aligned} \tag{1.4}$$

The solution to (1.4) is the normalized eigenvector of the covariance matrix associated with the largest eigenvalue. This can be generalized to n dimensions by projecting the parameters along the n dominant eigenvectors—i.e., those associated with the largest eigenvalues. It should be noted that PCA is characterized by its definition of what is important—i.e., the covariance structure between parameters. It is not characterized by the use of the eigendecomposition (or singular value decomposition (SVD)) in discovering these important direction. Consider multidimensional scaling (MDS) which attempts to perform dimension reduction while preserving distances been the various data points [10]. This is done by constructing a squared distance matrix and converting it into a Gram matrix. The data is then projected along the dominant eigenvalues of this matrix. The term *spectral dimension reduction* refers to the generalized approach of constructing a similarity matrix from the data and computing its eigendecomposition to obtain appropriate linear transformations of the parameters [116].

Recall that the fundamental problem statement for this work is given by (1.1). Thus, we are primarily interested in dimension reduction that accounts for the deterministic relationship between the m-dimensional input and the scalar-valued output—i.e., dimension reduction in the supervised framework. Additionally, recall from (1.2) that we assume standardized and uncorrelated inputs. This suggests that the unsupervised dimension reduction methods described above will not be beneficial as there are no relationships between the various inputs that can be exploited.

One natural approach to perform this type of dimension reduction is to examine the relative impact of each input on the output. *Sensitivity analysis* is a field of research concerned with quantifying the importance of each input with respect to changing the output [101]. Note again that the idea of importance is defined by the specific techniques employed for performing the sensitivity analysis. For example, Sobol indices determine how much different inputs or interactions of inputs account for the total variance in the output [111]. Alternatively, the Morris method assesses sensitivities by using approximated partial derivatives with respect to each of the inputs [88]. Once the sensitivities have been quantified, inputs that have relatively little impact on the output can then be fixed at a nominal value, reducing the dimension of the function by exploiting coordinate-aligned anisotropy.

We can generalize the idea of coordinate-aligned dimension reduction via sensitivity analysis by considering arbitrary directions within the input space. One such method for accomplishing this is active subspaces [18]. Active subspaces determines a set of directions in the input space along which $f(\mathbf{x})$ is changing the most on average. We can then disregard the orthogonal directions along which $f(\mathbf{x})$ is changing less on average. This is done by examining the eigenspace of the matrix

$$\boldsymbol{C} \;=\; \int \nabla f(\mathbf{x})\, \nabla f(\mathbf{x})^\top \,\mathrm{d}\pi_{\mathbf{x}}(\mathbf{x}), \tag{1.5}$$

where $\nabla f(\mathbf{x}) \in \mathbb{R}^m$ is the gradient of (1.1) with respect to the inputs and $\pi_{\mathbf{x}}$ is the input probability measure introduced at the beginning of this chapter. We examine active subspaces in more detail in Chapter 2. Other approaches to this type of supervised dimension reduction solve an optimization problem over the space of possible directions [23, 70] or reformulate the problem in terms of a compressed sensing problem [15, 123, 118]. This idea of finding relatively important/unimportant directions over the input space is formalized by the concept of *ridge functions* [96]. In a sense, all of these dimension reduction techniques can be considered methods for finding ridge structure in computational models.

1.3 Ridge functions

In this thesis, we approach supervised linear dimension reduction from the perspective of ridge functions [96]. A function $f : \mathbb{R}^m \to \mathbb{R}$ is a ridge function if there exists a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$

with $n < m$ and a function $g : \mathbb{R}^n \to \mathbb{R}$ such that

$$y = f(\mathbf{x}) = g(\mathbf{u}), \tag{1.6}$$

where $\mathbf{u} = \boldsymbol{A}^\top \mathbf{x} \in \mathbb{R}^n$. Thus, a ridge function is nominally a function of m inputs, but it is intrinsically a function of $n < m$ derived inputs. The columns of $\boldsymbol{A}$ are the *directions* of the ridge function and $g : \mathbb{R}^n \to \mathbb{R}$ is the *ridge profile*. The term ridge function sometimes refers specifically to the $n = 1$ case, while $n > 1$ is called a generalized ridge function [96]. We do not distinguish between the $n = 1$ and $n > 1$ cases and refer to all such functions as ridge functions for convenience. Figure 1.4 provides an illustrative example of a ridge function.

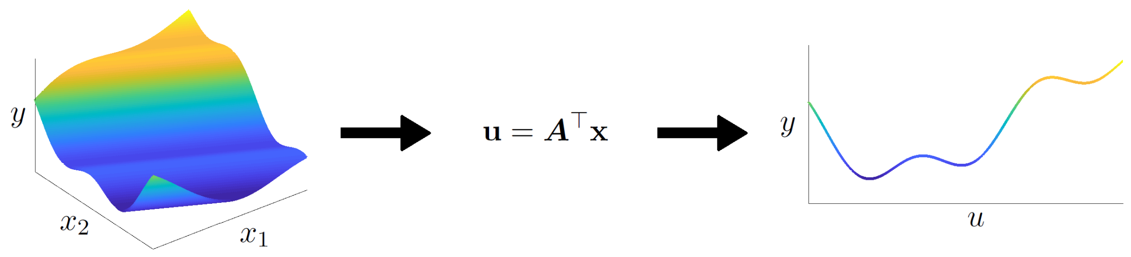


Figure 1.4: An illustrative example of a ridge function.

Ridge functions first appeared under the name *plane waves* in 1955 in the context of solutions to hyperbolic differential equations [72, 35]. The term ridge function was introduced in 1975 in relation to computerized tomography [86]. Ridge functions are the fundamental component of various regression and approximation techniques. For example, single index models assume a regression model comprised of a one-dimensional ridge function [68, Chap. 6] and projection pursuit regression generalizes this to a sum of n one-dimensional ridge functions, $\sum_{i=1}^{n} g_i(\mathbf{a}_i^\top \mathbf{x})$, where each g_i is a spline or some other nonparametric model [51]. The nodes of a multilayer feedforward neural network use functions of the form $\sigma(\boldsymbol{W}^\top \mathbf{x} + \mathbf{b})$, where $\boldsymbol{W}$ is a matrix of weights from the previous layer of neurons, $\mathbf{b}$ is a bias term for the model, and $\sigma(\cdot)$ is an activation function [66]. This activation function is essentially a ridge function of the m inputs from the previous layer.

Ridge functions are good low-dimensional representations of functions that exhibit off-axis anisotropic dependence on the inputs since they are constant along directions orthogonal to the ridge directions—i.e., the columns of $\boldsymbol{A}$. Consider a vector $\mathbf{w} \in \text{null}(\boldsymbol{A}^\top)$. Then, for any $\mathbf{x} \in \mathbb{R}^m$,

$$f(\mathbf{x}+\mathbf{w}) \;=\; g(\boldsymbol{A}^\top(\mathbf{x}+\mathbf{w})) \;=\; g(\boldsymbol{A}^\top\mathbf{x}+\mathbf{0}) \;=\; g(\boldsymbol{A}^\top\mathbf{x}) \;=\; f(\mathbf{x}). \tag{1.7}$$

Therefore, we need not consider the $m-n$ directions in the input space that are orthogonal to the columns of $\boldsymbol{A}$. Reflecting on Figure 1.3, this can result in an exponential savings in the computational costs of studying $f(\mathbf{x})$.

We may also be interested in functions that are well-approximated by a ridge function,

$$y \;=\; f(\mathbf{x}) \;\approx\; g(\mathbf{u}). \tag{1.8}$$

The best L^2 approximation of f by g is the expected value of the output conditioned on $u = \boldsymbol{A}^\top\mathbf{x}$ [96, Ch. 8]. That is,

$$g(\mathbf{u}) \;=\; \mathbb{E}\left[f(\mathbf{x})\,\middle|\,\mathbf{u}=\boldsymbol{A}^\top\mathbf{x}\right]. \tag{1.9}$$

We formalize the phrase "well-approximated" in this context to refer to the expected conditional variance,

$$\mathbb{E}\left[\text{Var}\left[f(\mathbf{x})\,\middle|\,\mathbf{u}=\boldsymbol{A}^\top\mathbf{x}\right]\right]. \tag{1.10}$$

In practice, we can approximate 1.10 using a hit-and-run algorithm for drawing conditional samples from the input space [110]. We explore this idea further in upcoming chapters. Note that the conditional variance and expectation in (1.10) are computed with respect to the input probability measure $\pi_{\mathbf{x}}$. This implies that we can allow the variance of $f(\mathbf{x})$ conditioned of $\mathbf{u} = \boldsymbol{A}^\top\mathbf{x}$ to be very large, provided that this occurs in regions of the domain where the probability density is small. In this way, we take a global perspective on ridge approximation with $\pi_{\mathbf{x}}$ playing an important role in determining how well-approximated a function is by a ridge function.

The problem of discovering $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ such that (1.6) or (1.8) holds is known as *ridge recovery* or *ridge approximation*, respectively. Recall the claim from the previous section that active subspaces addresses the ridge recovery problem. In the case of (1.6), active subspaces perfectly recovers the ridge directions, while in the case of (1.8), active subspaces minimizes an upper bound on the expected conditional variance from (1.10) [18]. We formalize the ridge recovery problem here.

Problem 1 (Ridge recovery)**.** *Given a queryable deterministic function $f : \mathbb{R}^m \to \mathbb{R}$ and an input probability measure $\pi_{\mathbf{x}}$, find $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ with $n < m$ such that*

$$f(\mathbf{x}) \;=\; g(\boldsymbol{A}^\top \mathbf{x}) \tag{1.11}$$

for some $g : \mathbb{R}^n \to \mathbb{R}$.

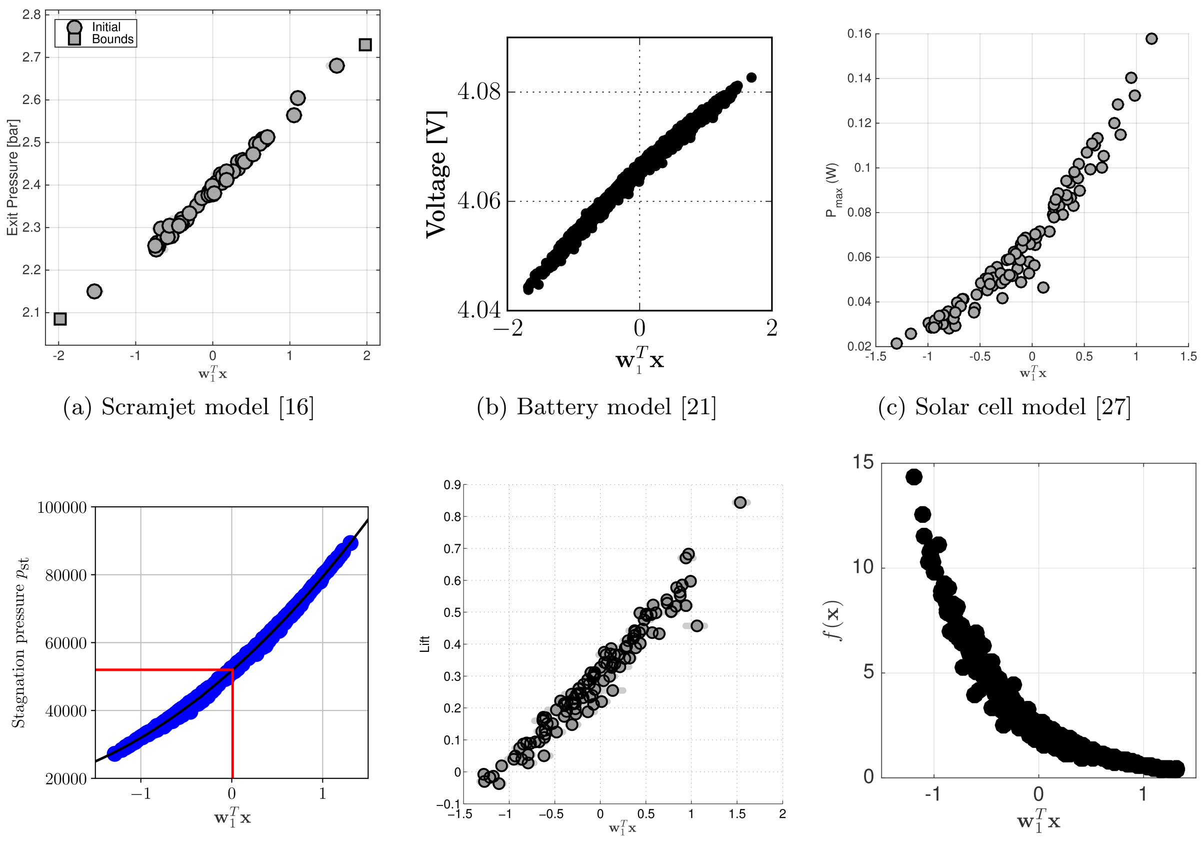
Note that principle component analysis (PCA) from Section 1.2 does not address the ridge recovery problem. PCA only considers covariant relationships between the various inputs in determining what direction are deemed important.

Note that the ridge directions—i.e., the columns of $\boldsymbol{A}$—are not unique in defining a ridge function. Consider $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ with full rank. Then,

$$f(\mathbf{x}) \;=\; g(\boldsymbol{A}^\top \mathbf{x}) \;=\; g(\boldsymbol{B}^{-1}\boldsymbol{B}\boldsymbol{A}^\top \mathbf{x}) \;=\; \tilde{g}(\boldsymbol{B}\boldsymbol{A}^\top \mathbf{x}) \;=\; \tilde{g}(\tilde{\boldsymbol{A}}^\top \mathbf{x}), \tag{1.12}$$

implying that the column space of $\boldsymbol{A}$ is the critical component in defining a ridge function. This motivates a class of ridge recovery methods that solve an optimization problem over the Grassmann manifold of n-dimensional subspaces of $\mathbb{R}^m$ [23, 70]. Additionally, this allows us to assume without loss of generality that the columns of $\boldsymbol{A}$ are orthonormal, such that $\boldsymbol{A}^\top \boldsymbol{A} = \boldsymbol{I} \in \mathbb{R}^{n \times n}$. Under this assumption, the transformation of the inputs in (1.6) is a rigid rotation of m-dimensional space followed by a reduction of $m - n$ unimportant directions.

In the space of general functions $f : \mathbb{R}^m \to \mathbb{R}$, assumptions of approximate or exact ridge structure might seem restrictive. For sure, it is easy to write down an example of a multivariate function that does not exhibit low-dimensional ridge structure. However, computational models for simulating physics-based phenomena often exhibit the type of anisotropy characteristic of ridge functions. Recent work has revealed ridge function structure in a variety of physics-based models, including lithium ion battery models [21], car aerodynamics models [93], integrated hydrologic models [71], hypersonic scramjet designs [16], among others [27, 57, 34]. Figure 1.5 shows examples of this ridge structure. The plots in this figure are known as *shadow plots* or *sufficient summary plots* [30]. These are one-dimensional scatter plots of the model output against the inputs projected along the most important direction in the input space. All of the shadow plots in this figure depict relatively strong one-dimension trends, despite the underlying models depending on higher-dimensional sets of inputs. These plots are powerful tools for qualitatively assessing how well a given model can be approximated by a ridge function. Deviations from the one-dimensional trend are qualitative examples of the conditional variance in (1.10).



(a) Scramjet model [16]

(b) Battery model [21]

(c) Solar cell model [27]

(d) Reentry vehicle model [34]

(e) Transonic airfoil model [18]

(f) MHD generator model [62]

Figure 1.5: One-dimensional shadow plots of data derived from computational models across applications.

There are several possible explanations for the abundance of ridge structure in computational science models. First, the models are physically-motivated. This implies a fundamental relationship between the inputs and the outputs that does not exist for general mathematical functions. We explore how this physically-driven relationship can induce ridge structure in more detail in Chapter 2. Another reason why the assumption of ridge structure in computational science models is nonrestrictive is that the models are typically built to study a very specific phenomenon, leading to modeling decisions that restrict the inputs to a relatively small domain where the model is applicable. This is reflected in the input probability measure $\pi_{\mathbf{x}}$, which is given along with the model $f(\mathbf{x})$. For example, we typically use different modeling approaches for laminar flow versus turbulent flow. Within each of these regimes, we might find different kinds of ridge structure; however, it would not make sense to perform ridge recovery for both regimes simultaneously since the underlying models are different.

1.4 Outline and contributions

This thesis is organized into four main chapter, each focusing on a specific project. These projects explore a variety of key research directions related to ridge recovery and approximation. In Chapter 2, we examine the interpretability of dimension reduction via ridge functions in the context of physically-motivated computer models. This work is based on [62]. We use the gradient-based active subspace analysis as a framework for studying ridge functions [18]. We look for low-dimensional structure in magnetohydrodynamics models that depend on several physical inputs [36]. Linear dimension reduction defines a reduced set of inputs by linear combinations of the original inputs. The contributions in this work consider how to understand this type of dimension reduction by connecting it to classical dimensional analysis. This work also suggests possible nonlinear transformations of the input space that can induce ridge structure in computational models.

Chapter 3 considers the application of sufficient dimension reduction (SDR)—a theoretical framework for subspace-based dimension reduction in statistical regression—to deterministic functions. We prove that, in this context, SDR provides a novel subspace-based perspective to addressing the ridge recovery problem (Problem 1). This work is based on [61]. One of the main contributions of this work is to draw formal connections between these two previously disjoint fields of research. We rigorously study two SDR algorithms—sliced inverse regression (SIR) and sliced average variance estimation (SAVE)—as ridge recovery methods. These methods look for low-dimensional structure by approximating statistical moments of key components of $f(\mathbf{x})$. This approach is significantly cheaper than approximating gradients (a necessary step for active subspace analysis). As part of this investigation, we perform rigorous numerical analysis of the SIR and SAVE algorithms that provides insight into the convergence and accuracy of these methods for discovering ridge subspaces. Furthermore, this works enables the development of improved algorithmic approaches for computing the underlying population matrices of SIR and SAVE.

The work in Chapter 4 builds upon Chapter 3 by introducing novel methods for computing the underlying population matrices of SIR and SAVE algorithms. This work is based on [59]. The new algorithms replace the crude slice-based approximations of SIR and SAVE with orthonormal polynomial expansions and high-order Gauss-Christoffel quadratures. We refer to these algorithms as Lanczos-Stieltjes inverse regression (LSIR) and Lanczos-Stieltjes average variance estimation (LSAVE) since they employ the Lanczos iteration as a discrete approximation to the Stieltjes procedure. These algorithms are shown to perform as well as the best-case scenario for their slice-based counterparts while also enabling the use of more accurate designs—e.g., tensor product quadrature rules—on the input space.

Finally, Chapter 5 presents a novel approach to constructing polynomial approximations of one-dimensional ridge functions. This work is based on [60]. The linear transformation of the inputs in a ridge function induces a probability measure on the one-dimensional ridge subspace. The new algorithms approximate this measure and use several of the same numerical techniques from Chapter 4 for constructing generalized quadrature rules. We perform several numerical studies of the convergence and behavior of the introduced methodologies and compare them to existing methods for constructing ridge function approximations.

Chapter 2

Ridge functions and dimensional analysis in magnetohydrodynamics

Using ridge functions for dimension reduction can lead to questions regarding the interpretability of the reduced inputs, particularly when applied to physically-motivated computational models. While useful as mathematical tools, the linear combinations of physical inputs in a ridge function can be difficult to understand from an engineering perspective. In this chapter, we connect ridge functions to physically-motivated dimension reduction based on dimensional analysis, providing useful insights into how the transformed inputs of ridge functions relate to the physical model in question. Recent work in the statistics literature has recognized the importance of dimensional analysis for inference and experimental design when data are derived from physical systems [108, 84, 107]; moreover, it is suggested that dimensional analysis is underutilized by statisticians [4]. For this work, we use active subspaces [18] as the mathematical formulation for exploring ridge structure in functions.

Magnetohydrodynamics (MHD)—an area of physics concerned with electrically conducting fluids—serves as the framework for making these connections [36]. MHD models are found in disparate fields from geophysics to fusion energy. The US Department of Energy's investment in MHD for power generation with low emissions generators dates back to the 1960s. Interest waned in the mid 1990s after a proof-of-concept program revealed several technical challenges to scaling and integrating MHD-based components into a practical power generator [124]. However, in the last two decades, several of these challenges have been addressed via other investments; a recent workshop at the National Energy Technology Laboratory brought together current MHD researchers from labs, academia, and industry to assess the state of the art in MHD power generation. In particular, given two decades of supercomputing advances, the workshop report calls for improved simulation tools that exploit modern supercomputers to build better MHD models and integrate them into full scale generator designs [92]. Recent work has developed scalable simulations for resistive MHD models [85, 104, 105, 112]. To incorporate such simulations into generator design, a designer must understand the sensitivities of model predictions to changes in model input parameters. To address this, MHD simulation codes with built-in adjoint capabilities have been developed that enable the computation of derivatives of output quantities of interest with respect to input parameters [106].

2.1 Dimension reduction methods

Recall from Section 1.1, that we represent a given computational science model mathematically by a deterministic function mapping a vector of m physically-motivated inputs to a scalar-valued output,

$$y \;=\; f(\mathbf{x}), \qquad y \in \mathbb{R}, \qquad \mathbf{x} \in \mathbb{R}^m. \tag{2.1}$$

We assume the physical model is accompanied by a known input probability measure $\pi_{\mathbf{x}}$ that represents uncertainty or variability in the model inputs. In the previous chapter, we discussed how ridge functions provide a framework for reducing the dimension in a way that takes advantage of structure in $f(\mathbf{x})$ from (2.1).

We consider two types of dimension reduction here. The first is dimensional analysis, which is a mature tool for reducing the number of variables in a physical system by examining the quantities' units. The second is active subspaces, which is a mathematical approach to ridge recovery and ridge approximation in functions. The former is analytical while the latter is computational. In other words, dimensional analysis follows from the quantities' units and can typically be performed without the aid of a computer in small systems. In contrast, the active subspace is estimated by evaluating the function and its gradient at several values of $\mathbf{x}$. We follow the linear algebra-based presentation of dimensional analysis from [11, Chapter 4], which enables a precise comparison with active subspaces in Section 2.1.3.

2.1.1 Dimensional analysis and Buckingham Pi Theorem

Dimensional analysis enables dimension reduction by examining the physical units of the output and inputs. The central result in dimensional analysis—the Buckingham Pi Theorem [6, Chapter 1]—states that the original model (2.1) can be written in a unitless form as

$$\Pi \;=\; \tilde{f}(\boldsymbol{\Pi}) \tag{2.2}$$

where $\Pi \in \mathbb{R}$ and $\boldsymbol{\Pi} \in \mathbb{R}^n$ denote the unitless output and inputs. The number of unitless inputs in (2.2) cannot be more than the number of inputs in (2.1), and often, it will be less (i.e., $n \leq m$). Thus, dimensional analysis and Buckingham Pi facilitate dimension reduction of the model. This section focuses on the development of the unitless quantities, Π and $\boldsymbol{\Pi}$.

To apply dimensional analysis, we assume (2.1) to be a physical law, meaning that the output and inputs are accompanied by physical units. These units are derived from $k \leq m$ base units, denoted generically as $L_1, \ldots, L_k$, which are a subset of the seven SI units; Table 2.1 shows the SI base units. Note that if k is larger than m, then there are more fundamental units than needed to describe the system [6].

Table 2.1: International system of units (SI). Table taken from [91].

Base quantity	Name	Symbol
length	meter	m
mass	kilogram	kg
time	second	s
electric current	ampere	A
thermodynamic temperature	kelvin	K
amount of a substance	mole	mol
luminous intensity	candela	cd

The unit function of a quantity, denoted by square brackets, returns the units of its argument. For example, if y is a velocity, then $[y] = \mathrm{m} \cdot \mathrm{s}^{-1}$. If a quantity is unitless, then its unit function returns 1. We can generically write the units of $\mathbf{x} = [x_1, \ldots, x_m]^\top$ and y as

$$[x_j] = \prod_{i=1}^{k} L_i^{d_{i,j}}, \qquad [y] = \prod_{i=1}^{k} L_i^{u_i}. \tag{2.3}$$

In words, the units of each physical quantity can be written as a product of powers of the k base units. To derive the unitless quantities from (2.2), define the $k \times m$ matrix $\boldsymbol{D}$ and the k-vector $\mathbf{u}$ as

$$\boldsymbol{D} = \begin{bmatrix} d_{1,1} & \cdots & d_{1,m} \\ \vdots & \ddots & \vdots \\ d_{k,1} & \cdots & d_{k,m} \end{bmatrix}, \qquad \mathbf{u} = \begin{bmatrix} u_1 \\ \vdots \\ u_k \end{bmatrix}, \tag{2.4}$$

which contain the powers from the quantities' units in (2.3). Assume that $\operatorname{rank}(\boldsymbol{D}) = r$. Let $\mathbf{v} = [v_1, \ldots, v_m]^\top$ satisfy $\boldsymbol{D}\mathbf{v} = \mathbf{u}$, and let $\boldsymbol{U} \in \mathbb{R}^{m \times n}$ be a basis for the null space of $\boldsymbol{D}$, i.e., $\boldsymbol{D}\boldsymbol{U} = \mathbf{0} \in \mathbb{R}^{k \times n}$, where $n = m - r$. If $r = m$, then dimensional analysis does not result in dimension reduction. Note that the elements of $\mathbf{v}$ and $\boldsymbol{U}$ are not unique. We can construct the unitless quantity of interest Π as

$$\Pi \;=\; y \prod_{i=1}^{m} x_i^{-v_i}. \tag{2.5}$$

The unit function, defined above, then returns $[\Pi] = 1$ by construction. We similarly construct unitless parameters Π_j as

$$\Pi_j \;=\; \prod_{i=1}^{m} x_i^{u_{i,j}}, \qquad j = 1, \ldots, n, \tag{2.6}$$

where $u_{i,j}$ is the (i, j) element of $\boldsymbol{U}$.

The Buckingham Pi Theorem [6] states that a physical law (2.1) can be written in unitless form

$$\Pi \;=\; \tilde{f}(\boldsymbol{\Pi}), \qquad \boldsymbol{\Pi} = \begin{bmatrix} \Pi_1 & \cdots & \Pi_n \end{bmatrix}^\top. \tag{2.7}$$

The number of inputs in the unitless form of the physical law (2.2) is $n < m$, which has reduced dimension compared to (2.1). This dimension reduction may enable more accurate semi-empirical modeling of the map $\tilde{f}$ given experimental data, since a model with fewer inputs typically has fewer parameters to fit with a given data set. Additionally, the unitless quantities allow one to devise scale-invariant experiments, since scaling the units does not change the form of the unitless physical law (2.2). These advantages of dimensional analysis can be exploited for statistical inference and experimental design [4].

2.1.2 Active subspaces

We use active subspaces as a mathematical approach for relating dimension reduction through ridge functions to dimensional analysis. In what follows, we provide some background on active subspaces and their relationship to ridge functions; further details on active subspaces can be found in [18].

The input space from (2.1) is equipped with a known probability measure $\pi_{\mathbf{x}}$. Additionally, assume that (i) f is square-integrable with respect to $\pi_{\mathbf{x}}$ and (ii) f is differentiable with square-integrable partial derivatives. Denote the gradient vector of f as $\nabla f(\mathbf{x})$. Define the $m \times m$ symmetric and positive semidefinite matrix $\boldsymbol{C}$ as

$$\boldsymbol{C} \;=\; \int \nabla f(\mathbf{x})\, \nabla f(\mathbf{x})^\top \, \mathrm{d}\pi_{\mathbf{x}}(\mathbf{x}). \tag{2.8}$$

Since $\boldsymbol{C}$ is symmetric, it admits a real eigenvalue decomposition,

$$\boldsymbol{C} \;=\; \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^\top, \tag{2.9}$$

where the columns of $\boldsymbol{W}$ are the orthonormal eigenvectors, and $\boldsymbol{\Lambda}$ is a diagonal matrix of the associated eigenvalues in descending order such that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m \geq 0$.

The eigenpairs are functionals of f for the given $\pi_{\mathbf{x}}$. Assume that $\lambda_n > \lambda_{n+1}$ (i.e., λ_n is strictly greater than λ_{n+1}) for some $n < m$. Then we can partition the eigenpairs as

$$\boldsymbol{\Lambda} = \begin{bmatrix} \boldsymbol{\Lambda}_1 & \\ & \boldsymbol{\Lambda}_2 \end{bmatrix}, \qquad \boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_1 & \boldsymbol{W}_2 \end{bmatrix}, \tag{2.10}$$

where $\boldsymbol{\Lambda}_1$ contains the first n eigenvalues of $\boldsymbol{C}$, and $\boldsymbol{W}_1$'s columns are the corresponding eigenvectors. The active subspace of dimension n is the span of $\boldsymbol{W}_1$'s columns; the active variables $\mathbf{y} \in \mathbb{R}^n$ are the coordinates of $\mathbf{x}$ in the active subspace. Note that the vector-valued $\mathbf{y}$ differs from the scalar-valued quantity of interest y in (2.1). The active subspace's orthogonal complement, called the inactive subspace, is the span of $\boldsymbol{W}_2$'s columns; its coordinates are denoted $\mathbf{z} \in \mathbb{R}^{m-n}$ and called the inactive variables.

The following property justifies the labels; see Lemma 2.2 from [22]:

$$\begin{aligned} \lambda_1 + \cdots + \lambda_n &= \int \nabla_{\mathbf{y}} f(\mathbf{x})^\top \nabla_{\mathbf{y}} f(\mathbf{x}) \, \mathrm{d}\pi_{\mathbf{x}}(\mathbf{x}), \\ \lambda_{n+1} + \cdots + \lambda_m &= \int \nabla_{\mathbf{z}} f(\mathbf{x})^\top \nabla_{\mathbf{z}} f(\mathbf{x}) \, \mathrm{d}\pi_{\mathbf{x}}(\mathbf{x}), \end{aligned} \tag{2.11}$$

where $\nabla_{\mathbf{y}}$ and $\nabla_{\mathbf{z}}$ denote the gradient of f with respect to the active and inactive variables, respectively. Since the eigenvalues are in descending order, and since f is such that $\lambda_n > \lambda_{n+1}$, (2.11) says that perturbations in $\mathbf{y}$ change f more, on average, than perturbations in $\mathbf{z}$. Equation (2.11) allows us to use active subspaces as an analog for ridge functions.

If $\lambda_{n+1} = \cdots = \lambda_m = 0$, then f is constant along the directions of the inactive variables—i.e., f is a ridge function. If these trailing eigenvalues are sufficiently small, then f can be approximated by a ridge function,

$$f(\mathbf{x}) \;\approx\; g(\boldsymbol{W}_1^\top \mathbf{x}), \tag{2.12}$$

where $g : \mathbb{R}^n \to \mathbb{R}$. If we are given N function evaluations $(\mathbf{x}_i, f(\mathbf{x}_i))$ and an estimate of $\boldsymbol{W}_1$, constructing g with the N pairs $(\boldsymbol{W}_1^\top \mathbf{x}_i, f(\mathbf{x}_i))$ in $n < m$ variables may be possible, whereas N may be too small to construct a function of all m variables. Thus, the active subspace enables dimension reduction in the space of $\mathbf{x}$ for response surface construction.

In practice, we estimate $\boldsymbol{C}$ from (2.8) with numerical integration; high-accuracy Gauss quadrature [39] may be appropriate when the dimension of $\mathbf{x}$ is small and the integrands are sufficiently smooth. Alternatively, we can approximate $\boldsymbol{C}$ using the Monte Carlo method [17]. Draw M independent samples $\mathbf{x}_i$ according to the given $\pi_{\mathbf{x}}$ and compute

$$\boldsymbol{C} \;\approx\; \hat{\boldsymbol{C}} \;=\; \frac{1}{M} \sum_{i=1}^{M} \nabla f(\mathbf{x}_i) \, \nabla f(\mathbf{x}_i)^\top \;=\; \hat{\boldsymbol{W}} \hat{\boldsymbol{\Lambda}} \hat{\boldsymbol{W}}^\top. \tag{2.13}$$

The eigenpairs $\hat{\boldsymbol{\Lambda}}$, $\hat{\boldsymbol{W}}$ of $\hat{\boldsymbol{C}}$ estimate those of $\boldsymbol{C}$. Using results from non-asymptotic random matrix theory [121, 58], we can understand how large the number M of samples must be to ensure quality estimates [17]; the analysis supports a heuristic of $M = \alpha(\delta) \, k \, \log(m)$ samples to estimate the first

k eigenvalues within a relative error δ, where α is typically between 2 and 10 and m is the dimension of $f(\mathbf{x})$. Let ε denote the distance between the true subspace and its Monte Carlo estimate defined as [64]

$$\varepsilon \;=\; \|\boldsymbol{W}_1\boldsymbol{W}_1^\top - \hat{\boldsymbol{W}}_1\hat{\boldsymbol{W}}_1^\top\|_2, \tag{2.14}$$

where $\hat{\boldsymbol{W}}_1$ contains the first n columns of $\hat{\boldsymbol{W}}$. Using standard perturbation theory for invariant subspaces [113], Corollary 3.7 from [17] shows that, for sufficiently large M,

$$\varepsilon \;\leq\; \frac{4\,\lambda_1\,\delta}{\lambda_n - \lambda_{n+1}}. \tag{2.15}$$

Equation (2.15) shows that a large eigenvalue gap $\lambda_n - \lambda_{n+1}$ implies that the active subspace can be accurately estimated with Monte Carlo. Therefore, the practical heuristic is to choose the dimension n of the active subspace according to the largest gap in the eigenvalues.

2.1.3 Connecting dimensional analysis to ridge functions

Next we study the relationship between these two dimension reduction techniques; this closely follows the development in [20]. Dimensional analysis produces the unitless physical law (2.2) by defining unitless quantities (2.5) and (2.6) as products of powers of the dimensional quantities. On the other hand, the inputs to a ridge function are defined by linear combinations of the system's original inputs—i.e., $\mathbf{y} = \boldsymbol{W}_1^\top\mathbf{x}$ and $\mathbf{z} = \boldsymbol{W}_2^\top\mathbf{x}$ for $\mathbf{x} \in \mathbb{R}^m$. Thus, these two techniques are connected via a logarithmic transformation of the input space. Combining (2.5), (2.6), and (2.2),

$$\begin{aligned} y\prod_{i=1}^{m} x_i^{-v_i} &= \tilde{f}\left(\prod_{i=1}^{m} x_i^{u_{i,1}}, \ldots, \prod_{i=1}^{m} x_i^{u_{i,n}}\right) \\ &= \tilde{f}\left(\exp\left(\log\left(\prod_{i=1}^{m} x_i^{u_{i,1}}\right)\right), \ldots, \exp\left(\log\left(\prod_{i=1}^{m} x_i^{u_{i,n}}\right)\right)\right) \\ &= \tilde{f}\left(\exp\left(\sum_{i=1}^{m} u_{i,1}\log(x_i)\right), \ldots, \exp\left(\sum_{i=1}^{m} u_{i,n}\log(x_i)\right)\right) \\ &= \tilde{f}\left(\exp\left(\mathbf{u}_1^\top\log(\mathbf{x})\right), \ldots, \exp\left(\mathbf{u}_n^\top\log(\mathbf{x})\right)\right) \end{aligned} \tag{2.16}$$

where $\log(\mathbf{x})$ returns an m-vector with the log of each component. Then we can rewrite y as

$$\begin{aligned} y &= \exp\left(\mathbf{v}^\top \log(\mathbf{x})\right) \cdot \tilde{f}\left(\exp\left(\mathbf{u}_1^\top \log(\mathbf{x})\right), \dots, \exp\left(\mathbf{u}_n^\top \log(\mathbf{x})\right)\right) \\ &= \tilde{g}(\boldsymbol{A}^\top \log(\mathbf{x})), \end{aligned} \tag{2.17}$$

where

$$\boldsymbol{A} \;=\; \begin{bmatrix} \mathbf{v} & \mathbf{u}_1 & \cdots & \mathbf{u}_n \end{bmatrix} \;\in\; \mathbb{R}^{m\times(n+1)}, \tag{2.18}$$

and $\tilde{g} : \mathbb{R}^{n+1} \to \mathbb{R}$. In other words, the unitless physical law can be transformed into a ridge function of the logs of the physical inputs. Compare (2.17) to the form of the ridge approximation (2.12).

Since the physical law can be written as a ridge function, its active subspace is related to the coefficient matrix $\boldsymbol{A}$. Let $\tilde{\mathbf{x}} = \log(\mathbf{x})$, and assume $\tilde{\pi}_{\tilde{\mathbf{x}}}$ is a probability measure on the space of $\tilde{\mathbf{x}}$ derived from $\pi_{\mathbf{x}}$. By the chain rule,

$$\nabla_{\tilde{\mathbf{x}}}\tilde{g}(\boldsymbol{A}^\top\tilde{\mathbf{x}}) \;=\; \boldsymbol{A}\,\nabla\tilde{g}(\boldsymbol{A}^\top\tilde{\mathbf{x}}), \tag{2.19}$$

where $\nabla_{\tilde{\mathbf{x}}}$ denotes the gradient with respect to $\tilde{\mathbf{x}}$, and $\nabla\tilde{g}$ is the gradient of $\tilde{g}$ with respect to its argument, $\boldsymbol{A}^\top\tilde{\mathbf{x}}$. Plugging (2.19) into (2.8),

$$\int \nabla_{\tilde{\mathbf{x}}}\tilde{g}(\boldsymbol{A}^\top\tilde{\mathbf{x}})\,\nabla_{\tilde{\mathbf{x}}}\tilde{g}(\boldsymbol{A}^\top\tilde{\mathbf{x}})^\top\,\mathrm{d}\tilde{\pi}_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}}) \;=\; \boldsymbol{A}\left(\int \nabla\tilde{g}(\boldsymbol{A}^\top\tilde{\mathbf{x}})\,\nabla\tilde{g}(\boldsymbol{A}^\top\tilde{\mathbf{x}})^\top\,\mathrm{d}\tilde{\pi}_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}})\right)\boldsymbol{A}^\top. \tag{2.20}$$

The first column of $\boldsymbol{A}$ is not in the null space of $\boldsymbol{D}$ from (2.4), and its remaining columns are a basis for $\boldsymbol{D}$'s null space. Therefore, $\boldsymbol{A}$ has full column rank. Then (2.20) shows that the active subspace for the physical law—as a function of the logs of its inputs—has dimension at least $n+1$, and it is a subspace of the $\boldsymbol{A}$'s column space.

The connection between the dimensional analysis and the active subspace provides an upper bound on the dimension of the active subspace. However, the eigenvalues of $\boldsymbol{C}$ from (2.8) rank the importance of each eigenvector-defined direction. So the eigenpairs of $\boldsymbol{C}$ reveal more about the input/output relationship than dimensional analysis alone.

2.2 Magnetohydrodynamics (MHD)

In this section, we perform dimensional analysis on the governing equations of MHD to study the number of unitless quantities affecting the system. MHD models the behavior of electrically-conducting fluids, such as ionized liquids or plasmas. The governing equations for MHD couple the Navier-Stokes equations for fluid dynamics with Maxwell's equations for electromagnetism. Under simplifying assumptions, we can write the equations for steady-state MHD as

$$\begin{aligned}
\nabla \cdot \left[\rho \mathbf{u} \otimes \mathbf{u} + (p_0 + p)\boldsymbol{I} - \mu\left(\nabla \mathbf{u} + \nabla \mathbf{u}^T\right) - \frac{1}{\mu_0}\left(\mathbf{B} \otimes \mathbf{B} - \frac{1}{2}||\mathbf{B}||^2 \boldsymbol{I}\right)\right] &= \mathbf{0}, \\
\nabla \cdot \left[\mathbf{u} \otimes \mathbf{B} - \mathbf{B} \otimes \mathbf{u} - \frac{\eta}{\mu_0}\left(\nabla \mathbf{B} - \nabla \mathbf{B}^T\right)\right] &= \mathbf{0}, \qquad (2.21) \\
\nabla \cdot \mathbf{u} = 0, \qquad \nabla \cdot \mathbf{B} &= 0,
\end{aligned}$$

where the unknown quantities are the fluid velocity $\mathbf{u}$, the magnetic field $\mathbf{B}$, and the fluid pressure p.

It is worthwhile to examine the parameters within the MHD model in detail. For an incompressible flow system solved in terms of the primitive variables $(\mathbf{u}, p, \mathbf{B})$, five parameters appear in the governing equations: i) the fluid viscosity μ, ii) the fluid density ρ, iii) the applied pressure p_0 (more specifically, the applied pressure gradient ∇p_0), iv) the magnetic resistivity of the fluid η, and v) the permeability of a vacuum μ_0. Two additional parameters do not appear in (2.21) but are essential to describing the physics of the system. The first is an external magnetic field B_0 which is applied to the fluid flow. This quantity appears in the boundary conditions, which we do not discuss here. The second is a length scale ℓ that determines the size of the channel through which the fluid is flowing.

Altogether, there are seven model parameters; however, for the active subspace analysis in Section 2.3, we fix two of these parameters—μ_0 and ℓ—such that we only have five independent variables. The quantity μ_0 is fixed since the permeability of a vacuum is a universal constant of a similar nature to the speed of light. Treating this value as variable would not provide useful information. The length scale is considered fixed in this study to represent a specific MHD generator configuration. Despite fixing these parameters for the active subspace analysis, they appear in the dimensional analysis. Proper application of the Buckingham Pi Theorem requires that we include all dimensional quantities relevant to describing the physics of the system, regardless of whether or not they are treated as fixed or variable. Furthermore, fixing these parameters for active subspace does not alter the upper bound on the dimensionality of the reduced parameter space provided by dimensional analysis.

We perform dimensional analysis for the MHD problem from two perspectives. Both approaches are valid and produce equivalent results; however, they each highlight different aspects of the problem. The first may be considered a classical perspective on dimensional analysis. It does not assume $y = f(\mathbf{x})$ from (2.1), i.e., an input/output system as described in Section 2.1. Instead, we apply the Buckingham Pi Theorem directly to governing system of PDEs in (2.21) by constructing the dimension matrix using all relelvant dimensional quantities. Such analysis produces the traditional unitless quantities (e.g., the Reynolds number and Hartmann number) that appear in MHD. The second perspective is motivated by the connection between dimensional analysis and active subspace explained in Section 2.1.3. For this analysis, we treat the seven model parameters as inputs for f. We assume the output is some functional of the solution fields from (2.21). We then follow the procedure discussed in Section 2.1.1.

2.2.1 Classical dimensional analysis for MHD

Classical dimensional analysis is performed directly on the governing equations, which allows us to reformulate them in terms of unitless (or non-dimensional) quantities. Performing such analysis requires that we consider all dimensional quantities that are fundamental to the model. These include the seven model parameters as well as a velocity—which we denote by v—due to the dependent variable $\mathbf{u}$ in (2.21). We need not include additional pressure and magnetic field terms since these appear in the parameters. The necessary base units are L = length, T = time, M = mass, and C = electric current; see Table (1). The model's dimensional quantities, with their units, are

- length, ℓ, with $[\ell] = \mathrm{L}$,

- velocity, v, with $[v] \;=\; \dfrac{\mathrm{L}}{\mathrm{T}}$,
- fluid viscosity, μ, with $[\mu] \;=\; \dfrac{\mathrm{M}}{\mathrm{L\,T}}$,
- fluid density, ρ, with $[\rho] \;=\; \dfrac{\mathrm{M}}{\mathrm{L}^3}$,
- pressure gradient, ∇p_0, with $[\nabla p_0] \;=\; \dfrac{\mathrm{M}}{\mathrm{L}^2\,\mathrm{T}^2}$,
- fluid magnetic resistivity, η, with $[\eta] \;=\; \dfrac{\mathrm{M\,L}^3}{\mathrm{T}^3\,\mathrm{C}^2}$,
- magnetic field, B_0, with $[B_0] \;=\; \dfrac{\mathrm{M}}{\mathrm{T}^2\,\mathrm{C}}$,
- and the vacuum permeability, μ_0, with $[\mu_0] \;=\; \dfrac{\mathrm{ML}}{\mathrm{T}^2\mathrm{C}^2}$.

The associated matrix $\boldsymbol{D}$ (see (2.4)) is

$$\boldsymbol{D} \;=\; \begin{array}{c} \\ \mathrm{L} \\ \mathrm{T} \\ \mathrm{M} \\ \mathrm{C} \end{array} \begin{array}{c} \begin{array}{cccccccc} \ell & v & \mu & \rho & \nabla p_0 & \eta & B_0 & \mu_0 \end{array} \\ \begin{bmatrix} 1 & 1 & -1 & -3 & -2 & 3 & 0 & 1 \\ 0 & -1 & -1 & 0 & -2 & -3 & -2 & -2 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & -2 & -1 & -2 \end{bmatrix} \end{array}. \tag{2.22}$$

A basis for the null space of $\boldsymbol{D}$ is

$$\left\{ \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ -1/2 \\ 0 \\ 0 \\ -1/2 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ -1 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right\}. \tag{2.23}$$

Since the dimension of the null space of $\boldsymbol{D}$ is four, the Buckingham Pi Theorem states that the system depends on four unitless quantities; see (2.6). In this case,

$$\begin{aligned}
\Pi_1 &= \ell^1 v^1 \mu^{-1} \rho^1 \nabla p_0^0 \eta^0 B^0 \mu_0^0 = \frac{\ell\, v\, \rho}{\mu}, \\
\Pi_2 &= \ell^1 v^0 \mu^{-1/2} \rho^0 \nabla p_0^0 \eta^{-1/2} B_0^1 \mu_0^0 = \frac{\ell\, B_0}{\mu^{1/2}\eta^{1/2}}, \\
\Pi_3 &= \ell^1 v^1 \mu^0 \rho^0 \nabla p_0^0 \eta^{-1} B_0^0 \mu_0^1 = \frac{\ell\, v\, \mu_0}{\eta}, \\
\Pi_4 &= \ell^1 v^{-2} \mu^0 \rho^{-1} \nabla p_0^1 \eta^0 B_0^0 \mu_0^0 = \frac{\ell\, \nabla p_0}{v^2 \rho}.
\end{aligned} \tag{2.24}$$

An alternative way to determine the unitless quantities for (2.21) is to nondimensionalize the equations. To do so, we express the spatial variables and the outputs as products of a unitless quantity (denoted by $*$) and a characteristic quantity (denoted with a subscript c),

$$\mathbf{x} = \mathbf{x}^* \ell_c, \quad \mathbf{u} = \mathbf{u}^* v_c, \quad \mathbf{B} = \mathbf{B}^* B_c, \quad p_0 = p_0^* p_c, \quad \text{and} \quad p = p^* p_c, \tag{2.25}$$

where the characteristic quantities have units matching their corresponding dimensional quantities.

Plug (2.25) in (2.21) and simplify to obtain

$$\begin{aligned}
\nabla^* \cdot \left[\mathbf{u}^* \otimes \mathbf{u}^* + (p_0^* + p^*)\boldsymbol{I} - \frac{1}{Re}\left(\nabla^* \mathbf{u}^* + \nabla^* \mathbf{u}^{*^T}\right) - \frac{1}{R_m}\frac{Ha^2}{Re}\left(\mathbf{B}^* \otimes \mathbf{B}^* - \frac{1}{2}||\mathbf{B}^*||^2 \boldsymbol{I}\right)\right] &= \mathbf{0}, \\
\nabla^* \cdot \left[\mathbf{u}^* \otimes \mathbf{B}^* - \mathbf{B}^* \otimes \mathbf{u}^* - \frac{1}{R_m}\left(\nabla^* \mathbf{B}^* - \nabla^* \mathbf{B}^{*^T}\right)\right] &= \mathbf{0}, \\
\nabla^* \cdot \mathbf{u}^* = 0, \qquad \nabla^* \cdot \mathbf{B}^* &= 0.
\end{aligned} \tag{2.26}$$

The unitless spatial variables appear in the derivatives

$$\nabla^* = \begin{bmatrix} \frac{\partial}{\partial x_1^*} & \frac{\partial}{\partial x_2^*} & \frac{\partial}{\partial x_3^*} \end{bmatrix}^T. \tag{2.27}$$

Equation (2.26) depends on four unitless quantities that are well-known from fluid dynamics [99] and electromagnetics [38]:

- the Reynolds number, $Re = \dfrac{\ell_c\, v_c\, \rho}{\mu}$,
- the Hartmann number, $Ha = \dfrac{\ell_c\, B_c}{\mu^{1/2}\eta^{1/2}}$,

- the magnetic Reynolds number, $R_m = \dfrac{\ell_c\, v_c\, \mu_0}{\eta}$,
- and a dimensionless pressure gradient, $\nabla^* \cdot (p_0^* \boldsymbol{I}) = \nabla^* p_0^* = \dfrac{\ell_c\, \nabla p_0}{v_c^2 \rho}$.

Scaling the MHD equations is consistent with the unitless quantities derived from the basis (2.23). In other words, the unitless quantities from the classical Buckingham Pi analysis match those in (2.26).

2.2.2 Active subspaces-motivated dimensional analysis for MHD

In this section, we perform dimensional analysis motivated by the connection between active subspaces and dimensional analysis discussed in Section 2.1.3. This analysis assumes a problem of the form $y = f(\mathbf{x})$. We consider the seven model parameters to be the inputs $\mathbf{x}$. The scalar-valued output is a functional of the unknown solution fields (e.g., velocity, magnetic field) in the governing equations. In Section 2.3, we consider two outputs: the average flow velocity and the magnitude of the induced magnetic field. For the dimensional analysis in this section, we have a single $\boldsymbol{D}$ matrix (see (2.4)) associated with the set of inputs and one distinct $\mathbf{u}$ vector for each output.

The matrix $\boldsymbol{D}$ from (2.4) is

$$\boldsymbol{D} = \begin{array}{c} \\ \text{L} \\ \text{T} \\ \text{M} \\ \text{C} \end{array} \begin{array}{c} \begin{array}{ccccccc} \ell & \mu & \rho & \nabla p_0 & \eta & B_0 & \mu_0 \end{array} \\ \begin{bmatrix} 1 & -1 & -3 & -2 & 3 & 0 & 1 \\ 0 & -1 & 0 & -2 & -3 & -2 & -2 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & -2 & -1 & -2 \end{bmatrix} \end{array}. \qquad (2.28)$$

A basis for the null space of $\boldsymbol{D}$ is

$$\left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ -2 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1/2 \\ 0 \\ 0 \\ -1/2 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ -1 \\ 0 \\ -1 \\ 0 \\ 1 \end{bmatrix} \right\}. \tag{2.29}$$

If we consider the average velocity output, then the $\mathbf{u}$ vector from (2.4) is

$$\mathbf{u} = \begin{bmatrix} 1 & -1 & 0 & 0 \end{bmatrix}^T. \tag{2.30}$$

Solving $\boldsymbol{D}\mathbf{v} = \mathbf{u}$ yields

$$\mathbf{v} = \begin{bmatrix} 0 & 0 & -1/3 & 1/3 & 1/3 & 0 & -1/3 \end{bmatrix}^T. \tag{2.31}$$

The matrix $\boldsymbol{A}$ from (2.18) is

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & -1/2 & 1 \\ -1/3 & 0 & 0 & -1 \\ 1/3 & 1 & 0 & 0 \\ 1/3 & 0 & -1/2 & -1 \\ 0 & -2 & 1 & 0 \\ -1/3 & 1 & 0 & 1 \end{bmatrix}. \tag{2.32}$$

This matrix has full column rank. Therefore, the average flow velocity depends on at most four linear combinations of the log-transformed input parameters, as shown in Section 2.1.3.

Next, we consider the total induced magnetic field. The $\mathbf{u}$ vector from (2.4) for this output is

$$\mathbf{u} = \begin{bmatrix} 0 & -2 & 1 & -1 \end{bmatrix}^T, \tag{2.33}$$

and the solution to $\boldsymbol{D}\mathbf{v} = \mathbf{u}$ is

$$\mathbf{v} = \begin{bmatrix} 0 & 0 & 1/6 & 1/3 & 1/3 & 0 & 1/6 \end{bmatrix}^T. \tag{2.34}$$

The matrix $\boldsymbol{A}$ is

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & -1/2 & 1 \\ 1/6 & 0 & 0 & -1 \\ 1/3 & 1 & 0 & 0 \\ 1/3 & 0 & -1/2 & -1 \\ 0 & -2 & 1 & 0 \\ 1/6 & 1 & 0 & 1 \end{bmatrix}. \tag{2.35}$$

Once again, $\boldsymbol{A}$ has full column rank implying that the induced magnetic field depends on four or fewer linear combinations of the log-transformed input parameters. In the next section, we analyze the active subspaces of the particular outputs, average velocity and total induced magnetic field, as functions of the model parameters; we numerically verify the theoretical results about the number of linear combinations of inputs that affect the outputs.

2.3 Active subspaces for MHD

The dimensional analysis in the previous section, coupled with the analysis from Section 2.1.3, indicates that both the average flow velcity and the induced magnetic field can be written as a ridge function of four linear combinations of the log transformed inputs. Therefore, we expect numerical tests to reveal an active subspace of dimension 4 or less. We study two MHD models with active subspaces: (i) the Hartmann problem that models a simplified duct flow and (ii) a numerical model of an idealized MHD generator in three spatial dimensions.

2.3.1 Hartmann problem

The Hartmann problem is a standard test problem in MHD. It models laminar flow between two parallel plates. In general, we assume these plates are separated by distance 2ℓ. We define

$\ell = 1$m for the numerical investigations in this section. The fluid is assumed to be a conducting fluid and a uniform magnetic field is applied perpendicular to the flow direction. As the fluid flow bends the magnetic field, an induced field in the horizontal direction is generated and a corresponding magnetic stress is developed that resists, or damps, the flow non-uniformly in the boundary layer and the core of the flow. This can be seen in Figure 2.1.

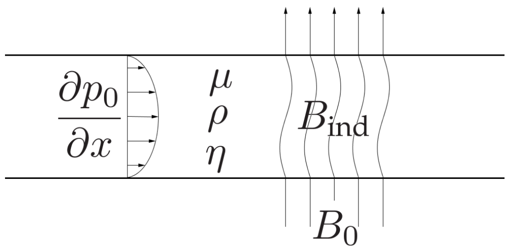


Figure 2.1: Depiction of the Hartmann problem. A magnetic fluid flows between two parallel plates in the presence of a perpendicular magnetic field. The field acts as a damping force on the fluid while the flow induces a horizontal magnetic field.

The advantage of working with the Hartmann problem is that it admits closed form analytical expressions for the quantities of interest in terms of their input parameters. This enables thorough numerical studies where errors can be computed exactly. As mentioned earlier, we consider two quantities of interest: (i) average flow velocity across the channel u_{avg} and (ii) the induced magnetic field B_{ind}. Derivations of these quantities of interest for the Hartmann problem from the governing MHD equations (2.21) are not presented here; they may be in found in [36]. As functions of the model inputs (see Section 2.2.1),

$$u_{\text{avg}} = -\frac{\partial p_0}{\partial x}\frac{\eta}{B_0^2}\left(1 - \frac{B_0\ell}{\sqrt{\eta\mu}}\coth\left(\frac{B_0\ell}{\sqrt{\eta\mu}}\right)\right) \tag{2.36}$$

and

$$B_{\text{ind}} = \frac{\partial p_0}{\partial x}\frac{\ell\mu_0}{2B_0}\left(1 - 2\frac{\sqrt{\eta\mu}}{B_0\ell}\tanh\left(\frac{B_0\ell}{2\sqrt{\eta\mu}}\right)\right). \tag{2.37}$$

The expressions for the Hartmann problem quantities of interest have five input parameters: fluid viscosity μ, fluid density ρ, applied pressure gradient $\partial p_0/\partial x$ (where the derivative is with respect to the flow field's spatial coordinate), resistivity η, and applied magnetic field B_0. Note that fluid

Table 2.2: Indices and intervals for the parameters $\mathbf{x}$ of the Hartmann problem. These intervals represent the expected operating conditions for an MHD generator modeled with the Hartmann problem.

Index	Name	Notation	Interval
1	fluid viscosity	$\log(\mu)$	$[\log(0.05), \log(0.2)]$
2	fluid density	$\log(\rho)$	$[\log(1), \log(5)]$
3	applied pressure gradient	$\log\left(\frac{\partial p_0}{\partial x}\right)$	$[\log(0.5), \log(3)]$
4	resistivity	$\log(\eta)$	$[\log(0.5), \log(3)]$
5	applied magnetic field	$\log(B_0)$	$[\log(0.1), \log(1)]$

density ρ does not appear in either (2.36) or (2.37). We treat ρ as an input parameter for two reasons. First, ρ appears in the governing MHD equations (2.21); its absence from the closed form solutions, (2.36) and (2.37), is not apparent without knowing the closed form solutions. Second, the problem provides an interesting test for whether or not the active subspace analysis can identify the lack of dependence on ρ.

Recall from Section 2.1.3 that we consider the quantities of interest as functions of the log-transformed inputs. Let

$$\mathbf{x} = \begin{bmatrix} \log(\mu) & \log(\rho) & \log\left(\frac{\partial p_0}{\partial x}\right) & \log(\eta) & \log(B_0) \end{bmatrix}^T \tag{2.38}$$

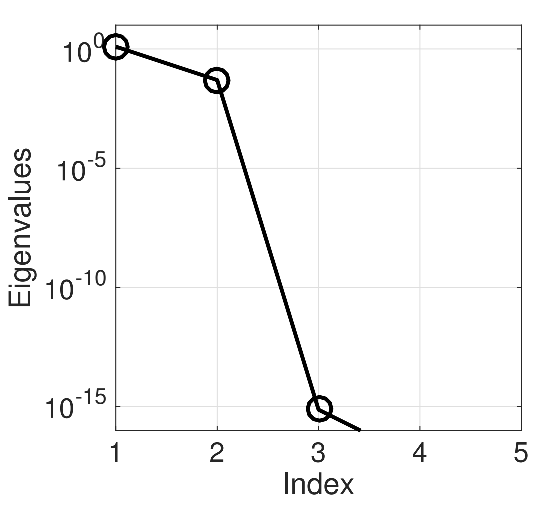
be the vector of inputs for the active subspace analysis from Section 2.1.2. To estimate the active subspace for each quantity of interest, we define $\pi_{\mathbf{x}}$ to have a uniform density over a five-dimensional hyperrectangle. The ranges of each of $\mathbf{x}$'s components are in Table 2.2; they are chosen to represent the expected operating conditions of an MHD generator modeled with the Hartmann problem. We estimate $\boldsymbol{C}$ from (2.8) with a tensor product Gauss-Legendre quadrature rule [39] with 11 points in each dimension—a total of $11^5 = 161051$ points. This is sufficient for 10 digits of accuracy in the eigenvalue estimates.

Figure 2.2 shows the results of analyzing the active subspace of the Hartmann problem's average flow velocity u_{avg} from (2.36). Figure 2.2a shows that all but two eigenvalues are zero (to machine precision). This implies that the active subspace of dimension $n = 2$ is sufficient to describe the relationship between the log-transformed inputs and the quantity of interest. Figure

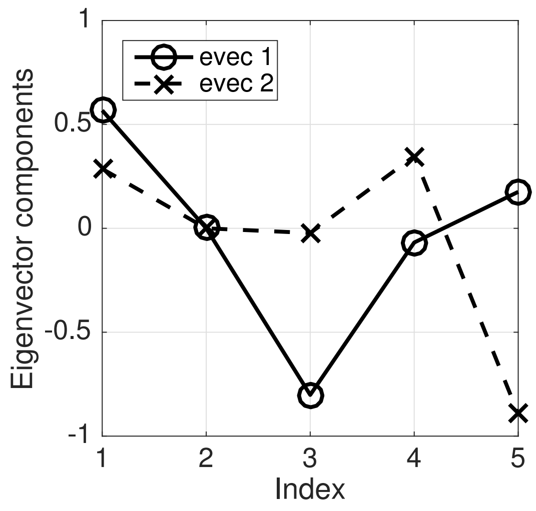
2.2b shows the components of the first two eigenvectors of $\boldsymbol{C}$'s quadrature estimate; the index on the horizontal axis maps to the specific input as in Table 2.2. A large eigenvector component reveals that the corresponding parameter is important in defining the active subspace. Notice that both eigenvector components corresponding to $\log(\rho)$ (the second input) are zero; this is consistent with the definition of u_{avg} in (2.36), which does not depend on fluid density ρ, as discussed earlier. Figure 2.2c is a shadow plot of 1000 samples of u_{avg}—taken from the quadrature evaluations used to estimate $\boldsymbol{C}$—versus the corresponding samples of the active variable. Such plots are commonly used in regression graphics [30]. The plot shows a strong relationship between the first active variable and u_{avg}, so a ridge function of the form (2.12) with one linear combination would be a good approximation; this is validated by the three-order-of-magnitude gap between the first and second eigenvalues. Figure 2.2d shows a two-dimensional shadow plot with the same data, where the color is the value of u_{avg}, the horizontal axis is the first active variable (defined by the first eigenvector of $\boldsymbol{C}$), and the vertical axis is the second active variable (defined by the second eigenvector of $\boldsymbol{C}$). Since the eigenvalues with index greater than two are zero, the two-dimensional shadow plot reveals the complete relationship between the log-transformed inputs and u_{avg}.

Figure 2.3 shows the same plots as Figure 2.2 for the induced magnetic field B_{ind} from (2.37) as the quantity of interest. Compared to u_{avg}, the one-dimensional shadow plot for B_{ind}, Figure 2.3c, shows greater departure from a univariate relationship when the first active variable $\mathbf{w}_1^T\mathbf{x}$ is less than 0. This structure is confirmed in the two-dimensional shadow plot, Figure 2.3d, which shows the truly two-dimensional structure in map from inputs to B_{ind}. The eigenvectors in Figure 2.3b differ substantially from the eigenvectors associated with u_{avg} in Figure 2.2b; these differences reflect the different dependence between inputs and outputs, which are verified in the analytical expressions from (2.36) and (2.37). Notably, the second component of the eigenvectors is zero for both outputs, which indicates that neither output depends on the fluid density ρ (the second input parameter).

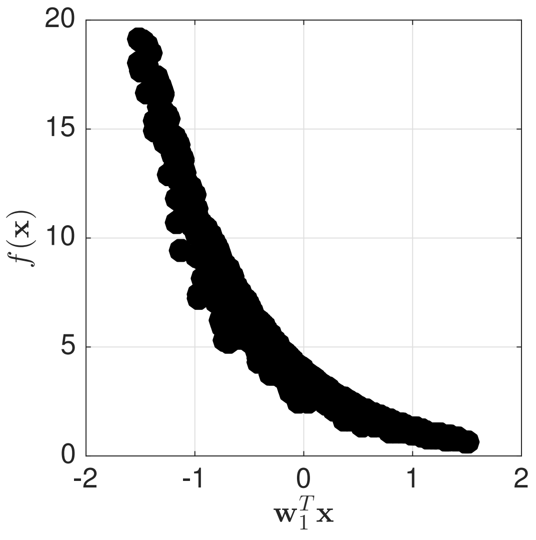
The dimensional analysis from Section 2.2 showed that the quantities of interest should depend on four unitless quantities. Equation (2.26) expresses the governing equations in terms of



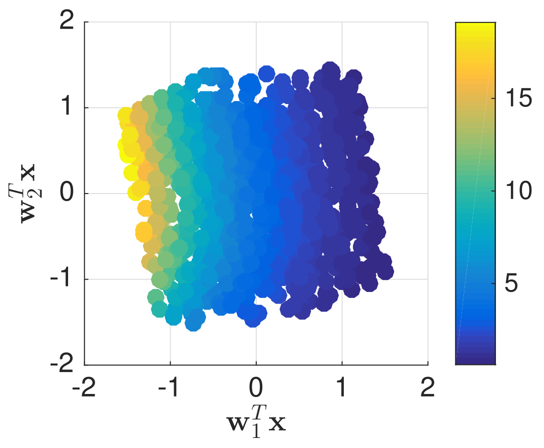
(a) Eigenvalues of $\boldsymbol{C}$



(b) Eigenvectors of $\boldsymbol{C}$



(c) 1-D shadow plot

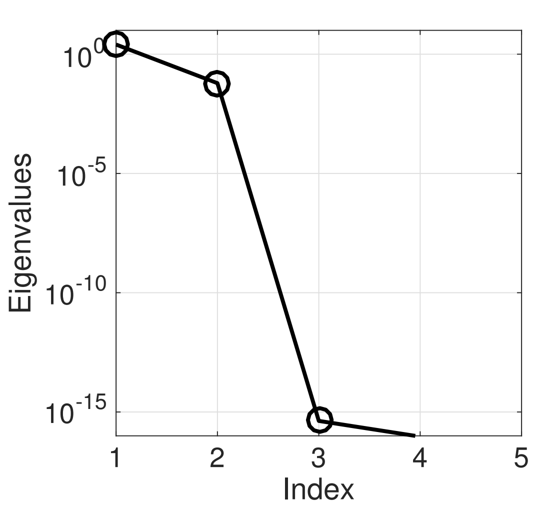


(d) 2-D shadow plot

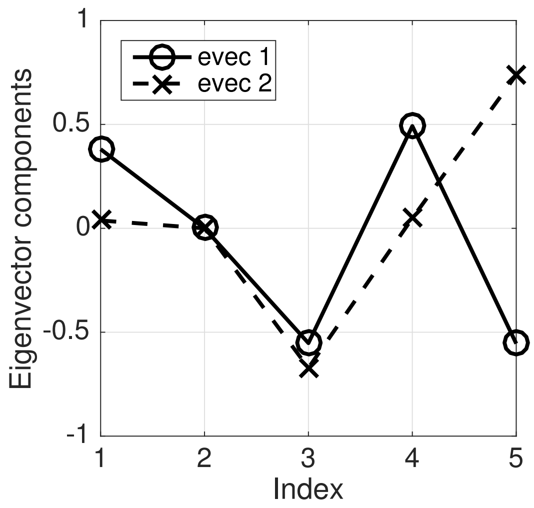
Figure 2.2: These figures represent the active subspace-based dimension reduction for the Hartmann problem's average flow velocity u_{avg} from (2.36). Figure 2.2a shows the eigenvalues of $\boldsymbol{C}$, and Figure 2.2b shows the components of $\boldsymbol{C}$'s first two eigenvectors. Figures 2.2c and 2.2d are one- and two-dimensional, respectively, shadow plots of the quantity of interest.

the Reynolds number, the Hartmann number, the magnetic Reynolds number and a dimensionless pressure gradient. With scaling, we can write unitless forms of the quantities of interest, u_{avg} from (2.36) and B_{ind} from (2.37), in terms of unitless parameters:

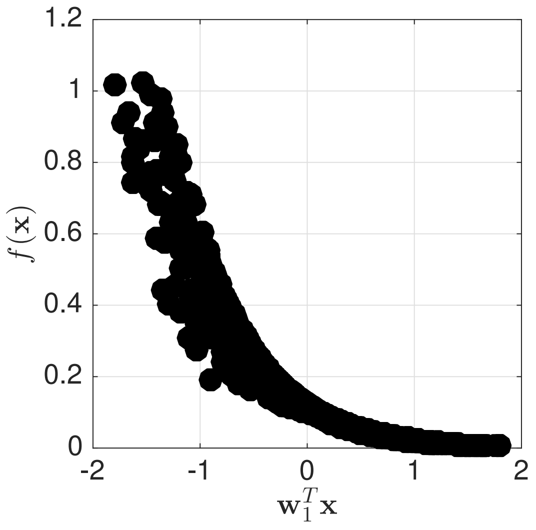
$$u^*_{\text{avg}} = -\frac{\partial p^*_0}{\partial x^*}\frac{Re}{Ha^2}\left(1 - Ha\coth\left(Ha\right)\right) \tag{2.39}$$



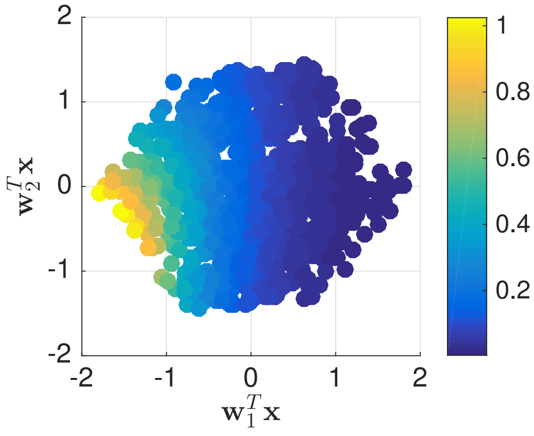
(a) Eigenvalues of $\boldsymbol{C}$



(b) Eigenvectors of $\boldsymbol{C}$



(c) 1-D shadow plot



(d) 2-D shadow plot

Figure 2.3: These figures represent the active subspace-based dimension reduction for the Hartmann problem's induced magnetic field B_{ind} from (2.37). Figure 2.3a shows the eigenvalues of $\boldsymbol{C}$, and Figure 2.3b shows the components of $\boldsymbol{C}$'s first two eigenvectors. Figures 2.3c and 2.3d are one- and two-dimensional, respectively, shadow plots of the quantity of interest.

and

$$B_{\text{ind}}^* = \frac{\partial p_0^*}{\partial x^*}\frac{Re}{2\,Ha^2}R_m\left(1-\frac{2}{Ha}\tanh\left(\frac{Ha}{2}\right)\right). \tag{2.40}$$

Notice several of the unitless quantities appear only as a product in (2.39) and (2.40). This product of unitless quantities defines a new unitless quantity. Thus, u_{avg}^* and B_{ind}^* depend only on two unitless quantities. This explains why the eigenvalues of $\boldsymbol{C}$ are zero after the second.

We next compare the ridge structure with respect to the log-transformed input space versus the original input space. To do this, we approximate the expected conditional variance from (1.10). Recall that this value serves as a metric for quantifying how well a function can be approximated by a ridge function. To approximate (1.10), we draw 10,000 random samples according to $\pi_{\mathbf{x}}$ and transform these samples into n-dimensional space via $\boldsymbol{A}^\top \mathbf{x}$. Then at each of these points, we use a hit-and-run algorithm to draw 1,000 conditional samples. We compute the conditional variance of each set of 1,000 samples and then average all of the conditional variances to approximate the expected conditional variance. The smaller this quantity, the better a given function is approximated by an n-dimensional ridge function. Such a study is only feasible because we have the closed form solutions to u_{avg} and B_{ind}, but it provides interesting insights into approximate ridge structure with respect to the log-transformed inputs.

Table (2.3) contains the results of this study performed on both B_{ind} and u_{avg}. Note that we normalize the expected conditional variance by the total variance of $f(\mathbf{x})$ so we can compare performance on the two different outputs. Additionally, note that the samples in each case are drawn from identical distributions since $\pi_{\mathbf{x}}$ can play a role in influencing approximate ridge structure. We perform this study using ridge functions of increasing dimension. Using the original inputs, our approximate ridge structure seems to perform reasonably well. The one-dimensional ridge function approximation captures around 90-95% of total variance in the output with improvement made by allowing more ridge directions—i.e., increasing n. However, the log-transformed inputs result in a numerically-exact ridge function after two dimensions for both B_{ind} and u_{avg}. Additionally, the one-dimensional ridge approximations perform better in each case for the log-transformed inputs. This aligns with Figures 2.2 and 2.3 and suggests that a log-transformation of the inputs can improve ridge approximation in physically-motivated problems.

Table 2.3: The normalized expected conditional variance from (1.10) for both B_{ind} and u_{avg} in terms of the original and log-transformed inputs.

	B_{ind} Original inputs	B_{ind} Log-transformed inputs	u_{avg} Original inputs	u_{avg} Log-transformed inputs
$n = 1$	1.01×10^{-1}	5.50×10^{-2}	5.71×10^{-2}	1.29×10^{-2}
$n = 2$	3.85×10^{-2}	1.42×10^{-23}	9.31×10^{-3}	1.47×10^{-23}
$n = 3$	1.93×10^{-2}	1.78×10^{-23}	5.81×10^{-4}	2.55×10^{-23}
$n = 4$	5.06×10^{-28}	4.75×10^{-28}	7.39×10^{-28}	6.01×10^{-28}

2.3.2 MHD generator problem

This model is a steady-state MHD duct flow configuration representing an idealized MHD generator; the governing equations for the flow and magnetic field are the resistive MHD equation—a slightly modified version[1] of the governing equations presented in Section 2.2; see [105] for details. The MHD generator induces an electrical current by supplying a set flow-rate of a conducting fluid through an externally supplied vertical magnetic field. The bending of the magnetic field lines produces a horizontal electrical current. The geometric domain for this problem is a square cross-sectional duct of dimensions 15m $\times$ 1m $\times$ 1m. The simple geometry problem facilitates scalability studies as different mesh sizes can be easily generated. The velocity boundary conditions are set with Dirichlet inlet velocity of $[1, 0, 0]\,\mathrm{m}\cdot\mathrm{s}^{-1}$, no slip on the top, bottom and sides of the channel, and natural boundary conditions on the outflow. The magnetic field boundary conditions on the top and bottom are specified as a set magnetic field configuration $(0, B_y^{gen}, 0)$, where

$$B_y^{gen} = \frac{1}{2}B_0\left[\tanh\left(\frac{x - x_{\text{on}}}{\delta}\right) - \tanh\left(\frac{x - x_{\text{off}}}{\delta}\right)\right]. \tag{2.41}$$

The values x_{on} and x_{off} indicate the locations in the x direction where the magnetic field is active. The inlet, outlet, and sides are perfect conductors with $\mathbf{B}\cdot\hat{\mathbf{n}} = \mathbf{0}$ and $\mathbf{E}\times\hat{\mathbf{n}} = \mathbf{0}$, i.e., the current and magnetic fluxes are zero at these boundaries. Homogeneous Dirichlet boundary conditions are used on all surfaces for the Lagrange multiplier enforcing the solenoidal constraint $\nabla\cdot\mathbf{B} = 0$; see [105]. This problem has similar characteristics to the Hartmann problem with (i) viscous boundary layers, (ii) Hartmann layers occurring at the boundaries, and (iii) a flow field that is strongly

[1] The minor modification to the governing equations does not change the dimensional analysis.

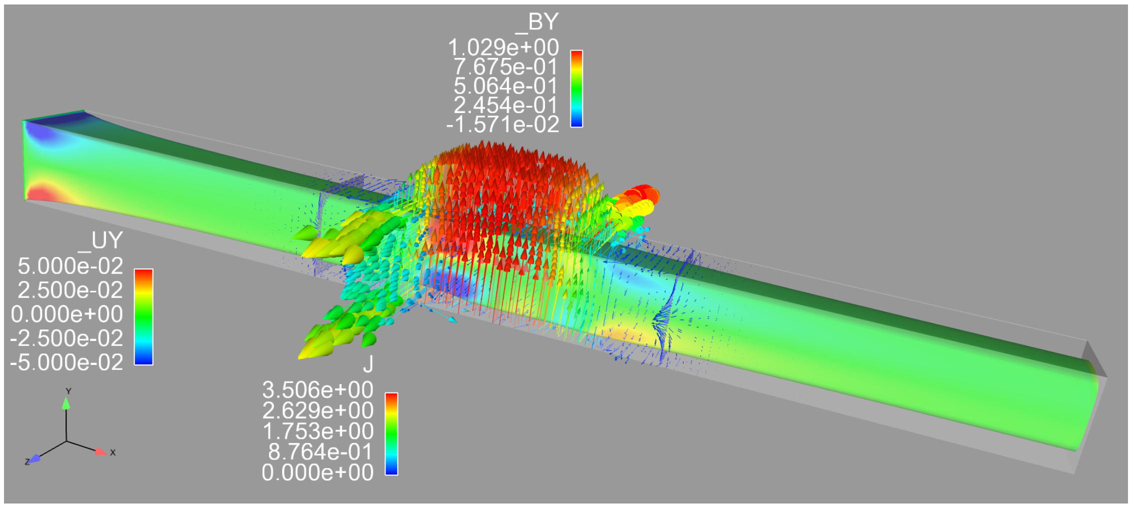


Figure 2.4: Visualization of flow field from an idealized 3D MHD generator model. The image shows x velocity iso-surface colored by the y velocity. Vectors (colored by magnitude) show vertical magnetic field (applied and induced) and horizontal induced current.

modified by the magnetic field in the section of the duct where it is active. Figure 2.4 shows a solution for this problem for Reynolds number $Re = 2500$, magnetic Reynolds number $Re_m = 10$, and Hartmann number $Ha = 5$. The image shows x velocity iso-surface colored by y velocity, where the modification of the inlet constant profile and the parabolic profile at the region where the magnetic field is active are evident. Vectors (colored by magnitude) show the vertical magnetic field (applied and induced) and horizontal induced current from the bending of the magnetic field lines.

The fixed physical parameters for the MHD generator are $\mu_0 = 1$, $x_{\text{on}} = 4.0$, $x_{\text{off}} = 6.0$, and $\delta = 0.1$. The variable input parameters are the same as in the Hartmann problem. However, the generator uses different input ranges, which can be found in Table 2.4. The probability measure on the space of inputs is uniform over the hyperrectangle of log-transformed parameters defined by the ranges in Table 2.4. The quantities of interest are the average flow velocity u_{avg} and the induced magnetic field B_{ind}, as in the Hartmann problem.

Given values for the input parameters, the MHD generator's solution fields are computed with the Sandia National Laboratory's Drekar multiphysics solver package [106, 95]. The package

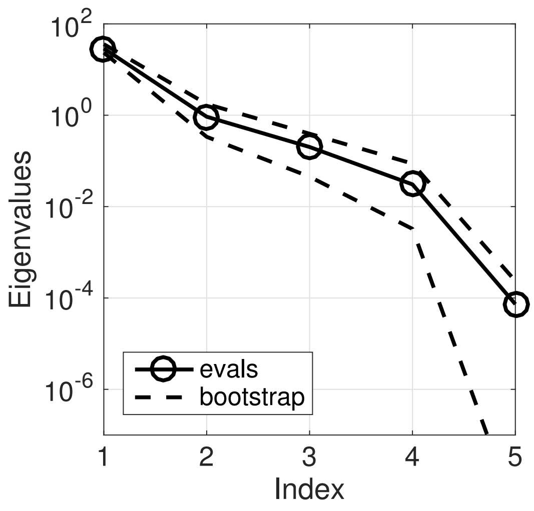
Table 2.4: Indices and intervals for the parameters $\mathbf{x}$ of the MHD generator problem. These intervals represent the expected operating conditions for the idealized MHD generator.

Index	Name	Notation	Interval
1	fluid viscosity	$\log(\mu)$	$[\log(0.001), \log(0.01)]$
2	fluid density	$\log(\rho)$	$[\log(0.1), \log(10)]$
3	applied pressure gradient	$\log\left(\frac{\partial p_0}{\partial x}\right)$	$[\log(0.1), \log(0.5)]$
4	resistivity	$\log(\eta)$	$[\log(0.1), \log(10)]$
5	applied magnetic field	$\log(B_0)$	$[\log(0.1), \log(1)]$

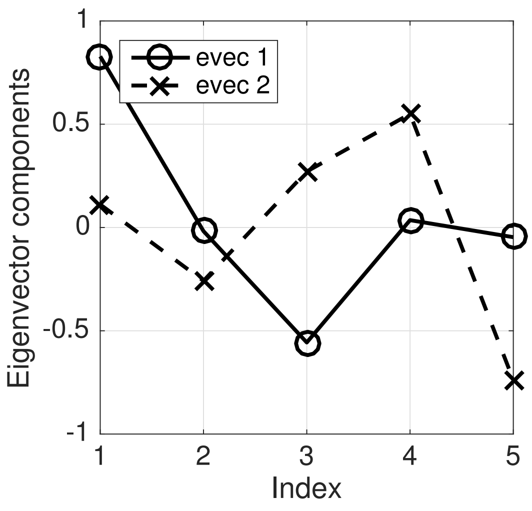
has adjoint capabilities, which enables computation of the derivatives of the quantities of interest with respect to the input parameters [106]. Each MHD generator model run uses 5.3 CPU-hours (10 minutes on 32 cores), so estimating $\boldsymbol{C}$ from (2.8) with a tensor product Gauss-Legendre quadrature rule is not possible. Instead, we use a Monte Carlo method to estimate $\boldsymbol{C}$ using $M = 483$ independent samples from the uniform probability measure on the log-transformed parameters. For this study, our budget was roughly 500 simulations, which is common for simulation models whose size and complexity are comparable to the Drekar-based MHD model. For details on the accuracy of the Monte Carlo method for estimating active subspaces, see [17].

Figure 2.5a shows the eigenvalue estimates computed with Monte Carlo for the u_{avg} quantity of interest. The dashed lines show upper and lower bounds on the eigenvalue estimates computed with a nonparametric bootstrap with 500 bootstrap replicates from the set of 483 gradient samples. We emphasize that since there is no randomness in the map from inputs to outputs (i.e., the computer simulation is deterministic), the bootstrap is a heuristic to estimate the variability due to the Monte Carlo sampling. Estimates of standard error from sample variances are not appropriate, since the eigenvalues are nonlinear functions of the gradient samples. For an example of a similar bootstrap computation for eigenvalues, see [45, Chapter 7.2].

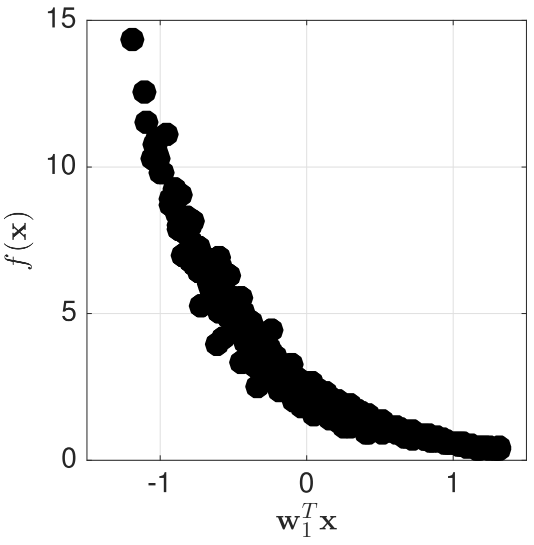
The fifth eigenvalue is 0.00024% of the sum of the five eigenvalues, which is consistent with the dimensional analysis from Section 2.2—i.e., there should be no more than 4 linear combinations of the model parameters that affect the quantity of interest. And this restriction is reflected in the small fifth eigenvalue.



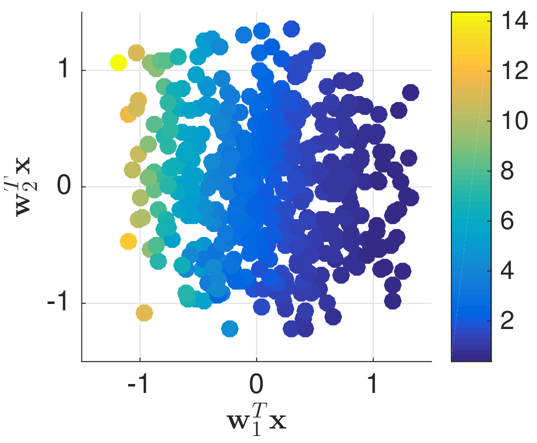
(a) Eigenvalues of $\boldsymbol{C}$



(b) Eigenvectors of $\boldsymbol{C}$



(c) 1-D shadow plot



(d) 2-D shadow plot

Figure 2.5: These figures represent the active subspace-based dimension reduction for the MHD generator problem's average flow velocity u_{avg}. Figure 2.5a shows the eigenvalues of $\boldsymbol{C}$, and Figure 2.5b shows the components of $\boldsymbol{C}$'s first two eigenvectors. Figures 2.5c and 2.5d are one- and two-dimensional, respectively, shadow plots of the quantity of interest.

The first two eigenvectors of $\boldsymbol{C}$'s Monte Carlo estimate (for u_{avg}) are shown in Figure 2.6b. The magnitudes of the eigenvector components can be used to determine which physical parameters influence the active subspace—i.e., they provide sensitivity information. (See [41] for how to construct sensitivity metrics from active subspaces.) The fluid viscosity μ and the pressure gradient $\partial p_0/\partial x$ are the most important parameters for the average fluid velocity. This insight agrees with physical intuition, and it is consistent with the same metrics from the Hartmann problem; see Figure 2.2b.

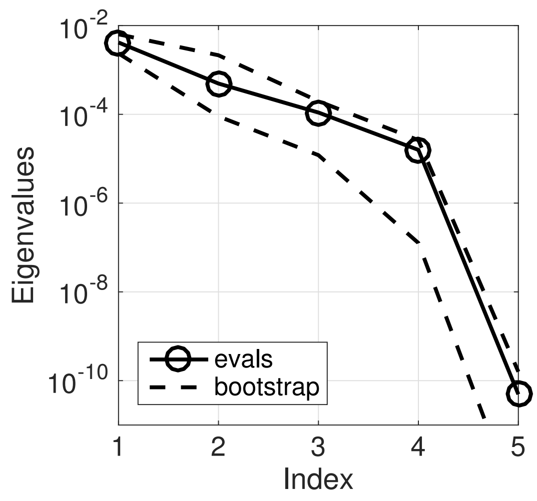
Figure 2.5c and 2.5d show the one- and two-dimensional shadow plots for u_{avg} as a function of the first two active variables using all 483 samples. Similar to Figures 2.2c and 2.2d, we see a nearly one-dimensional relationship between the log-transformed input parameters and the average velocity, where the one dimension is the first active variable.

Figure 2.6a shows the eigenvalues for $\boldsymbol{C}$'s Monte Carlo estimate, with bootstrap ranges, for the induced magnetic field quantity of interest B_{ind}. In this case, the fifth eigenvalue is 0.000001% of the sum of the eigenvalues, which is consistent with the dimensional analysis from Section 2.1.3 that shows that any quantity of interest will depend on at most four linear combinations of the log-transformed input parameters. The first eigenvector in Figure 2.6b shows that B_{ind} depends on all input parameters except the fluid density ρ. This is remarkably similar to the dependence seen in the Hartmann problem; see Figure 2.3b.

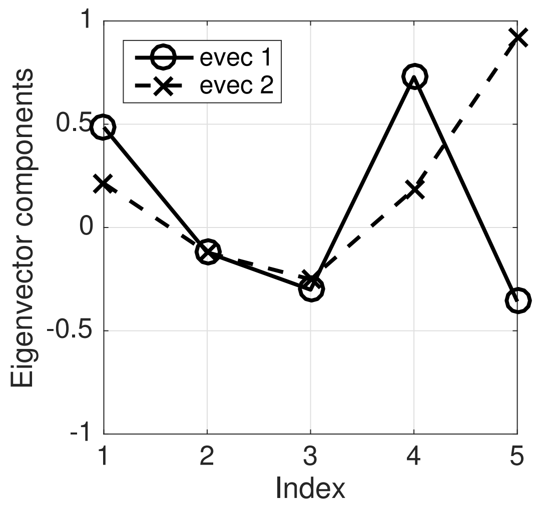
The one- and two-dimensional shadow plots for B_{ind} are in Figures 2.6c and 2.6d. There appears to be a region in the parameter space—when the first active variable is positive—where the relationship between the inputs and B_{ind} is well characterized by one linear combination of the log-transformed inputs. However, the one-dimensional character of that relationship degrades as the first active variable decreases. (Note that the first eigenvector is only unique up to a sign, so this relationship could be inverted.) The relative signs of the eigenvector components yield insight into the how the output varies as each input is varied. For example, the fourth and fifth components of the first eigenvector have opposite signs. Thus, the corresponding model parameters—resistivity η and applied magnetic field B_0—affect B_{ind} in opposite directions, on average.

2.4 Summary

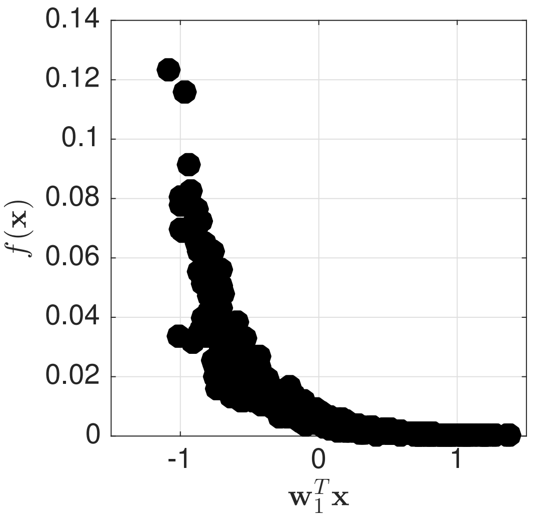
In this chapter, we reviewed two methods for dimension reduction in physical systems: (i) dimensional analysis that uses the physical quantities' units and (ii) active subspaces that use the



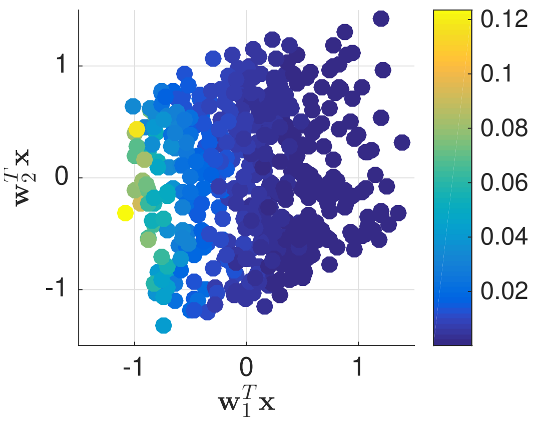
(a) Eigenvalues of $\boldsymbol{C}$



(b) Eigenvectors of $\boldsymbol{C}$



(c) 1-D shadow plot



(d) 2-D shadow plot

Figure 2.6: These figures represent the active subspace-based dimension reduction for the MHD generator's induced magnetic field B_{ind}. Figure 2.6a shows the eigenvalues of $\boldsymbol{C}$, and Figure 2.6b shows the components of $\boldsymbol{C}$'s first two eigenvectors. Figures 2.6c and 2.6d are one- and two-dimensional, respectively, shadow plots of the quantity of interest.

gradient of the output with respect to the inputs to address the ridge recovery problem. We showed the connection between these two methods via a log transform of the input parameters—namely, that the dimensional analysis provides an upper bound on the number of linear combinations of log-transformed parameters that control any quantity of interest.

We applied these techniques to two quantities of interest—average flow velocity and induced magnetic field—from two magnetohydrodynamics models that apply to power generation: (i) the Hartmann problem that admits closed form expressions for the quantities of interest and (ii) a large-scale computational model of coupled fluid flow, magnetic fields, and electric current in a three-dimensional duct. The computational model has adjoint capabilities that enable gradient evaluations.

The insights from the active subspace analysis is consistent with the dimensional analysis. In particular, there are at most four linear combinations of log-transformed parameters that affect the quantity of interest—which is a reduction from an ambient dimension of 5 to an intrinsic dimension of 4. The Hartmann problem has a further reduction to an intrinsic dimension of 2—i.e., two linear combinations of log-transformed parameters are sufficient to characterize the quantities of interest. Furthermore, the eigenvalues of the active subspace matrix $\boldsymbol{C}$ rank the relative importance of each linear combination. This offers more insight into the input/output relationships than the dimensional analysis alone.

Chapter 3

Inverse regression for ridge recovery

In the previous chapter, we explored how to interpret ridge functions in the context of computational science models by drawing connections between active subspaces and physically-motivated dimension reduction through dimensional analysis. We used active subspaces as a tool for discovering ridge structure since the column space of the matrix $\boldsymbol{C}$ from (2.8) provides a basis for the ridge directions. Computing this matrix requires the gradient of $f(\mathbf{x})$ with respect to the m inputs. For complex models, such as those arising from systems of coupled partial differential equations, it may be impossible to analytically compute the gradient. We can approximate the gradient using adjoints, as was done for the MHD generator model in Chapter 2. However, implementing adjoint capabilities in complex physical models can be difficult and solving the adjoint system is as expensive as a typical forward solve. Alternatively, we can approximate the gradient of f using finite differences, but this requires $m+1$ model evaluations to approximate the m-dimensional gradient. In this chapter, we seek out methods for ridge recovery that do not require gradients.

In regression modeling, subspace-based predictor dimension reduction goes by the name sufficient dimension reduction (SDR) [30, 2, 79]. Techniques for SDR include sliced inverse regression (SIR) [80], sliced average variance estimation (SAVE) [33], ordinary least squares (OLS) [82], and principal Hessian directions (pHd) [81]—among several others. These techniques seek a low-dimensional subspace in the predictor space that is sufficient to statistically characterize the relationship between predictors and response. Related techniques and extensions have enabled feature space dimension reduction in supervised learning models [89, 130, 52].

In this work, we make connections between SDR and ridge functions. We show that in the context of deterministic functions, such as those underlying computational science models, the fundamental conditions of SDR are equivalent to those of ridge functions. We then evaluate and analyze the inverse regression methods SIR and SAVE as candidates for gradient-free ridge recovery in deterministic functions. Recent work by Fornasier, Schnass, and Vybiral [49] and related work by Tyagi and Cevher [123] develop ridge recovery algorithms that exploit a connection between a linear measurement of a gradient vector and the function's directional derivative to estimate $\boldsymbol{A}$ with finite differences. Constantine, Eftekhari, and Wakin [24] exploit a similar connection to estimate active subspaces with directional finite differences. In contrast, SIR- and SAVE-style dimension reduction follows from low-rank structure in the regression's inverse conditional moment matrices; we detail how this difference (gradients versus inverse conditional moments) affects the methods' ability to recover the ridge directions.

Lastly, we analyze the numerical convergence of SIR and SAVE subspaces as the number N of samples increases and derive the expected $\mathcal{O}_p(N^{-1/2})$ rate, where $\mathcal{O}_p$ denotes convergence in probability and the constant depends inversely on associated eigenvalue gap. This provides useful insights into how to obtain accurate estimates of the ridge directions. Moreover, this view enables the development of more efficient numerical integration methods than Monte Carlo for SIR and SAVE, which we examine in Chapter 4.

3.1 Sufficient dimension reduction

We review the essential theory of sufficient dimension reduction (SDR); our notation and development closely follow Cook's *Regression Graphics: Ideas for Studying Regressions through Graphics* [30]. The theory of SDR provides a framework for subspace-based dimension reduction in statistical regression. A regression problem begins with predictor/response pairs $\{[\,\mathbf{x}_i^\top\,,\,y_i\,]\}$, $i = 1, \dots, N$, where $\mathbf{x}_i \in \mathbb{R}^m$ and $y_i \in \mathbb{R}$ denote the vector-valued predictors and scalar-valued response, respectively. These pairs are assumed to be independent realizations from the random vector $[\,\mathbf{x}^\top\,,\,y\,]$ with unknown joint probability measure $\pi_{\mathbf{x},y}$. The object of interest in regression is the conditional random variable $y|\mathbf{x}$; the statistician uses the given predictor/response pairs to estimate statistics of $y|\mathbf{x}$—e.g., moments or quantiles. SDR searches for a subspace of the predictors that is sufficient to describe $y|\mathbf{x}$ with statistics derived from the given predictor/response pairs.

The basic tool of SDR is the dimension reduction subspace (DRS). Consider a random vector $[\mathbf{x}^\top, y]$, and let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ with $n \leq m$ be such that

$$y \perp\!\!\!\perp \mathbf{x}|\boldsymbol{A}^\top\mathbf{x}, \tag{3.1}$$

where $\perp\!\!\!\perp$ denotes independence of y and $\mathbf{x}$. A dimension reduction subspace $\mathcal{S}_{\text{DRS}}$ for $y|\mathbf{x}$ is

$$\mathcal{S}_{\text{DRS}} \;=\; \mathcal{S}_{\text{DRS}}(\boldsymbol{A}) \;=\; \text{colspan}(\boldsymbol{A}), \tag{3.2}$$

where colspan($\boldsymbol{A}$) denotes the column space of the $\boldsymbol{A}$. Equation (3.1) denotes the conditional independence of the response and predictors given $\boldsymbol{A}^\top\mathbf{x}$. We can define a new random variable $y|\boldsymbol{A}^\top\mathbf{x}$ that is the response conditioned on the n-dimensional vector $\boldsymbol{A}^\top\mathbf{x}$. If $\mathcal{S}_{\text{DRS}}(\boldsymbol{A})$ is a DRS for $y|\mathbf{x}$, then the conditional CDF for $y|\mathbf{x}$ is the conditional CDF for $y|\boldsymbol{A}^\top\mathbf{x}$ [30, Chapter 6]. If a regression admits a low-dimensional ($n < m$) DRS, then the predictor dimension can be reduced by considering $y|\boldsymbol{A}^\top\mathbf{x}$.

Note that the matrix $\boldsymbol{A}$ in (3.2) is not unique in defining a DRS [28]. For any matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m\times n}$ such that colspan($\boldsymbol{A}$) = colspan($\boldsymbol{B}$),

$$y \perp\!\!\!\perp \mathbf{x}|\boldsymbol{A}^\top\mathbf{x} \quad\iff\quad y \perp\!\!\!\perp \mathbf{x}|\boldsymbol{B}^\top\mathbf{x}. \tag{3.3}$$

Moreover, it is possible for two distinct subspaces to be DRSs for the same regression. Suppose a particular regression admits the DRS $\mathcal{S}_{\text{DRS}}(\boldsymbol{A})$ with $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $n < m$, and let $\mathbf{v} \in \mathbb{R}^m$ such that $\mathbf{v} \notin \text{colspan}(\boldsymbol{A})$. Consider the $m \times (n+1)$ matrix $\boldsymbol{B} = \begin{bmatrix}\boldsymbol{A} & \mathbf{v}\end{bmatrix}$. The subspace colspan($\boldsymbol{B}$) is also a DRS for the same regression. However, $\mathcal{S}_{\text{DRS}}(\boldsymbol{A})$ is the preferred subspace since $\dim(\mathcal{S}_{\text{DRS}}(\boldsymbol{A})) \leq \dim(\mathcal{S}_{\text{DRS}}(\boldsymbol{B}))$ where $\dim(\cdot)$ denotes the subspace dimension. It is not necessary that a DRS reduce the predictor dimension of the regression. In fact, all regressions admit the trivial DRS, $\mathbb{R}^m$ [30, Chapter 6]. Since the goal of SDR is to reduce the predictor dimension, we are interested in finding the one which has minimum dimension.

A subspace $\mathcal{S}$ is called a minimum DRS for $y|\mathbf{x}$ if it is a DRS and $\dim(\mathcal{S}) \leq \dim(\mathcal{S}_{\mathrm{DRS}})$, where $\mathcal{S}_{\mathrm{DRS}}$ is any DRS for $y|\mathbf{x}$. Since the predictor dimension is finite, a minimum DRS always exists—though the minimum DRS may be $\mathbb{R}^m$—i.e., the full predictor space. Existence of the minimum DRS indicates that all regression problems exhibit a minimum dimension for sufficient dimension reduction. This minimum dimension, referred to as the structural dimension of $y|\mathbf{x}$, represents the largest possible reduction of the predictor space while maintaining the structure of $y|\mathbf{x}$ [29]. Despite uniqueness of the structural dimension, a unique minimum DRS is not guaranteed—as the following example illustrates [28].

Example 1. *Assume that $\mathbf{x} \in \mathbb{R}^2$ has a uniform marginal density on the unit circle. Suppose that the random variable $y|\mathbf{x} = x_1^2 + \epsilon$, where ϵ is random noise that is independent of $\mathbf{x}$. This implies that $\mathcal{S}_{\mathrm{DRS}}([\,1\,,\,0\,]^\top)$ is a DRS for this problem. However, the marginal density of $\mathbf{x}$ is such that $1 = x_1^2 + x_2^2$. This means $y|\mathbf{x} = 1 - x_2^2 + \epsilon$. From this expression, $\mathcal{S}_{\mathrm{DRS}}([\,0\,,\,1\,]^\top)$ is also a DRS. Both of these DRSs have dimension 1 and there does not exist a DRS of dimension 0—i.e., $\{\mathbf{0}\}$ is not a DRS. Therefore, there does not exist a unique minimum DRS for this regression.*

As demonstrated in Example 1, a given regression problem may admit multiple DRSs. A uniquely-defined DRS is required to ensure a well-posed SDR problem. To this end, we define the the central DRS (or simply the central subspace) of $y|\mathbf{x}$ to be the DRS $\mathcal{S}_{y|\mathbf{x}}$ such that $\mathcal{S}_{y|\mathbf{x}} \subseteq \mathcal{S}_{\mathrm{DRS}}$, where $\mathcal{S}_{\mathrm{DRS}}$ is any DRS for $y|\mathbf{x}$. When the central subspace exists it is the intersection of all other DRSs,

$$\mathcal{S}_{y|\mathbf{x}} \;=\; \bigcap \mathcal{S}_{\mathrm{DRS}} \tag{3.4}$$

for all $\mathcal{S}_{\mathrm{DRS}}$ of $y|\mathbf{x}$. The intersection in (3.4) always defines a subspace, but this subspace need not satisfy the conditional independence from (3.1). Therefore, a regression need not admit a central subspace. However, when a central subspace exists, it is the unique DRS of minimum dimension for $y|\mathbf{x}$ [30, Chapter 6].

There are a variety of conditions that ensure the existence of $\mathcal{S}_{y|\mathbf{x}}$ for a given regression problem. We consider the following condition on the marginal density of the predictors.

Theorem 1 ([30]). *Suppose that $\mathcal{S}_1$ and $\mathcal{S}_2$ are two DRSs for $y|\mathbf{x}$ where the marginal measure of the predictors $\pi_{\mathbf{x}}$ exhibits a density function that has support over a convex set. Then, $\mathcal{S}_1 \cap \mathcal{S}_2$ is also a DRS.*

According to Theorem 1, if predictor space has density function with support over a convex set, then any intersection of DRSs will be a DRS—i.e., $\mathcal{S}_{y|\mathbf{x}}$ from (3.4) is a DRS and (hence) the central subspace. In regression practice, this existence condition may be difficult to verify from the given predictor/response pairs. Existence of the central subspace can be proven under other sets of assumptions, including a generalization of Theorem 1 to M-sets [133] and another based on location regressions [30, Chap. 6]. The existence criteria in Theorem 1 is the most pertinent when we employ inverse regression for ridge recovery in Section 3.2.

There are two useful properties of the central subspace that enable convenient transformations. The first involves the effect of affine transformations in the predictor space. Let $\mathbf{z} = \boldsymbol{B}\mathbf{x} + \mathbf{b}$ for full rank $\boldsymbol{B} \in \mathbb{R}^{m\times m}$ and $\mathbf{x} \in \mathbb{R}^m$. If $\text{colspan}(\boldsymbol{A}) = \mathcal{S}_{y|\mathbf{x}}$ for some $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, then $\text{colspan}(\boldsymbol{B}^{-\top}\boldsymbol{A}) = \mathcal{S}_{y|\mathbf{z}}$ [29]. This allows us to assume a standardized predictor space, similar to the assumption from (1.2), without loss of generality. The second property involves mappings of the response. Let $h : \mathbb{R} \to \mathbb{R}$ be a function applied to the responses that produces a new regression problem, $\{[\,\mathbf{x}_i^\top\,,\, h(y_i)\,]\}$, $i = 1, \dots, N$. The central subspace associated with the new regression problem is contained within the original central subspace,

$$\mathcal{S}_{h(y)|\mathbf{x}} \subseteq \mathcal{S}_{y|\mathbf{x}}, \tag{3.5}$$

with equality holding when h is strictly monotone [31]. Equation (3.5) is essential for studying the slicing-based algorithms, SIR and SAVE, for estimating the central subspace, where the mapping h partitions the response space; see Sections 3.1.1 and 3.1.2.

The goal of SDR is to estimate the central subspace for the regression from the given response/predictor pairs.

Problem 2 (SDR problem). *Given response/predictor pairs $\{[\,\mathbf{x}_i^\top\,,\, y_i\,]\}$, with $i = 1, \dots, N$, assumed to be independent draws from a random vector $[\,\mathbf{x}^\top\,,\, y\,]$ with joint probability measure $\pi_{\mathbf{x},y}$,* *compute a basis* $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ *for the central subspace* $\mathcal{S}_{y|\mathbf{x}}$ *of the random variable* $y|\mathbf{x}$.

Next, we review two algorithms that address Problem 2—sliced inverse regression (SIR) [80] and sliced average variance estimation (SAVE) [33]. These algorithms use the given regression data to approximate population moment matrices of the inverse regression $\mathbf{x}|y$.

3.1.1 Sliced inverse regression

Sliced inverse regression (SIR) [80] is an algorithm for approximating the matrix

$$\boldsymbol{C}_{\mathrm{IR}} \;=\; \mathrm{Cov}\left[\mathbb{E}\left[\mathbf{x}|y\right]\right] \tag{3.6}$$

using the given predictor/response pairs. $\boldsymbol{C}_{\mathrm{IR}}$ is defined by the inverse regression function $\mathbb{E}\left[\mathbf{x}|y\right]$, which draws a curve through the m-dimensional predictor space parameterized by the scalar-valued response. If the given regression problem satisfies the linearity condition [80], then

$$\mathrm{colspan}(\boldsymbol{C}_{\mathrm{IR}}) \subseteq \mathcal{S}_{y|\mathbf{x}}. \tag{3.7}$$

In words, the column space of $\boldsymbol{C}_{\mathrm{IR}}$ is a subset of the central subspace for the regression. The linearity condition is satisfied when $\pi_{\mathbf{x}}$ is elliptically symmetric (e.g., a multivariate standard Gaussian) [44].

SIR approximates $\boldsymbol{C}_{\mathrm{IR}}$ using given predictor/response pairs by partitioning—i.e., slicing—the response space. Consider a partition of the observed response space,

$$\min_{1\leq i\leq N} y_i = \tilde{y}_0 < \tilde{y}_1 < \cdots < \tilde{y}_{R-1} < \tilde{y}_R = \max_{1\leq i\leq N} y_i, \tag{3.8}$$

and let $J_r = [\tilde{y}_{r-1}, \tilde{y}_r]$ denote the rth partition for $r = 1, \ldots, R$. Define the function

$$h(y) \;=\; r \quad \text{for} \quad y \in J_r. \tag{3.9}$$

Applying h to the given responses creates a new regression problem $\{[\,\mathbf{x}_i^\top\,,\, h(y_i)\,]\}$, where (3.6) becomes

$$\boldsymbol{C}_{\mathrm{SIR}} \;=\; \mathrm{Cov}\left[\mathbb{E}\left[\mathbf{x}|h(y)\right]\right]. \tag{3.10}$$

The notation $\boldsymbol{C}_{\mathrm{SIR}}$ emphasizes that this matrix is with respect to the sliced version of the original regression problem. Combining (3.7) and (3.5),

$$\operatorname{colspan}(\boldsymbol{C}_{\mathrm{SIR}}) \subseteq \mathcal{S}_{h(y)|\mathbf{x}} \subseteq \mathcal{S}_{y|\mathbf{x}}. \tag{3.11}$$

The sliced partition of the response space and the sliced mapping h bin the predictor/response pairs to enable sample estimates of $\mathbb{E}\left[\mathbf{x}|r\right]$ for $r = 1, \dots, R$. This is the basic idea behind the SIR algorithm; see Algorithm 1. Note that if the response is discrete, then Algorithm 1 produces the maximum likelihood estimator of the central subspace [32].

Eigenvectors of $\boldsymbol{C}_{\mathrm{SIR}}$ associated with nonzero eigenvalues provide a basis for the SIR subspace, $\operatorname{colspan}(\boldsymbol{C}_{\mathrm{SIR}})$. If the approximated eigenvalues $\hat{\lambda}_{n+1}, \dots, \hat{\lambda}_m$ from Algorithm 1 are small, then $m \times n$ matrix $\hat{\boldsymbol{A}}$ approximates a basis for this subspace. However, determining the appropriate value of n requires care. Li [80] and Cook [33] propose significance tests based on the distribution of the average of the $m - n$ trailing estimated eigenvalues. These testing methods also apply to the SAVE algorithm in Section 3.1.2.

For a fixed number of slices, SIR has been shown to be $N^{-1/2}$-consistent for approximating $\operatorname{colspan}(\boldsymbol{C}_{\mathrm{SIR}})$ [80]. In principle, increasing the number of slices may provide improved estimation of the central DRS. The number of slices should be chosen such that there are enough samples in each slice to estimate the conditional expectations accurately. For this reason, Li [80] suggests constructing slices such that the response samples are distributed nearly equally.

3.1.2 Sliced average variance estimation

SAVE uses the variance of the inverse regression. Li [80] recognized the potential for $\operatorname{Cov}\left[\mathbf{x}|y\right]$ to provide insights into the central subspace by noting that

$$\mathbb{E}\left[\operatorname{Cov}\left[\mathbf{x}|y\right]\right] = \operatorname{Cov}\left[\mathbf{x}\right] - \operatorname{Cov}\left[\mathbb{E}\left[\mathbf{x}|y\right]\right] = \boldsymbol{I} - \operatorname{Cov}\left[\mathbb{E}\left[\mathbf{x}|y\right]\right], \tag{3.15}$$

under the assumption of standardized predictors. Equation (3.15) implies that $\mathbb{E}\left[\boldsymbol{I} - \operatorname{Cov}\left[\mathbf{x}|y\right]\right]$ may be useful in addressing Problem 2. Cook suggests using $\mathbb{E}\left[(\boldsymbol{I} - \operatorname{Cov}\left[\mathbf{x}|y\right])^2\right]$, which has nonnegative

Algorithm 1 Sliced inverse regression [80]

Given: N samples $\{[\,\mathbf{x}_i^\top\,,\, y_i\,]\}$, $i = 1, \ldots, N$, drawn independently according to $p_{\mathbf{x},y}$.

(1) Partition the response space as in (3.8), and let $J_r = [\tilde{y}_{r-1}, \tilde{y}_r]$ for $r = 1, \ldots, R$. Let $\mathcal{I}_r \subset \{1, \ldots, N\}$ be the set of indices i for which $y_i \in J_r$ and define N_r to be the cardinality of $\mathcal{I}_r$.

(2) For $r = 1, \ldots, R$, compute the sample mean $\hat{\boldsymbol{\mu}}_h(r)$ of the predictors whose associated responses are in J_r,

$$\hat{\boldsymbol{\mu}}_h(r) = \frac{1}{N_r} \sum_{i \in \mathcal{I}_r} \mathbf{x}_i. \tag{3.12}$$

(3) Compute the weighted sample covariance matrix

$$\hat{\boldsymbol{C}}_{\text{SIR}} \; = \; \frac{1}{N} \sum_{r=1}^{R} N_r \, \hat{\boldsymbol{\mu}}_h(r) \hat{\boldsymbol{\mu}}_h(r)^\top. \tag{3.13}$$

(4) Compute the eigendecomposition,

$$\hat{\boldsymbol{C}}_{\text{SIR}} \; = \; \hat{\boldsymbol{W}} \hat{\boldsymbol{\Lambda}} \hat{\boldsymbol{W}}^\top, \tag{3.14}$$

where the eigenvalues are in descending order $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \cdots \geq \hat{\lambda}_m \geq 0$ and the eigenvectors are orthonormal.

Output: The first n eigenvectors of $\hat{\boldsymbol{C}}_{\text{SIR}}$, denoted by $\hat{\boldsymbol{A}}$.

eigenvalues [33]. Define

$$\boldsymbol{C}_{\text{AVE}} \;=\; \mathbb{E}\left[(\boldsymbol{I} - \text{Cov}\left[\mathbf{x}|y\right])^2\right]. \tag{3.16}$$

Under the linearity condition [80] and the constant covariance condition [31], the column space of $\boldsymbol{C}_{\text{AVE}}$ is contained within the central subspace,

$$\text{colspan}(\boldsymbol{C}_{\text{AVE}}) \subseteq \mathcal{S}_{y|\mathbf{x}}. \tag{3.17}$$

Both of these conditions are satisfied when $\pi_{\mathbf{x}}$ is elliptically symmetric [44].

Using the partition (3.8) and the map (3.9),

$$\boldsymbol{C}_{\text{SAVE}} \;=\; \mathbb{E}\left[(\boldsymbol{I} - \text{Cov}\left[\mathbf{x}|h(y)\right])^2\right]. \tag{3.18}$$

The notation $\boldsymbol{C}_{\text{SAVE}}$ indicates application to the sliced version of the original regression problem. Combining (3.17) and (3.5),

$$\text{colspan}(\boldsymbol{C}_{\text{SAVE}}) \;\subseteq\; \mathcal{S}_{h(y)|\mathbf{x}} \;\subseteq\; \mathcal{S}_{y|\mathbf{x}}. \tag{3.19}$$

Algorithm 2 shows the SAVE algorithm, which computes a basis for the column span of the sample estimate $\hat{\boldsymbol{C}}_{\text{SAVE}}$. This basis is a $N^{-1/2}$-consistent estimate of colspan$(\boldsymbol{C}_{\text{SAVE}})$ [31]. Increasing the number of slices improves the estimate but suffers the same drawbacks as SIR. In practice, SAVE performs poorly compared to SIR when few predictor/response pairs are available. This is due to difficulties approximating the covariances within the slices using too few samples. For this reason, Cook [32] suggests trying both methods to approximate the central DRS.

3.2 Inverse regression as ridge recovery

This section develops SIR (Algoroithm 1) and SAVE (Algorithm 2) as tools for ridge recovery (Problem 1). These algorithms seek to construct a basis for the optimal dimension reduction subspace—i.e., $\boldsymbol{A}$ that satisfies the conditional independence requirement from (3.1). The next theorem connects the conditional independence underlying sufficient dimension reduction to ridge functions in deterministic functions.

```
Algorithm 2 Sliced average variance estimation [31]
Given: $N$ samples $\{[\mathbf{x}_i^\top, y_i]\}$, $i = 1, \dots, N$, drawn independently according to $p_{\mathbf{x},y}$.

(1) Define a partition of the response space as in (3.8), and let $J_r = [\tilde{y}_{r-1}, \tilde{y}_r]$ for $r = 1, \dots, R$.
    Let $\mathcal{I}_r \subset \{1, \dots, N\}$ be the set of indices $i$ for which $y_i \in J_r$ and define $N_r$ to be the
    cardinality of $\mathcal{I}_r$.
(2) For $r = 1, \dots, R$,
    (a) Compute the sample mean $\hat{\boldsymbol{\mu}}_h(r)$ of the predictors whose associated responses are in
        the $J_r$,
            $\hat{\boldsymbol{\mu}}_h(r) = \frac{1}{N_r} \sum_{i \in \mathcal{I}_r} \mathbf{x}_i.$    (3.20)
    (b) Compute the sample covariance $\hat{\boldsymbol{\Sigma}}_h(r)$ of the predictors whose associated responses
        are in $J_r$,
            $\hat{\boldsymbol{\Sigma}}_h(r) = \frac{1}{N_r - 1} \sum_{i \in \mathcal{I}_r} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_h(r)) (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_h(r))^\top$    (3.21)
(3) Compute the matrix,
        $\hat{\boldsymbol{C}}_{\text{SAVE}} = \frac{1}{N} \sum_{r=1}^{R} N_r \left(\boldsymbol{I} - \hat{\boldsymbol{\Sigma}}_h(r)\right)^2.$    (3.22)
(4) Compute the eigendecomposition,
        $\hat{\boldsymbol{C}}_{\text{SAVE}} = \hat{\boldsymbol{W}} \hat{\boldsymbol{\Lambda}} \hat{\boldsymbol{W}}^\top,$    (3.23)
    where the eigenvalues are in descending order $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \cdots \geq \hat{\lambda}_m \geq 0$ and the eigenvectors
    are orthonormal.
Output: The first $n$ eigenvectors of $\hat{\boldsymbol{C}}_{\text{SAVE}}$, denoted by $\hat{\boldsymbol{A}}$.
```

Theorem 2. *Let (Ω, Σ, P) be a probability triple. Suppose that $\mathbf{x} : \Omega \to \mathbb{R}^m$ and $y : \Omega \to \mathbb{R}$ are random variables related by a measurable function $f : \mathbb{R}^m \to \mathbb{R}$ so that $y = f(\mathbf{x})$. Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ be a constant matrix. Then $y \perp\!\!\!\perp \mathbf{x} | \boldsymbol{A}^\top \mathbf{x}$ if and only if $y = g(\boldsymbol{A}^\top \mathbf{x})$ where $g : \mathbb{R}^n \to \mathbb{R}$ is a measurable function.*

The proof is in Appendix A.1. Theorem 2 states that conditional independence of the inputs and output in a deterministic function is equivalent to the function being a ridge function. This provides a subspace-based perspective on ridge functions that uses the DRS as the foundation. That is, the directions of the ridge function are relatively unimportant compared to the subspace they span when capturing ridge structure of f with sufficient dimension reduction.

The central subspace from (3.4) corresponds to the unique subspace of smallest dimension that completely describes the ridge structure of f. Moreover, our focused concern on the subspace instead of the precise basis implies that we can assume standardized inputs $\mathbf{x}$ (see (1.2)) without loss of generality, which simplifies discussion of the inverse conditional moment matrices that underlie the SIR and SAVE algorithm for ridge recovery.

Theorem 1 guarantees existence of the central subspace in regression when the marginal density of the predictors has convex support. However, this condition is typically difficult or impossible to verify for the regression problem. In contrast, the deterministic function is accompanied by a known input measure $\pi_{\mathbf{x}}$. In practice, a relatively non-informative measure is used such as a multivariate Gaussian or uniform density on a hyper-rectangle defined by the ranges of each physical input parameter. Such choices for modeling input uncertainty satisfy Theorem 1 and guarantee the existence of the central subspace.

The input probability measure can influence the structure of the central subspace. For example, let $\mathbf{a}, \mathbf{b} \in \mathbb{R}^m$ be constant vectors pointing in different directions and consider the function

$$y \;=\; f(\mathbf{x}) \;=\; \begin{cases} (\mathbf{a}^\top \mathbf{x})^2 & \text{if } \mathbf{x}_i > 0 \text{ for } i = 1, \dots, m \\ (\mathbf{b}^\top \mathbf{x})^2 & \text{otherwise.} \end{cases} \tag{3.24}$$

If $\pi_{\mathbf{x}}$ has a density function with support only for positive values of $\mathbf{x}$ (e.g., uniform over $[0, 1]^m$), then the central subspace is span $\mathbf{a}$. Alternatively, if the input density has support over all of $\mathbb{R}^m$ (e.g., multivariate Gaussian), then the central subspace is span $\mathbf{a}, \mathbf{b}$. For this reason, we denote the central subspace for a deterministic function by $\mathcal{S}_{f,\pi_{\mathbf{x}}}$ to emphasize that this subspace is a property of the given function and the associated input probability measure.

In the next two sections, we reformulate the key matrices underlying the SIR and SAVE algorithms (Equations (3.6) and (3.16), respectively) to enable proper analysis of these algorithms for deterministic ridge recovery. However, we must first understand how to interpret one of the basic components of these matrices in the deterministic framework. Both algorithms are based on statistical moments of the inverse regression $\mathbf{x}|y$. The deterministic analog of this object is the inverse image of f for the output value y,

$$f^{-1}(y) \;=\; \{\mathbf{x} \in \mathbb{R}^m \,:\, f(\mathbf{x}) = y\}\,. \tag{3.25}$$

Unlike the random vector inverse regression, f^{-1} is a fixed set determined by f's contours. Furthermore, the inverse image begets the conditional probability measure $\pi_{\mathbf{x}|y}$, which is the restriction of $\pi_{\mathbf{x}}$ to the set $f^{-1}(y)$ [13].

3.2.1 Sliced inverse regression for ridge recovery

For deterministic functions, we can write $\boldsymbol{C}_{\mathrm{IR}}$ from (3.6) as an integral against probability measures:

$$\boldsymbol{C}_{\mathrm{IR}} \;=\; \int \boldsymbol{\mu}(y)\,\boldsymbol{\mu}(y)^\top \,\mathrm{d}\pi_y(y) \tag{3.26}$$

where the conditional expectation over the inverse image $f^{-1}(y)$ is

$$\boldsymbol{\mu}(y) \;=\; \int \mathbf{x}\,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}). \tag{3.27}$$

This term represents the average of all input values that map to a fixed value of the output. The integration in (3.26) is performed with respect to π_y. This is an induced probability measure on the output space. That is, the given input probability measure $\pi_{\mathbf{x}}$ is propagated forward through f and induces π_y on the output space.

To understand how $\boldsymbol{C}_{\text{IR}}$ can be used for dimension reduction in deterministic functions, consider the following. For $\mathbf{w} \in \mathbb{R}^m$ with unit norm,

$$\mathbf{w}^\top \boldsymbol{C}_{\text{IR}} \mathbf{w} \;=\; \mathbf{w}^\top \left(\int \boldsymbol{\mu}(y)\, \boldsymbol{\mu}(y)^\top \, \mathrm{d}\pi_y(y) \right) \mathbf{w} \;=\; \int \left(\boldsymbol{\mu}(y)^\top \mathbf{w} \right)^2 \mathrm{d}\pi_y(y). \tag{3.28}$$

If $\mathbf{w} \in \operatorname{null}(\boldsymbol{C}_{\text{IR}})$, then one possibility is that $\boldsymbol{\mu}(y)$ is orthogonal to $\mathbf{w}$ for all y. The following theorem relates this case to possible ridge structure in f.

Theorem 3. *Let $f : \mathbb{R}^m \to \mathbb{R}$ with input probability measure $\pi_{\mathbf{x}}$ admit a central subspace $\mathcal{S}_{f,\pi_{\mathbf{x}}}$, and assume $\pi_{\mathbf{x}}$ admits an elliptically symmetric and standardized density function. Then,* $\operatorname{colspan}(\boldsymbol{C}_{\text{IR}}) \subseteq \mathcal{S}_{f,\pi_{\mathbf{x}}}$.

See Appendix A.2 for the proof. Theorem 3 shows that a basis for the range of $\boldsymbol{C}_{\text{IR}}$ can be used to estimate f's central subspace. However, this idea has two important limitations. First, by (3.28), we can write the inner product in the rightmost integrand in terms of the cosine of the angle between the vectors $\boldsymbol{\mu}(y)$ and $\mathbf{w}$,

$$\mathbf{w}^\top \boldsymbol{C}_{\text{IR}} \mathbf{w} \;=\; \int ||\boldsymbol{\mu}(y)||_2^2 \cos^2(\theta(y))\, \mathrm{d}\pi_y(y), \tag{3.29}$$

where $\theta(y)$ is the angle between $\boldsymbol{\mu}(y)$ and $\mathbf{w}$. Theorem 3 uses orthogonality of $\boldsymbol{\mu}(y)$ and $\mathbf{w}$—i.e., $\cos(\theta(y)) = 0$ for all y—to show containment of the column space of $\boldsymbol{C}_{\text{IR}}$ within the central subspace; however, the integrand in (3.29) also contains the squared 2-norm of $\boldsymbol{\mu}(y)$, which does not depend on $\mathbf{w}$. If $\boldsymbol{\mu}(y) = \mathbf{0}$ for all y, then $\mathbf{w}^\top \boldsymbol{C}_{\text{IR}} \mathbf{w} = 0$ for all $\mathbf{w}$. Consider the following example.

Example 2. *Assume $\mathbf{x} \in \mathbb{R}^2$ is weighted with a bivariate standard Gaussian. Let $y = f(\mathbf{x}) = x_1 x_2$. For any value of y, $\mathbf{x} \in f^{-1}(y)$ implies $-\mathbf{x} \in f^{-1}(y)$. Therefore, $\boldsymbol{\mu}(y) = \mathbf{0}$ for all y, and $\boldsymbol{C}_{\text{IR}} = \mathbf{0}$. But y is not constant over $\mathbb{R}^2$. Thus, $\{\mathbf{0}\} = \operatorname{colspan}(\boldsymbol{C}_{\text{IR}}) \subset \mathcal{S}_{f,\pi_{\mathbf{x}}} = \mathbb{R}^2$.*

This example shows how $\boldsymbol{C}_{\text{IR}}$, as a tool for ridge recovery, can mislead the practitioner by suggesting ridge structure in a function that is not a ridge function. This could lead one to ignore input space directions that should not be ignored. Note that if we shift the function such that

$y = f(\mathbf{x}) = (x_1 + c_1)(x_2 + c_2)$ for some constants $c_1, c_2 \neq 0$, then the symmetry is broken and $\boldsymbol{\mu}(y) \neq 0$ for all y. In this case, $\boldsymbol{C}_{\text{IR}}$ will recover the central subspace—i.e., all of $\mathbb{R}^2$.

The second limitation of $\boldsymbol{C}_{\text{IR}}$ for ridge recovery follows from the required elliptic symmetry of the input density. This assumption is satisfied if the density is a multivariate Gaussian, but it is violated if it is uniform over the m-dimension hypercube. If f is a ridge function and $\mathbf{w} \in \text{null}(\boldsymbol{A}^\top)$, then $\mathbf{x} \in f^{-1}(y)$ implies $\mathbf{x} + \mathbf{w} \in f^{-1}(y)$ so that $f^{-1}(y)$ can be expressed as the union of lines parallel to $\mathbf{w}$. If the inputs are weighted by an elliptically symmetric density, then the expectation over $f^{-1}(y)$ will be centered such that $\boldsymbol{\mu}(y)$ is orthogonal to $\mathbf{w}$. If the inputs do not have an elliptically symmetric density, then the weighting can cause the conditional expectation to deviate in the direction of $\mathbf{w}$. The magnitude of this deviation also depends on the magnitude of the conditional expectation $||\boldsymbol{\mu}(y)||_2$.

Next, we examine the sliced approximation of $\boldsymbol{C}_{\text{IR}}$. Recall the output partition from (3.8) and the slicing map $h(y)$ (3.9). Applying h to the deterministic function $y = f(\mathbf{x})$ produces the discretized function $r = h(y) = h(f(\mathbf{x}))$, where $r \in \{1, \ldots, R\}$. The output space of h is weighted by the probability mass function

$$\omega(r) \;=\; \int_{J_r} \mathrm{d}\pi_y(y), \qquad r \in \{1, \ldots, R\}. \tag{3.30}$$

Without loss of generality, we assume that the slices are constructed such that $\omega(r) > 0$ for all r. If $\omega(r) = 0$ for some r, then we can combine this slice with an adjacent slice. The conditional expectation for the sliced output is

$$\boldsymbol{\mu}_h(r) \;=\; \int \mathbf{x}\, \mathrm{d}\pi_{\mathbf{x}|r}(\mathbf{x}), \tag{3.31}$$

where $\pi_{\mathbf{x}|r}$ is the conditional measure defined over the set $f^{-1}(h^{-1}(r)) = \{\, \mathbf{x} \in \mathbb{R}^m \,:\, h(f(\mathbf{x})) = h(y) = r \,\}$. Using (3.30) and (3.31), the sliced version of $\boldsymbol{C}_{\text{IR}}$ is

$$\boldsymbol{C}_{\text{SIR}} \;=\; \sum_{r=1}^{R} \omega(r)\, \boldsymbol{\mu}_h(r)\, \boldsymbol{\mu}_h(r)^\top. \tag{3.32}$$

By Theorem 2, properties of the central subspace extend to the ridge recovery problem. This includes containment of the central subspace under any mapping of the output,

$$\operatorname{colspan}(\boldsymbol{C}_{\mathrm{SIR}}) \;\subseteq\; \mathcal{S}_{h\circ f,\pi_{\mathbf{x}}} \;\subseteq\; \mathcal{S}_{f,\pi_{\mathbf{x}}}. \tag{3.33}$$

By approximating $\boldsymbol{C}_{\mathrm{SIR}}$, we obtain an approximation of part of f's central subspace. An important corollary of (3.33) is that the rank of $\boldsymbol{C}_{\mathrm{SIR}}$ is bounded above by the dimension of $\mathcal{S}_{f,\pi_{\mathbf{x}}}$.

Note that $\boldsymbol{C}_{\mathrm{SIR}}$ from (3.32) is a finite sum approximation of the integral with respect to π_y in $\boldsymbol{C}_{\mathrm{IR}}$ from (3.26). Since $f^{-1}(h^{-1}(r)) = \cup_{y\in J_r} f^{-1}(y)$, then $\boldsymbol{\mu}_h(r)$ is the average of the conditional expectations with $y \in J_r$. That is,

$$\boldsymbol{\mu}_h(r) \;=\; \int_{J_r} \boldsymbol{\mu}(y)\,\mathrm{d}\pi_y(y). \tag{3.34}$$

Therefore, $\boldsymbol{C}_{\mathrm{SIR}}$ approximates $\boldsymbol{C}_{\mathrm{IR}}$ by a weighted sum of the average values of $\boldsymbol{\mu}(y)$ within each slice. If $\boldsymbol{\mu}(y)$ is continuous almost everywhere with respect to π_y, then $\boldsymbol{C}_{\mathrm{IR}}$ is Riemann-integrable [48, Chap. 2]. This ensures that sum approximations using the supremum and infimums of $\boldsymbol{\mu}(y)$ over each slice converge to the same value. By the sandwich theorem, the average value will converge to this value as well [1]. Therefore, $\boldsymbol{C}_{\mathrm{SIR}}$ is a Riemann sum approximation of $\boldsymbol{C}_{\mathrm{IR}}$; as the number of slices R increases, $\boldsymbol{C}_{\mathrm{SIR}}$ converges to $\boldsymbol{C}_{\mathrm{IR}}$. We explore the nature of this approximation in more detail in Chapter 4.

We next study the asymptotic convergence of Algorithm 1 for ridge recovery. To generate the data for Algorithm 1, we choose N points $\{\mathbf{x}_i\}$ in the input space consistent with $\pi_{\mathbf{x}}$. For each $\mathbf{x}_i$, we query the function to produce the corresponding output $y_i = f(\mathbf{x}_i)$. In the computational science context, this corresponds to running the simulation model at particular sets of inputs. If we choose each $\mathbf{x}_i$ independently according to $\pi_{\mathbf{x}}$, then we can analyze SIR as a Monte Carlo method for estimating $\boldsymbol{C}_{\mathrm{SIR}}$ from (3.32). Given the input/output pairs, Algorithm 1 constructs the random matrix $\hat{\boldsymbol{C}}_{\mathrm{SIR}}$. To be clear, $\hat{\boldsymbol{C}}_{\mathrm{SIR}}$ is a random estimate of $\boldsymbol{C}_{\mathrm{SIR}}$ because of how we chose the points $\{\mathbf{x}_i\}$—not because of any randomness in the map f. Eigenpairs derived from $\hat{\boldsymbol{C}}_{\mathrm{SIR}}$ are also random, and the convergence analysis for Algorithm 1 is probabilistic.

The convergence depends on the smallest number of samples per slice over all the slices:

$$N_{r_{\min}} \;=\; \min_{1 \le r \le R} N_r, \tag{3.35}$$

where N_r is from Algorithm 1. Recall that the slices are assumed to be constructed such that $\omega(r) > 0$. Thus, $N_{r_{\min}} > 0$ with probability 1 as $N \to \infty$. The following theorem shows that the eigenvalues of $\hat{\boldsymbol{C}}_{\text{SIR}}$ converge to those of $\boldsymbol{C}_{\text{SIR}}$ in a mean-squared sense.

Theorem 4. *Assume that Algorithm 1 has been applied to the data set $\{[\mathbf{x}_i^\top, y_i]\}$, with $i = 1, \dots, N$, where the $\mathbf{x}_i$ are drawn independently according to $\pi_{\mathbf{x}}$ and $y_i = f(\mathbf{x}_i)$ are point evaluations of f. Then, for $k = 1, \dots, m$,*

$$\mathbb{E}\left[\left(\lambda_k(\boldsymbol{C}_{\text{SIR}}) - \lambda_k(\hat{\boldsymbol{C}}_{\text{SIR}})\right)^2\right] \;=\; \mathcal{O}(N_{r_{\min}}^{-1}) \tag{3.36}$$

where $\lambda_k(\cdot)$ denotes the kth eigenvalue of the given matrix.

See Appendix A.3 for the proof. In words, the mean-squared error in the eigenvalues of $\hat{\boldsymbol{C}}_{\text{SIR}}$ decays at a $N_{r_{\min}}^{-1}$ rate. Since $\omega(r) > 0$ for all r, $N_{r_{\min}} \to \infty$ as $N \to \infty$. Moreover, the convergence rate suggests that one should choose the slices in Algorithm 1 such that the same number of samples appears in each slice. This maximizes $N_{r_{\min}}$ and reduces the error in the eigenvalues.

An important consequence of Theorem 4 is that the column space of the finite-sample $\hat{\boldsymbol{C}}_{\text{SIR}}$ is not contained in f's central subspace. In fact, due to finite sampling, $\hat{\boldsymbol{C}}_{\text{SIR}}$ is not precisely low-rank. With a fixed number of samples, it is difficult to distinguish the effects of finite sampling from actual variability in f. However, the eigenvalue convergence implies that one can potentially devise practical tests for low-rank-ness based on sets of samples with increasing size.

The next theorem shows the value of understanding the approximation errors in the eigenvalues for quantifying the approximation errors in the subspaces. We measure convergence of the subspace estimates using the subspace distance [64, Chapter 2.5],

$$\text{dist}\left(\text{ran}(\boldsymbol{A}),\, \text{ran}(\hat{\boldsymbol{A}})\right) \;=\; \left\|\boldsymbol{A}\boldsymbol{A}^\top - \hat{\boldsymbol{A}}\hat{\boldsymbol{A}}^\top\right\|_2, \tag{3.37}$$

where $\boldsymbol{A}, \hat{\boldsymbol{A}}$ are the first n eigenvectors of $\boldsymbol{C}_{\text{SIR}}$ and $\hat{\boldsymbol{C}}_{\text{SIR}}$, respectively. The distance metric (3.37) is the principal angle between the subspaces colspan($\boldsymbol{A}$) and colspan($\hat{\boldsymbol{A}}$).

Theorem 5. *Assume the same conditions from Theorem 4. Then, for sufficiently large N,*

$$\operatorname{dist}\left(\operatorname{ran}(\boldsymbol{A}),\, \operatorname{ran}(\hat{\boldsymbol{A}})\right) \;=\; \frac{1}{\lambda_n(\boldsymbol{C}_{\mathrm{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\mathrm{SIR}})}\,\mathcal{O}_p(N_{r_{\min}}^{-1/2}), \tag{3.38}$$

where $\mathcal{O}_p$ denotes convergence in probability.

Appendix A.4 contains the proof. The subspace error decays with asymptotic rate $N_{r_{\min}}^{-1/2}$. The more interesting result from Theorem 5 is the inverse relationship between the subspace error and the magnitude of the gap between the nth and $(n+1)$th eigenvalues. That is, a large gap between eigenvalues suggests a better estimate of the subspace for a fixed number of samples. We do not hide this factor in the $\mathcal{O}_p$ notation to emphasize the importance of Theorem 4, which provides insights into the accuracy of the estimated eigenvalues of $\boldsymbol{C}_{\mathrm{SIR}}$.

3.2.2 SAVE for ridge recovery

Similar to (3.26), we express $\boldsymbol{C}_{\mathrm{AVE}}$ from (3.16) as an integral,

$$\boldsymbol{C}_{\mathrm{AVE}} \;=\; \int (\boldsymbol{I} - \boldsymbol{\Sigma}(y))^2 \,\mathrm{d}\pi_y(y). \tag{3.39}$$

The conditional covariance $\boldsymbol{\Sigma}(y)$ in (3.39) is an integral against the conditional probability measure $\pi_{\mathbf{x}|y}$,

$$\boldsymbol{\Sigma}(y) \;=\; \int (\mathbf{x} - \boldsymbol{\mu}(y))\,(\mathbf{x} - \boldsymbol{\mu}(y))^\top \,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}). \tag{3.40}$$

To see the relationship between the $\boldsymbol{C}_{\mathrm{AVE}}$ matrix and ridge functions, let $\mathbf{w} \in \mathbb{R}^m$ with unit norm:

$$\mathbf{w}^\top \boldsymbol{C}_{\mathrm{AVE}} \mathbf{w} \;=\; \mathbf{w}^\top \left(\int (\boldsymbol{I} - \boldsymbol{\Sigma}(y))^2 \,\mathrm{d}\pi_y(y) \right) \mathbf{w} \;=\; \int ||(\boldsymbol{I} - \boldsymbol{\Sigma}(y))\,\mathbf{w}||_2^2 \,\mathrm{d}\pi_y(y). \tag{3.41}$$

Equation (3.41) relates the column space of $\boldsymbol{C}_{\mathrm{AVE}}$ to the ridge structure in f.

Theorem 6. *Let $f : \mathbb{R}^m \to \mathbb{R}$ with input probability measure $\pi_{\mathbf{x}}$ admit a central subspace $\mathcal{S}_{f,\pi_{\mathbf{x}}}$, and assume $\pi_{\mathbf{x}}$ admits an elliptically symmetric and standardized density function. Then,* $\operatorname{colspan}(\boldsymbol{C}_{\mathrm{AVE}}) \subseteq \mathcal{S}_{f,\pi_{\mathbf{x}}}$.

See the proof in Appendix A.5. This result shows the usefulness of $\boldsymbol{C}_{\text{AVE}}$ for revealing ridge structure in deterministic functions: by estimating the column space of $\boldsymbol{C}_{\text{AVE}}$, we obtain an estimate of a subspace of f's central subspace. However, $\boldsymbol{C}_{\text{AVE}}$ suffers two similar pitfalls as $\boldsymbol{C}_{\text{IR}}$. First, $\boldsymbol{C}_{\text{AVE}}$ can mislead the practitioner by suggesting ridge structure that does not exist—i.e., when $\text{colspan}(\boldsymbol{C}_{\text{AVE}}) \subset \mathcal{S}_{f,\pi_{\mathbf{x}}}$—as the following example illustrates.

Example 3. *Assume $\mathbf{x} \in \mathbb{R}^2$ is weighted by a bivariate standard Gaussian. Let*

$$y \;=\; f(\mathbf{x}) \;=\; \begin{cases} y_1 & \text{if } ||\mathbf{x}||_2 \le r_1 \text{ or } ||\mathbf{x}||_2 \ge r_2, \\ \\ y_2 & \text{if } r_1 < ||\mathbf{x}||_2 < r_2, \end{cases} \tag{3.42}$$

for some $0 < r_1 < r_2$. This functions looks like a bullseye with the central circle and outer ring mapping to y_1 and the middle ring mapping to y_2. By adjusting r_1 and r_2, we can obtain

$$\boldsymbol{\Sigma}(y_1) \;=\; \boldsymbol{\Sigma}(y_2) \;=\; \boldsymbol{I}. \tag{3.43}$$

Note that $\boldsymbol{\mu}(y_1) = \boldsymbol{\mu}(y_2) = 0$ for all choices of r_1 and r_2. Then,

$$\begin{aligned} \boldsymbol{\Sigma}(y_1) \;&=\; \int \mathbf{x}\mathbf{x}^\top \, \mathrm{d}\pi_{\mathbf{x}|y_1}(\mathbf{x}), \\ &=\; \left(1 + \frac{r_2^2 e^{-r_2^2/2} - r_1^2 e^{-r_1^2/2}}{2\left(1 + e^{-r_2^2/2} - e^{-r_1^2/2}\right)}\right) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \end{aligned} \tag{3.44}$$

and

$$\begin{aligned} \boldsymbol{\Sigma}(y_2) \;&=\; \int \mathbf{x}\mathbf{x}^\top \, \mathrm{d}\pi_{\mathbf{x}|y_2}(\mathbf{x}), \\ &=\; \left(1 + \frac{r_2^2 e^{-r_2^2/2} - r_1^2 e^{-r_1^2/2}}{2\left(e^{-r_2^2/2} - e^{-r_1^2/2}\right)}\right) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \end{aligned} \tag{3.45}$$

Thus, (3.43) *holds when $r_1^2 e^{-r_1^2/2} = r_2^2 e^{-r_2^2/2}$, provided that $0 < r_1 < r_2$. Choosing r_1 and r_2 that satisfy these requirements results in $\boldsymbol{C}_{\text{AVE}} = \mathbb{E}\left[(\boldsymbol{I} - \text{Cov}\left[\mathbf{x}|y\right])^2\right] = \mathbb{E}\left[(\boldsymbol{I} - \boldsymbol{I})^2\right] = \mathbb{E}\left[\mathbf{0}\right] = \mathbf{0}$. However, we can see from inspection of* (3.42) *that $\mathcal{S}_{f,\pi_{\mathbf{x}}} = \mathbb{R}^2$. Thus, $\text{colspan}(\boldsymbol{C}_{\text{AVE}}) \subset \mathcal{S}_{f,\pi_{\mathbf{x}}}$.*

Example 3 explores one way in which $\boldsymbol{C}_{\text{AVE}}$ would suggest that low-dimensional structure in a function that is not present. Note that the key feature of this function that results in a degenerate

$\boldsymbol{C}_{\text{AVE}}$ matrix is its rotational symmetry. The symmetry in this function would also fool $\boldsymbol{C}_{\text{IR}}$. In fact, it can be shown that, in general,

$$\text{colspan}(\boldsymbol{C}_{\text{IR}}) \subseteq \text{colspan}(\boldsymbol{C}_{\text{AVE}}), \tag{3.46}$$

which suggests that this sort of false positive is less likely to occur with $\boldsymbol{C}_{\text{AVE}}$ than with $\boldsymbol{C}_{\text{IR}}$ [79, Chap. 5]. The exhaustiveness of $\boldsymbol{C}_{\text{AVE}}$ in capturing central subspace can be proven provided that at least one of $\mathbb{E}\left[\mathbf{w}^\top\mathbf{x}|y\right]$ and $\text{Var}\left[\mathbf{w}^\top\mathbf{x}|y\right]$ are nondegenerate—i.e., explicitly depends on y—for all $\mathbf{w} \in \mathcal{S}_{f,\pi_{\mathbf{x}}}$. Notice that the function in Example 3 agrees with this statement.

The second limitation of $\boldsymbol{C}_{\text{AVE}}$ arises from the elliptic symmetry requirement on the input density. When this density function is not elliptically symmetric, we cannot guarantee that the column space of $\boldsymbol{C}_{\text{AVE}}$ is contained within the central subspace. Thus, a basis for the column space of $\boldsymbol{C}_{\text{AVE}}$ could be contaminated by effects of $\pi_{\mathbf{x}}$—independent of whether or not f is a ridge function.

Next, we consider the sliced version of $\boldsymbol{C}_{\text{AVE}}$. We use the same slicing function h from (3.9) to approximate $\boldsymbol{C}_{\text{AVE}}$ from (3.40). The sliced approximation of $\boldsymbol{C}_{\text{AVE}}$ is

$$\boldsymbol{C}_{\text{SAVE}} \;=\; \sum_{r=1}^{R} \omega(r)\; \left(\boldsymbol{I} - \boldsymbol{\Sigma}_h(r)\right)^2, \tag{3.47}$$

where $\omega(r)$ is the probability mass function from (3.30) and

$$\boldsymbol{\Sigma}_h(r) \;=\; \int \left(\mathbf{x} - \boldsymbol{\mu}_h(r)\right)\, \left(\mathbf{x} - \boldsymbol{\mu}_h(r)\right)^\top \, \mathrm{d}\pi_{\mathbf{x}|r}(\mathbf{x}). \tag{3.48}$$

By containment of the central subspace,

$$\text{colspan}(\boldsymbol{C}_{\text{SAVE}}) \;\subseteq\; \mathcal{S}_{h\circ f,\pi_{\mathbf{x}}} \;\subseteq\; \mathcal{S}_{f,\pi_{\mathbf{x}}}. \tag{3.49}$$

We can interpret $\boldsymbol{C}_{\text{SAVE}}$ as a Riemann sum approximation of $\boldsymbol{C}_{\text{AVE}}$ using a similar argument as in Section 3.2.1. An important corollary of (3.49) is that the rank of $\boldsymbol{C}_{\text{SAVE}}$ is bounded above by the dimension of f's central subspace.

Algorithm 2 computes a sample approximation of $\boldsymbol{C}_{\text{SAVE}}$, denoted $\hat{\boldsymbol{C}}_{\text{SAVE}}$, using given input/output pairs $\{[\,\mathbf{x}_i^\top\,,\, y_i\,]\}$. When the $\mathbf{x}_i$ are sampled independently according to $\pi_{\mathbf{x}}$ and each

$y_i = f(\mathbf{x}_i)$ is a deterministic function query, we can interpret $\hat{\boldsymbol{C}}_{\text{SAVE}}$ as a Monte Carlo approximation to $\boldsymbol{C}_{\text{SAVE}}$. Thus, $\hat{\boldsymbol{C}}_{\text{SAVE}}$ and its eigenpairs are random—not because of any randomness in the map f but because of the random choices of $\mathbf{x}_i$. The following theorem shows the rate of mean-squared convergence of the eigenvalues of $\hat{\boldsymbol{C}}_{\text{SAVE}}$.

Theorem 7. *Assume that Algorithm 2 has been applied to the data set $\{[\,\mathbf{x}_i^\top\,,\, y_i\,]\}$, with $i = 1,\dots,N$, where the $\mathbf{x}_i$ are drawn independently according to $\pi_{\mathbf{x}}$ and $y_i = f(\mathbf{x}_i)$ are point evaluations of f. Then, for $k = 1,\dots,m$,*

$$\mathbb{E}\left[\Big(\lambda_k(\boldsymbol{C}_{\text{SAVE}}) - \lambda_k(\hat{\boldsymbol{C}}_{\text{SAVE}})\Big)^2\right] \;=\; \mathcal{O}(N_{r_{\min}}^{-1}) \tag{3.50}$$

where $\lambda_k(\cdot)$ denotes the kth eigenvalue of the given matrix.

The proof is in Appendix A.6. We note that the column space of $\hat{\boldsymbol{C}}_{\text{SAVE}}$ is not contained in f's central subspace because of finite sampling. However, using a sequence of estimates with increasing N, one may be able to distinguish effects of finite sampling from true directions of variability in f.

Next, we examine the convergence of the subspaces Algorithm 2 produces, where the subspace distance is from (3.37).

Theorem 8. *Assume the same conditions from Theorem 7. Then, for sufficiently large N,*

$$\text{dist}\,\Big(\text{ran}(\boldsymbol{A}),\, \text{ran}(\hat{\boldsymbol{A}})\Big) \;=\; \frac{1}{\lambda_n(\boldsymbol{C}_{\text{SAVE}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SAVE}})}\; \mathcal{O}_p(N_{r_{\min}}^{-1/2}), \tag{3.51}$$

where $\mathcal{O}_p$ denotes convergence in probability.

The proof is in Appendix A.7. The subspace error for Algorithm 2 decays asymptotically like $N_{r_{\min}}^{-1/2}$ with high probability. Similar to the estimated SIR subspace from Algorithm 1, the error depends inversely on the eigenvalue gap. If the gap between the nth and $(n+1)$th eigenvalues is large, then the error in the estimated n-dimensional subspace is small for a fixed number of samples.

3.3 Numerical results

We apply SIR and SAVE to four test problems to study the methods' applicability for ridge recovery and approximation and to verify our convergence analysis. The first two are quadratic functions with known ridge structure, the third is the output from a parameterized boundary value problem, and the fourth is the Hartmann problem from Section 2.3.1.

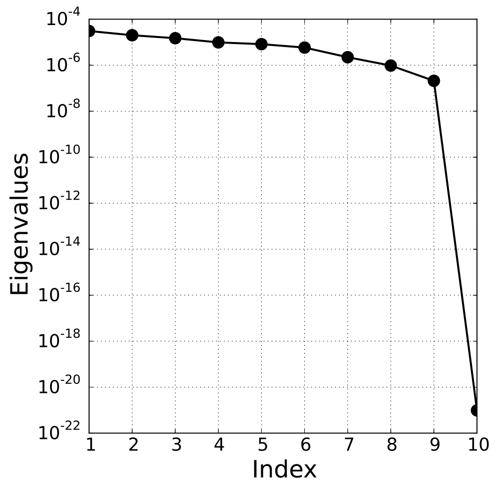
3.3.1 One-dimensional quadratic problem

We study a simple one-dimensional quadratic ridge function to contrast the recovery properties of SIR versus SAVE. Let $\pi_{\mathbf{x}}$ have a standard multivariate Gaussian density function on $\mathbb{R}^{10}$. Define

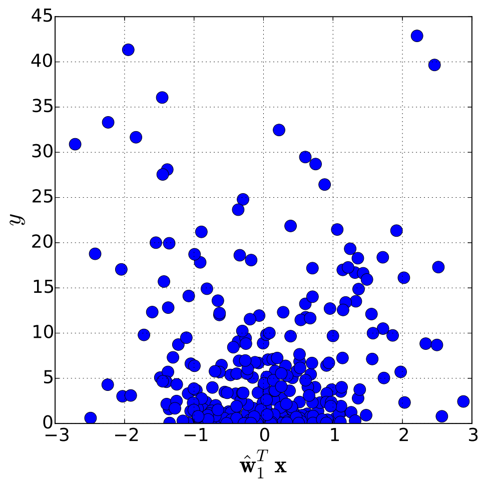
$$y \;=\; f(\mathbf{x}) \;=\; \left(\mathbf{b}^\top \mathbf{x}\right)^2, \tag{3.52}$$

where $\mathbf{b} \in \mathbb{R}^{10}$ is a constant vector. The span of $\mathbf{b}$ is the central subspace. First, we attempt to estimate the central subspace using SIR (Algorithm 1), which is known to fail for functions symmetric about $\mathbf{x} = \mathbf{0}$ [33]; Figure 3.1 confirms this failure. In fact, $\boldsymbol{C}_{\text{IR}}$ is zero since the conditional expectation of $\mathbf{x}$ for any value of y is zero. Figure 3.1a shows that all estimated eigenvalues of the SIR matrix are nearly zero as expected. Figure 3.1b is a one-dimensional shadow plot of y_i against $\hat{\mathbf{w}}_1^\top \mathbf{x}_i$, where $\hat{\mathbf{w}}_1$ denotes the normalized eigenvector associated with the largest eigenvalue of $\hat{\boldsymbol{C}}_{\text{SIR}}$ from Algorithm 1. If (i) the central subspace is one-dimensional (as in this case) and (ii) the chosen SDR algorithm correctly identifies the one basis vector, then the shadow plot will show a univariate relationship between the linear combination of input evaluations and the associated outputs. Due to the symmetry in the quadratic function, SIR fails to recover the basis vector; the shadow plot's lack of univariate relationship confirms the failure.

Figure 3.2 shows results from applying SAVE (Algorithm 2) to the quadratic function (3.52). Figure 3.2a shows the eigenvalues of $\hat{\boldsymbol{C}}_{\text{SAVE}}$ from Algorithm 2. Note the large gap between the first and second eigenvalues, which suggests that the SAVE subspace is one-dimensional. Figure 3.2b shows the shadow plot using the first eigenvector $\hat{\mathbf{w}}_1$ from Algorithm 2, which reveals the univariate



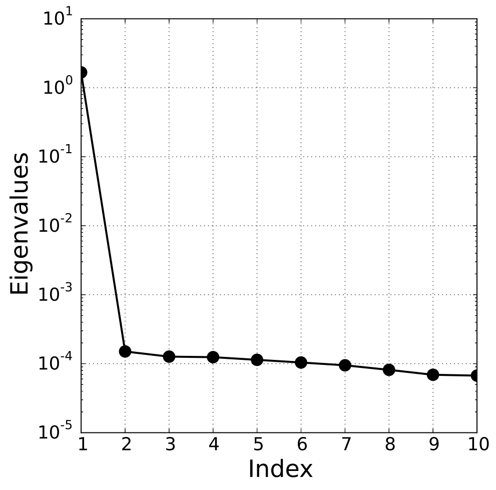
(a) Eigenvalues of $\hat{\boldsymbol{C}}_{\mathrm{SIR}}$ from (3.13)



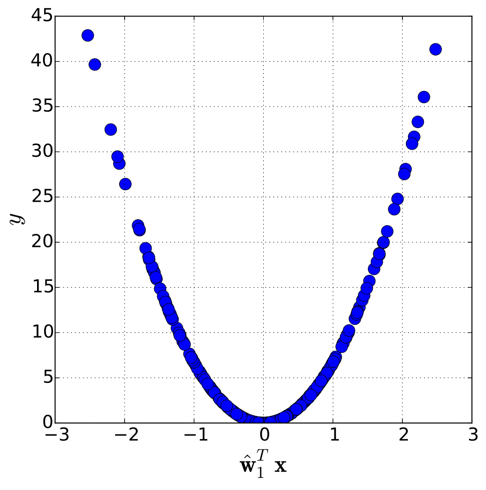
(b) Shadow plot for SIR

Figure 3.1: The results of the SIR algorithm applied to (3.52).

quadratic relationship between $\hat{\mathbf{w}}_1^\top \mathbf{x}$ and y.



(a) Eigenvalues of $\hat{\boldsymbol{C}}_{\mathrm{SAVE}}$ from (3.22)



(b) Shadow plot for SAVE

Figure 3.2: The results of the SAVE algorithm applied to (3.52).

3.3.2 Three-dimensional quadratic problem

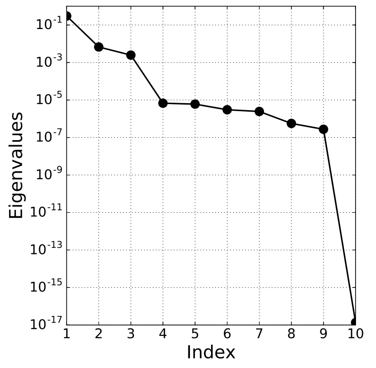
Next, we numerically study the convergence properties of the SIR and SAVE algorithms using a more complex quadratic function. Let $\pi_{\mathbf{x}}$ have a standard multivariate Gaussian density function on $\mathbb{R}^{10}$. Define

$$y = f(\mathbf{x}) = \mathbf{x}^\top \boldsymbol{B}\boldsymbol{B}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{x}, \tag{3.53}$$

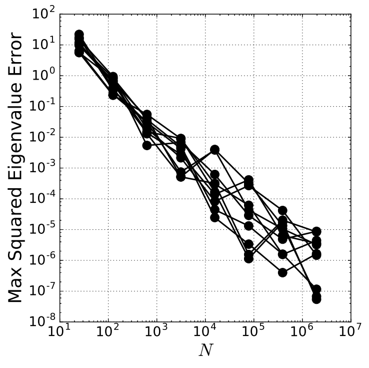
where $\boldsymbol{B} \in \mathbb{R}^{10\times 2}$ and $\mathbf{b} \in \mathbb{R}^{10}$ with $\mathbf{b} \notin \operatorname{colspan}(\boldsymbol{B})$. Figure 3.3a shows the eigenvalues of $\hat{\boldsymbol{C}}_{\text{SIR}}$; note the gap between the third and fourth eigenvalues. Figure 3.3b shows the maximum squared eigenvalue error normalized by the largest eigenvalue,

$$\max_{1\le i\le m} \frac{\left(\lambda_i(\hat{\boldsymbol{C}}_{\text{SIR}}) - \lambda_i(\boldsymbol{C}_{\text{SIR}})\right)^2}{\lambda_1(\boldsymbol{C}_{\text{SIR}})^2}, \tag{3.54}$$

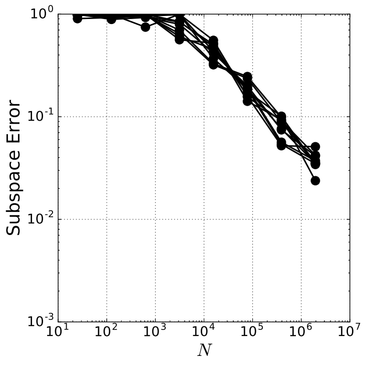
for increasing numbers of samples in 10 independent trials. We estimate the true eigenvalues using SIR with 10^7 samples. The average error decays at a rate slightly faster than the $\mathcal{O}(N^{-1})$ from Theorem 4. The improvement can likely be attributed to the adaptive slicing procedure discussed at the beginning of this section. Figure 3.3c shows the error in the estimated three-dimensional SIR subspace (see (3.37)) for increasing numbers of samples. We use 10^7 samples to estimate the true SIR subspace. The subspace errors decrease asymptotically at a rate of approximately $\mathcal{O}(N^{-1/2})$, which agrees with Theorem 5.



(a) Eigenvalues of $\hat{\boldsymbol{C}}_{\text{SIR}}$ from (3.13)



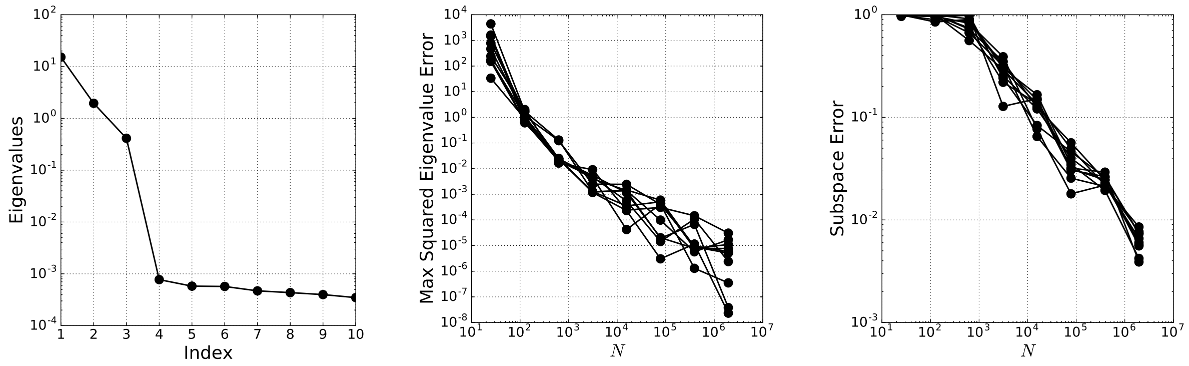
(b) Maximum squared eigenvalue error, SIR



(c) SIR subspace errors for $n = 3$

Figure 3.3: Eigenvalues, eigenvalue errors, and subspace errors for SIR applied to (3.53). The error decreases with increasing samples consistent with the convergence theory in Section 3.2.1.

Figure 3.4 shows the results of a similar convergence study using SAVE (Algorithm 2). The eigenvalues of $\hat{\boldsymbol{C}}_{\text{SAVE}}$ from (3.22) are shown in Figure 3.4a. Note the large gap between the third and fourth eigenvalues, which is consistent with the three-dimensional central subspace in f from (3.53). Figures 3.4b and 3.4c show the maximum squared eigenvalue error and the subspace error, respectively, for $n = 3$. The eigenvalue error again decays at a faster rate than expected in Theorem 7—likely due to the adaptive slicing implemented in the code. The subspace error decays consistently according to Theorem 8.



(a) Eigenvalues of $\hat{\boldsymbol{C}}_{\text{SAVE}}$ from (3.22) (b) Maximum squared eigenvalue error, SAVE (c) SAVE subspace errors for $n = 3$

Figure 3.4: Eigenvalues, eigenvalue errors, and subspace errors for SAVE applied to (3.53). The error decreases with increasing samples consistent with the convergence theory in Section 3.2.2.

3.3.3 Boundary value problem

The next problem is a standard test problem in dimension reduction [131, 46]. It is derived from the boundary value problem

$$\begin{aligned} \left(e^{\alpha(t,\mathbf{x})}\, u'(t,\mathbf{x})\right)' \;&=\; 1, \quad t \in [-1,1] \\ u(-1,\mathbf{x}) = 0,\;\; u'(1,\mathbf{x}) &= 0, \end{aligned} \tag{3.55}$$

where the parameters $\mathbf{x} \in \mathbb{R}^{10}$ characterize the boundary value problem through α. This coefficient is defined by

$$\alpha(t,\mathbf{x}) \;=\; \sum_{i=1}^{m} \lambda_i^{1/2} \psi_i(t)\, x_i, \tag{3.56}$$

where λ_i and $\psi_i(t)$ are the eigenvalue and eigenfunction of the kernel $k(s,t) = \sigma^2 e^{-\frac{|s-t|}{2\ell}}$. The parameters σ and ℓ represent the scaling and correlation length of the kernel. By tuning these parameters, we can control the difficulty in numerically approximating (3.55). For this study, we use $\sigma = 1$ and $\ell = 0.5$. We approximate $u(t, \mathbf{x})$ using an M-point numerical discretization over the domain $t \in [-1, 1]$ [19]. For this problem, $u(1, \mathbf{x})$ is the scalar-valued output of interest,

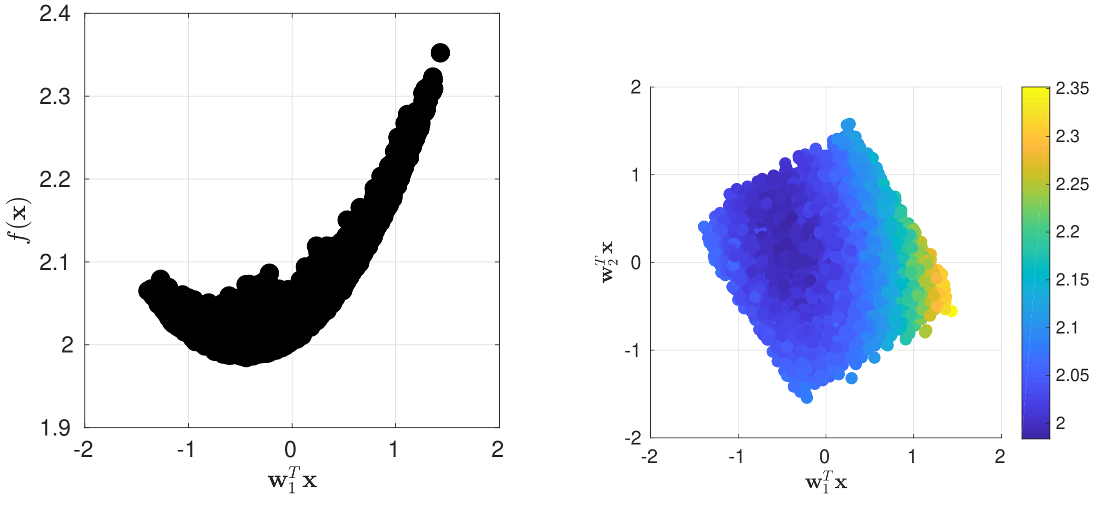
$$y \;=\; f(\mathbf{x}) \;=\; u(1, \mathbf{x}). \tag{3.57}$$

The inputs are uniformly distributed over the $[-1, 1]^{10}$ hypercube.

Figure 3.5 contains the one- and two-dimensional shadow plots for (3.57). These plots show reasonable low-dimensional structure and provide some insight into the structure of $f(\mathbf{x})$. For this problem, we are more interested in studying how refining the mesh for the numerical approximation of (3.55) affects the ridge approximation. Figure 3.6 contains plots of SIR and SAVE subspace distances for (3.57) against the numerical discretization used to solve (3.55). To perform Algorithms 1 and 2, we use the same set of 10,000 randomly sampled parameters $\mathbf{x}_i \sim \mathcal{U}([-1, 1]^{10})$ for all discretization. This ensures that the numerical approximation of (3.55) is the only factor varying the approximation of the central subspace. For both SIR and SAVE, the approximation of the central subspace is improved by refining the numerical discretization. For a sufficiently refined discretization, we expect the dominant subspaces errors to be caused by the numerical method used to uncover the ridge directions.

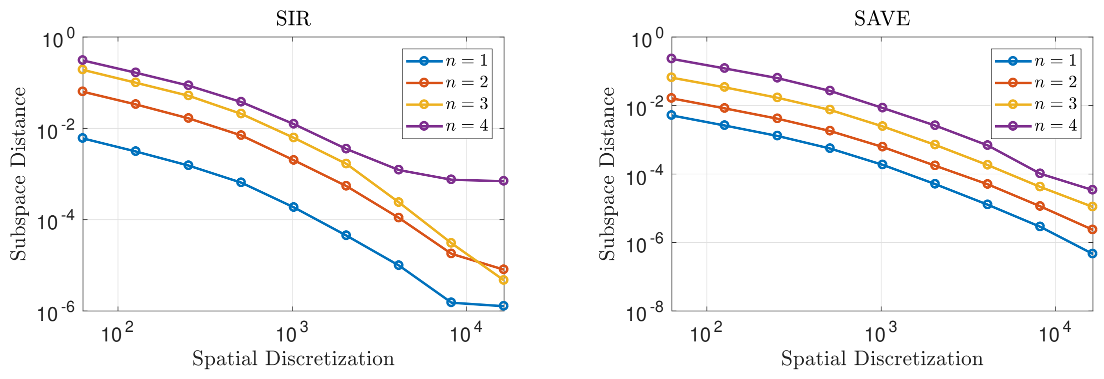
3.3.4 Hartmann problem

The following study steps in the direction of using SIR and SAVE for parameter space dimension reduction in a physics-based model. We use the Hartmann problem from Chapter 2. See Section 2.3.1 and Figure 2.1 for details on the Hartmann problem. We use the same (log-transformed) inputs and input probability measure as in Chapter 2. Recall that the Hartmann problem considered two useful outputs: (i) the induced magnetic field B_{ind} (see Equation 2.37) and (ii) the average flow velocity u_{avg} (see Equation 2.36). In this section, we study and analyze the SIR and SAVE algorithms for B_{ind}. For completeness, we then include plots showing the results of the same studies applied to u_{avg}. It should be noted that the theory in the paper requires that



(a) One-dimensional shadow plot for (3.57) (b) Two-dimensional shadow plot for (3.57)

Figure 3.5: One- and two-dimensional shadow plots of the output from the ODE in (3.57).



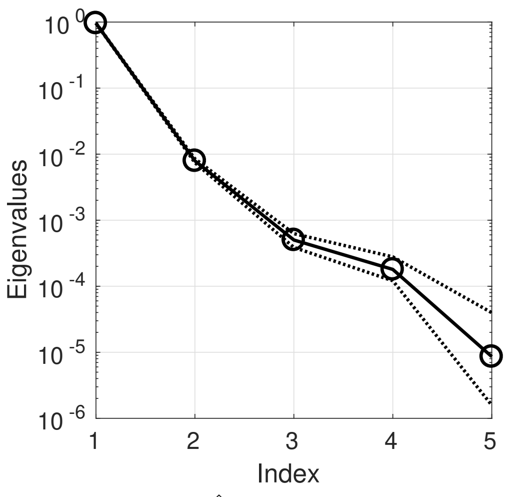
(a) Subspace errors for SIR (b) Subspace errors for SAVE

Figure 3.6: Subspace errors as a function of the discretization used to solve the ODE in (3.55).

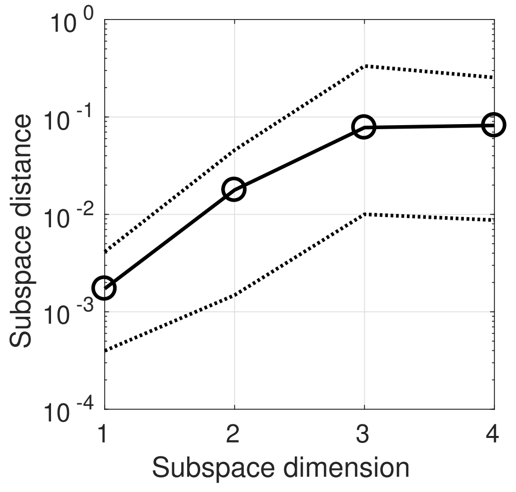
the input density be elliptically symmetric. This does not hold in this case where we are applying uniform densities; however, the results are still useful from the standpoint of evaluating SIR and SAVE as heuristic methods for ridge approximation.

Figure 3.7 shows the results of applying SIR (Algorithm 1) to the Hartmann problem for the induced magnetic field B_{ind} using $N = 10^6$ randomly drawn samples. The eigenvalues of $\hat{\boldsymbol{C}}_{\text{SIR}}$ from (3.13) with bootstrap ranges are shown in Figure 3.7a. Large gaps appear after the first and second eigenvalues, which indicates possible two-dimensional ridge structure. Figures 3.7c and 3.7d contain one- and two-dimensional shadow plots of B_{ind} against $\hat{\mathbf{w}}_1^\top\mathbf{x}$ and $\hat{\mathbf{w}}_2^\top\mathbf{x}$, where $\hat{\mathbf{w}}_1$ and $\hat{\mathbf{w}}_2$

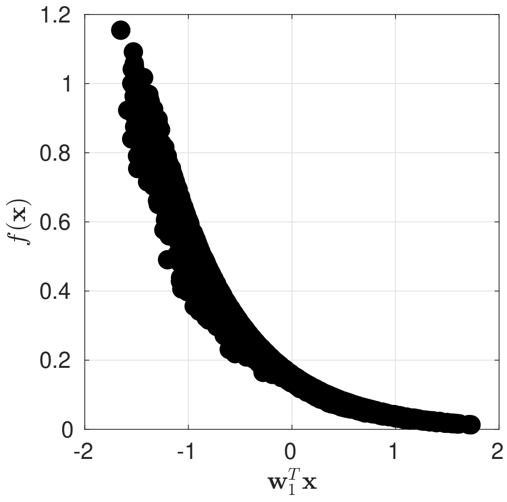
are the first two eigenvectors of $\hat{\boldsymbol{C}}_{\text{SIR}}$. We see a strong one-dimensional relationship in terms of $\hat{\mathbf{w}}_1^\top\mathbf{x}$, but there is some slight curvature with changes in $\hat{\mathbf{w}}_2^\top\mathbf{x}$. These results suggest that SIR may provide a heuristic approach to ridge approximation as well. Figure 3.7b shows the subspace errors as a function of the subspace dimension. Recall from Theorem 5 that the subspace error depends inversely on the eigenvalue gap. The largest eigenvalue gap occurs between the first and second eigenvalues, which is consistent with the smallest subspace error for $n = 1$.



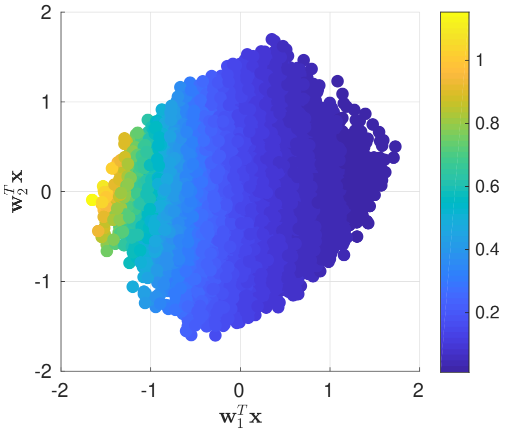
(a) Eigenvalues of $\hat{\boldsymbol{C}}_{\text{SIR}}$ with bootstrap ranges



(b) Subspace errors from $\hat{\boldsymbol{C}}_{\text{SIR}}$ with bootstrap ranges



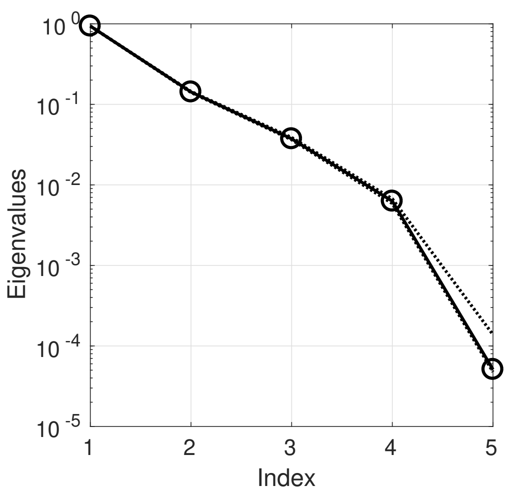
(c) One-dimensional SIR shadow plot for B_{ind}



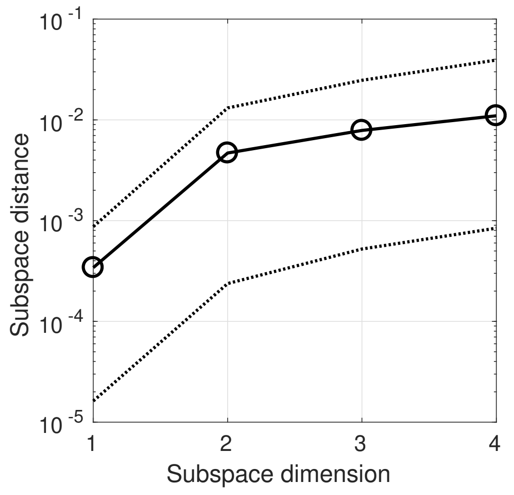
(d) Two-dimensional SIR shadow plot for B_{ind}

Figure 3.7: Eigenvalues, estimated subspace errors, and shadow plots for SIR (Algorithm 1) applied to B_{ind}.

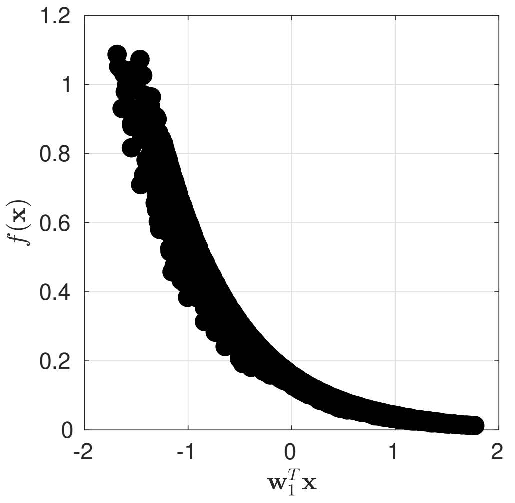
We perform the same numerical studies using SAVE (Algorithm 2). Figure 3.8a shows the eigenvalues of the $\hat{\boldsymbol{C}}_{\text{SAVE}}$ from (3.22) for the induced magnetic field B_{ind} from (2.37). Note the large gaps after the first and second eigenvalues. These gaps are consistent with the subspace errors in Figure 3.8b, where the one- and two-dimensional subspace estimates have the smallest errors. Figures 3.8c and 3.8d contain shadow plots for $\hat{\mathbf{w}}_1^\top \mathbf{x}$ and $\hat{\mathbf{w}}_2^\top \mathbf{x}$, where $\hat{\mathbf{w}}_1$ and $\hat{\mathbf{w}}_2$ are the first two eigenvectors from $\hat{\boldsymbol{C}}_{\text{SAVE}}$ in (3.22).



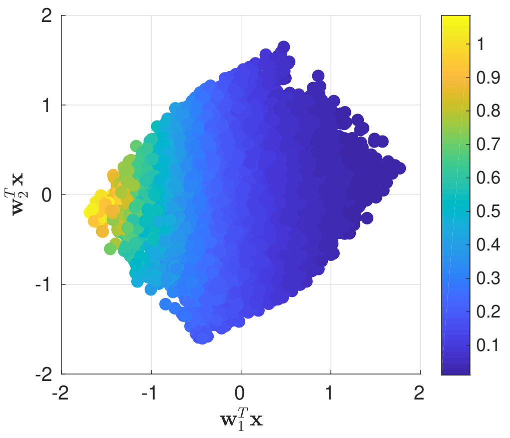
(a) Eigenvalues of $\hat{\boldsymbol{C}}_{\text{SAVE}}$ with bootstrap ranges



(b) Subspace errors from $\hat{\boldsymbol{C}}_{\text{SAVE}}$ with bootstrap ranges



(c) One-dimensional SAVE shadow plot for B_{ind}



(d) Two-dimensional SAVE shadow plot for B_{ind}

Figure 3.8: Eigenvalues, estimated subspace errors, and shadow plots for SAVE (Algorithm 2) applied to B_{ind}.

For completeness, we include the same results from Algorithms 1 and 2 applied to the average flow velocity at the end of this section. The results are similar to those from Figures 3.7 and 3.8, and the interpretations of these results are the same as discussed above.

Lastly, we study how well the SIR and SAVE algorithms perform in finding ridge structure in the Hartmann problem. We use the expected conditional variance from (1.10) normalized by the total variance of the output as a metric for the amount of variation capture by the computed ridge directions. Equation (1.10) is approximated in the same way as in the comparable study from Section 2.3.1. We draw 10,000 random samples according to $\pi_{\mathbf{x}}$ and transform these samples into n-dimensional space via $\boldsymbol{A}^\top \mathbf{x}$. Then at each of these points, we use a hit-and-run algorithm to draw 1,000 conditional samples. We compute the conditional variance of each set of 1,000 samples and then average all of the conditional variances to approximate the expected conditional variance. The smaller this quantity, the better a given function is approximated by an n-dimensional ridge function.

Tables 3.1 and 3.2 contain the results of this study for the ridge directions computed using the SIR and SAVE algorithms, respectively. In each case, we see the inverse regression algorithms have captured about 95% of the total variation in the output with just one ridge direction. We also notice that the expected conditional variances do not decay to zero as rapidly as in Section 2.3.1. This should be interpreted as error in the computed ridge directions. There are two likely contributions to this error. First, the input space is weight by a uniform density over the hyperrectangle, which violates a fundamental assumption required for SIR and SAVE to perform exact ridge recovery. However, the ability of these algorithms to capture most of the variation in B_{ind} and u_{avg} supports their use as a heuristic for ridge approximation. Second, we use a Monte Carlo approximation of the input space, compared to the high-order tensor product quadrature used in the last chapter. Due to the slice-based nature of the SIR and SAVE algorithms, only Monte Carlo approximations of the input space can be used; this is an issue we address in the next chapter.

Table 3.1: The normalized expected conditional variance from (1.10) for the ridge functions computed using the SIR algorithm.

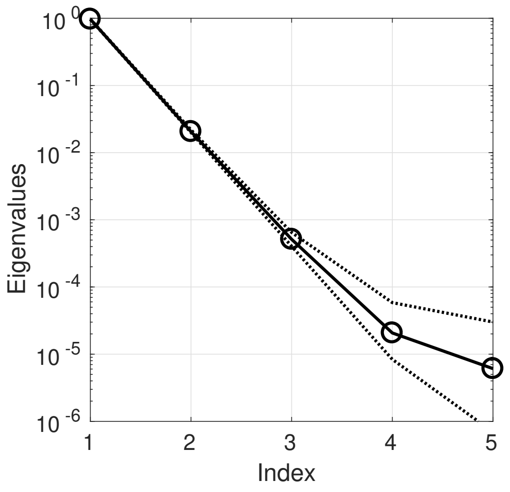
	B_{ind} Original inputs	B_{ind} Log-transformed inputs	u_{avg} Original inputs	u_{avg} Log-transformed inputs
$n = 1$	8.25×10^{-2}	3.32×10^{-2}	5.99×10^{-2}	1.31×10^{-2}
$n = 2$	4.46×10^{-2}	1.92×10^{-3}	1.23×10^{-2}	7.97×10^{-3}
$n = 3$	1.54×10^{-2}	5.21×10^{-4}	3.04×10^{-3}	8.45×10^{-5}
$n = 4$	1.22×10^{-5}	5.77×10^{-6}	4.33×10^{-8}	5.59×10^{-6}

Table 3.2: The normalized expected conditional variance from (1.10) for the ridge functions computed using the SAVE algorithm.

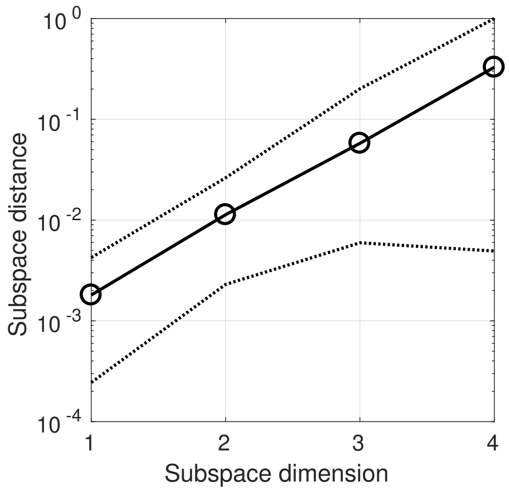
	B_{ind} Original inputs	B_{ind} Log-transformed inputs	u_{avg} Original inputs	u_{avg} Log-transformed inputs
$n = 1$	9.15×10^{-2}	3.35×10^{-2}	5.23×10^{-2}	1.46×10^{-2}
$n = 2$	5.17×10^{-2}	2.12×10^{-3}	1.33×10^{-2}	$8.05.23 \times 10^{-3}$
$n = 3$	1.42×10^{-2}	2.39×10^{-4}	1.38×10^{-3}	2.13×10^{-4}
$n = 4$	3.02×10^{-7}	5.68×10^{-9}	4.23×10^{-7}	4.29×10^{-7}

3.4 Summary

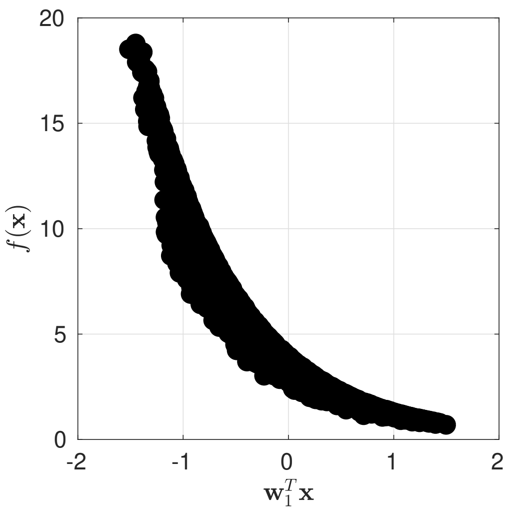
Seeking data-driven machine learning methods for computational science models, we investigate sufficient dimension reduction from statistical regression as a tool for subspace-based input space dimension reduction in deterministic functions. We show that SDR is theoretically justified as a tool for ridge recovery by proving equivalence of the dimension reduction subspace and the ridge subspace for some deterministic $y = f(\mathbf{x})$. We interpret two SDR algorithms for the ridge recovery problem: sliced inverse regression and sliced average variance estimation. In regression, these methods use moments of the inverse regression $\mathbf{x}|y$ to estimate subspaces relating to the central subspace. In ridge recovery, we reinterpret SIR and SAVE as numerical integration methods for estimating inverse conditional moment matrices, where the integrals are over contour sets of f. We show that the column spaces of the conditional moment matrices are contained in the ridge subspace, which justifies their eigenspaces as tools ridge recovery.



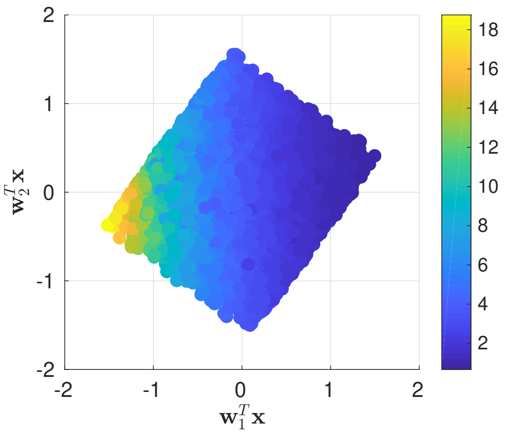
(a) Eigenvalues of $\hat{\boldsymbol{C}}_{\mathrm{SIR}}$ with bootstrap ranges



(b) Subspace errors from $\hat{\boldsymbol{C}}_{\mathrm{SIR}}$ with bootstrap ranges

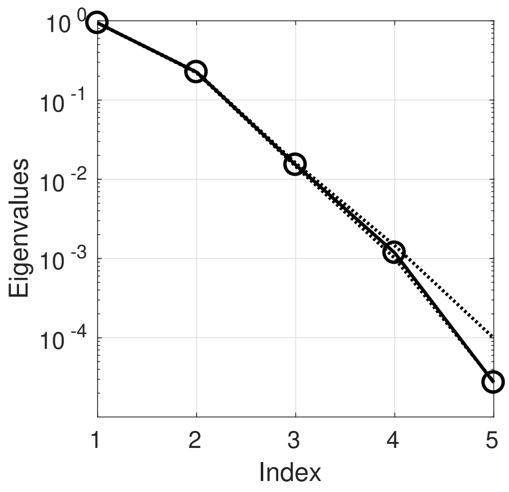


(c) One-dimensional SIR shadow plot for u_{avg}

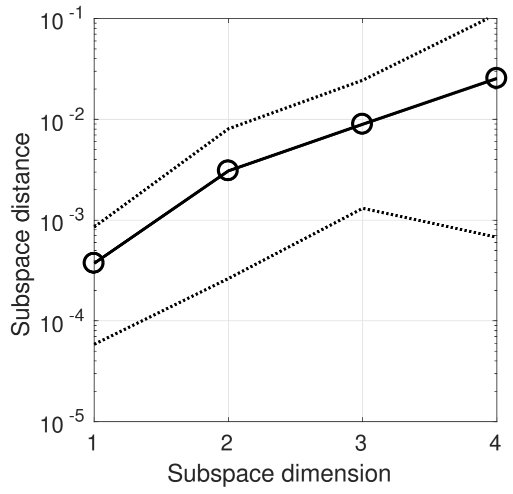


(d) Two-dimensional SIR shadow plot for u_{avg}

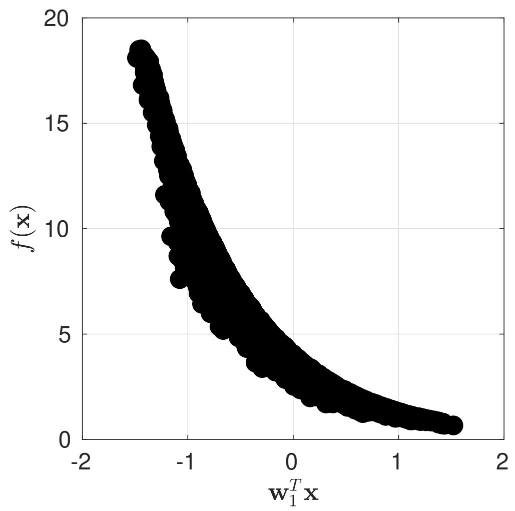
Figure 3.9: Eigenvalues, estimated subspace errors, and shadow plots for SIR (Algorithm 1) applied to u_{avg}.



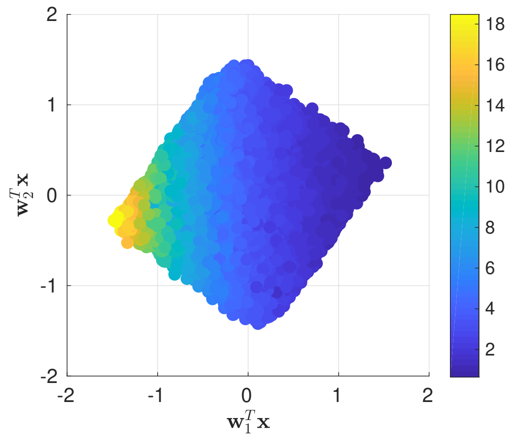
(a) Eigenvalues of $\hat{\boldsymbol{C}}_{\text{SAVE}}$ with bootstrap ranges



(b) Subspace errors from $\hat{\boldsymbol{C}}_{\text{SAVE}}$ with bootstrap ranges



(c) One-dimensional SAVE shadow plot for u_{avg}



(d) Two-dimensional SAVE shadow plot for u_{avg}

Figure 3.10: Eigenvalues, estimated subspace errors, and shadow plots for SAVE (Algorithm 2) applied to u_{avg}.

Chapter 4

Gauss-Christoffel quadrature for inverse regression

In the previous chapter, we explore the connection between sufficient dimension reduction (SDR) and ridge functions. In particular, we showed how the inverse regression algorithms sliced inverse regression (SIR) [80] and sliced average variance estimation (SAVE) [33] address the ridge recovery problem by computing a basis for the central subspace using statistical characteristics of $f(\mathbf{x})$. SIR and SAVE each approximate a specific matrix of expectations by slicing the output space based on a given dataset. In the context of ridge recovery, these expectations become Lebesgue integrals over the function's range—i.e., the space of simulation outputs. Assuming the output distributions are sufficiently smooth, the integrals can be written as Riemann integrals against the push-forward density induced by the function and the distribution on the input space. Then the slicing from SIR and SAVE can be interpreted as a Riemann sum approximation of these integrals [39]. The approximation accuracy of SIR and SAVE depends on the number of terms in the Riemann sum—i.e., the number of slices. The Riemann sum approximation acts as an accuracy bottleneck for these algorithms; obtaining additional data or using a better design on the input space cannot maximally improve the approximation. In this chapter, we introduce new algorithms—Lanczos-Stieltjes inverse regression (LSIR) and Lanczos-Stieltjes average variance estimation (LSAVE)—that improve the accuracy and convergence rates compared to slicing by employing high-order Gauss-Christoffel quadrature to approximate the integrals with respect to the output. The new algorithms eliminate the bottleneck due to Riemann sums and place the burden of accuracy on the input space design. Thus, LSIR and LSAVE enable the use of higher-order numerical integration rules on the input space such as sparse grids or tensor product quadrature when such approaches are appropriate.

This work contains two main contributions. First, by characterizing the matrices of expectations from SIR and SAVE as integrals in the context of dimension reduction for deterministic functions, we interpret the slice-based methods as Riemann sum approximations of these integral. We recognize that this approach produces an accuracy bottleneck on the approximation of the matrices of expectations underlying SIR and SAVE. Second, we develop high-order quadrature methods for these integrals that remove this accuracy bottleneck imposed by the Riemann sums on the output space. When the input space dimension is sufficiently small and the function of interest is sufficiently smooth so that high-order numerical integration is justified on the input space, our new LSIR and LSAVE methods produce exponentially converging estimates of the matrices of expectations used for subspace-based dimension reduction—compared to the maximal first-order algebraic convergence rate resulting from the slicing. When low-order, dimension-independent numerical integration methods—e.g., simple Monte Carlo—are more appropriate due to a high input space dimension, the LSIR and LSAVE methods perform as well as the best slice-based approaches—e.g., adaptive partitioning of the output space.

4.1 SIR & SAVE for deterministic functions

We briefly revisit the inverse regression methods SIR and SAVE introduced in Chapter 3 before exploring the numerical techniques we employ in the proposed algorithms. Recall from the (1.1) that we represent a given computational model by a deterministic function that maps m simulation inputs to a scalar-valued output,

$$y \;=\; f(\mathbf{x}), \qquad \mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^m, \quad y \in \mathcal{F} \subseteq \mathbb{R}, \tag{4.1}$$

as is the typical setup in the field of computer experiments [100, 76, 102]. Note that for this work we specify the space of inputs and outputs by $\mathcal{X}$ and $\mathcal{F}$, respectively. We assume that (4.1) is accompanied by a known input probability measure $\pi_{\mathbf{x}}$. This probability measure, when propagated forward through $f(\mathbf{x})$, induces a probability measure over $\mathcal{F}$, which we denote by π_y. This measure is fully determined by $\pi_{\mathbf{x}}$ and f, but its form is assumed to be unknown.

Recall from Sections 3.1.1 and 3.1.2 that the SIR and SAVE algorithms use N random samples $\{[\,\mathbf{x}_i^\top\,,\, y_i\,]\}$ (where $y_i = f(\mathbf{x}_i)$) to approximate the underlying population matrices

$$\begin{aligned} \boldsymbol{C}_{\mathrm{IR}} &= \mathrm{Cov}\left[\mathbb{E}\left[\mathbf{x}|y\right]\right], \\ \boldsymbol{C}_{\mathrm{AVE}} &= \mathbb{E}\left[\left(\boldsymbol{I} - \mathrm{Cov}\left[\mathbf{x}|y\right]\right)^2\right], \end{aligned} \tag{4.2}$$

respectively. In the context of ridge recovery, we reformulate these matrices as integrals against probability measures,

$$\begin{aligned} \boldsymbol{C}_{\mathrm{IR}} &= \int \boldsymbol{\mu}(y)\,\boldsymbol{\mu}(y)^\top\,\mathrm{d}\pi_y(y), \\ \boldsymbol{C}_{\mathrm{AVE}} &= \int \left(\boldsymbol{I} - \boldsymbol{\Sigma}(y)\right)^2\,\mathrm{d}\pi_y(y), \end{aligned} \tag{4.3}$$

where the conditional expectation and conditional covariance are

$$\begin{aligned} \boldsymbol{\mu}(y) &= \int \mathbf{x}\,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}), \\ \boldsymbol{\Sigma}(y) &= \int \left(\mathbf{x} - \boldsymbol{\mu}(y)\right)\,\left(\mathbf{x} - \boldsymbol{\mu}(y)\right)^\top\,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}), \end{aligned} \tag{4.4}$$

respectively.

The integrals in (4.3) and (4.4) are difficult to approximate from the random samples since the induced measure π_y and the conditional $\pi_{\mathbf{x}|y}$ are unknown and potentially complex. Algorithms 1 and 2 slice the range of output values to enable this approximation. Let $h(y)$ denote the slicing function from (3.9) and define the probability mass function

$$\omega(r) = \int_{J_r} \mathrm{d}\pi_y(y), \qquad r = 0, \dots, R-1, \tag{4.5}$$

where J_r is the rth interval over the range of outputs. We can then express the slice-based matrices $\boldsymbol{C}_{\mathrm{SIR}}$ and $\boldsymbol{C}_{\mathrm{SAVE}}$ as sums over the R slices,

$$\begin{aligned} \boldsymbol{C}_{\mathrm{SIR}} &= \sum_{r=0}^{R-1} \omega(r)\,\boldsymbol{\mu}(r)\,\boldsymbol{\mu}(r)^\top, \\ \boldsymbol{C}_{\mathrm{SAVE}} &= \sum_{r=0}^{R-1} \omega(r)\,\left(\boldsymbol{I} - \boldsymbol{\Sigma}(r)\right)^2, \end{aligned} \tag{4.6}$$

where $\boldsymbol{\mu}(r)$ and $\boldsymbol{\Sigma}(r)$ are as in (4.4) but applied to the sliced output $h(y) = r$.

By expressing the various elements of SIR and SAVE as integrals, we emphasize the two levels of approximation occurring in these algorithms: (i) approximation in terms of the number of samples N and (ii) approximation in terms of the number of slices R. That is,

$$\begin{aligned} \hat{\boldsymbol{C}}_{\text{SIR}} &\approx \boldsymbol{C}_{\text{SIR}} \approx \boldsymbol{C}_{\text{IR}}, \\ \hat{\boldsymbol{C}}_{\text{SAVE}} &\approx \boldsymbol{C}_{\text{SAVE}} \approx \boldsymbol{C}_{\text{AVE}}. \end{aligned} \tag{4.7}$$

Theorems 5 and 8 from Chapter 3 show that the approximation due to sampling—i.e., the leftmost approximations in (4.7)—converge like $\mathcal{O}(N_{r_{\min}}^{-1/2})$. From the integral perspective, the slicing approach can be interpreted as a Riemann sum approximation of the integrals in (4.3). Riemann sum approximations estimate integrals by a finite sum. These approximations converge like R^{-1} for continuous functions on compact domains, where R denotes the number of terms in the Riemann sum [39, Chap. 2].

In the next section, we introduce several numerical tools and link them to the various elements of the SIR and SAVE algorithms discussed so far. We use these tools, including orthogonal polynomials and numerical quadrature, in the proposed algorithms to enable approximation of $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$ without using Riemann sums.

4.2 Orthogonal polynomials and Gauss-Christoffel quadrature

Orthogonal polynomials and numerical quadrature are fundamental to numerical analysis and have been studied extensively; detailed discussions are available in [83, 54, 87, 63]. Our discussion is based on these references, reviewing key concepts necessary to develop the new algorithms for approximating the matrices in (4.3). We begin with the Stieltjes procedure for constructing orthonormal polynomials with respect to a given measure. We then relate this procedure to Gauss-Christoffel quadrature, polynomial expansions, and the Lanczos algorithm.

4.2.1 The Stieltjes procedure

The Stieltjes procedure recursively constructs a sequence of polynomials that are orthonormal with respect to a given measure [83, Chap. 3]. Let π denote a given probability measure over $\mathbb{R}$, and let $\phi, \psi : \mathbb{R} \to \mathbb{R}$ be two scalar-valued functions that are square-integrable with respect to π. The continuous inner product relative to π is

$$(\phi, \psi)_\pi \;=\; \int \phi(y)\,\psi(y)\,\mathrm{d}\pi(y), \tag{4.8}$$

and the induced norm is $||\phi||_\pi = \sqrt{(\phi, \phi)_\pi}$. Consider a sequence of polynomials $\{\phi_0, \phi_1, \phi_2, \dots\}$ where each ϕ_i has degree i. This sequence is orthonormal with respect to π if

$$(\phi_i, \phi_j)_\pi \;=\; \delta_{i,j}, \quad i, j = 0, 1, 2, \dots, \tag{4.9}$$

where $\delta_{i,j}$ is the Kronecker delta. Algorithm 3 contains a method for constructing such a sequence of orthonormal polynomials with respect to π. This algorithm is known as the Stieltjes procedure and was first introduced in [115].

Algorithm 3 Stieltjes procedure [54, Section 2.2.3.1]

Given: The probability measure π.
Assumptions: Let $\phi_{-1}(y) = 0$ and $\tilde{\phi}_0(y) = 1$.

```
(1) For $i \;=\; 0, 1, 2, \dots$
    (i) $\beta_i \;=\; ||\tilde{\phi}_i(y)||_\pi$
    (ii) $\phi_i(y) \;=\; \tilde{\phi}_i(y) \,/\, \beta_i$
    (iii) $\alpha_i \;=\; (y\,\phi_i(y),\, \phi_i(y))_\pi$
    (iv) $\tilde{\phi}_{i+1}(y) \;=\; (y - \alpha_i)\,\phi_i(y) - \beta_i\,\phi_{i-1}(y)$
```

Output: The orthonormal polynomials $\{\phi_0, \phi_1, \phi_2, \dots\}$ and recurrence coefficients α_i, β_i for $i = 0, 1, 2, \dots$.

The last step of Algorithm 3 defines the three-term recurrence relationship for orthonormal polynomials,

$$\beta_{i+1}\phi_{i+1}(y) \;=\; (y - \alpha_i)\,\phi_i(y) - \beta_i\phi_{i-1}(y), \tag{4.10}$$

for $i = 0, 1, 2, \dots$. Any sequence of polynomials that satisfies (4.10) is orthonormal with respect to the given measure. If we consider the first k terms, then we can rearrange it to obtain

$$y\,\phi_i(y) \;=\; \beta_i\phi_{i-1}(y) + \alpha_i\phi_i(y) + \beta_{i+1}\phi_{i+1}(y), \tag{4.11}$$

for $i = 0, 1, 2, \dots, k-1$. Let

$$\boldsymbol{\phi}(y) = [\,\phi_0(y)\,,\, \phi_1(y)\,,\, \dots\,,\, \phi_{k-1}(y)\,]^\top. \tag{4.12}$$

We can then write (4.11) in matrix form as

$$y\,\boldsymbol{\phi}(y) \;=\; \boldsymbol{J}\,\boldsymbol{\phi}(y) + \beta_k\,\phi_k(y)\,\mathbf{e}_k, \tag{4.13}$$

where $\mathbf{e}_k \in \mathbb{R}^k$ is the vector of zeros with a one in the kth entry and $\boldsymbol{J} \in \mathbb{R}^{k\times k}$ is

$$\boldsymbol{J} \;=\; \begin{bmatrix} \alpha_0 & \beta_1 & & & \\ \beta_1 & \alpha_1 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{k-2} & \alpha_{k-2} & \beta_{k-1} \\ & & & \beta_{k-1} & \alpha_{k-1} \end{bmatrix}, \tag{4.14}$$

where α_i, β_i are the recurrence coefficients from Algorithm 3. This matrix—known as the Jacobi matrix—is symmetric and tridiagonal. Let the eigendecomposition of $\boldsymbol{J}$ be

$$\boldsymbol{J} \;=\; \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^{\top}. \tag{4.15}$$

From (4.13), the eigenvalues of $\boldsymbol{J}$, denoted by λ_i, $i = 0, \ldots, k-1$, are the zeros of the degree-k polynomial $\phi_k(y)$. Furthermore, the eigenvector associated with λ_i is $\boldsymbol{\phi}(\lambda_i)$. We assume the eigenvectors of $\boldsymbol{J}$ are normalized such that $\boldsymbol{Q}$ is an orthogonal matrix with entries

$$(\boldsymbol{Q})_{i+1,j+1} \;=\; \frac{\phi_i(\lambda_j)}{||\boldsymbol{\phi}(\lambda_j)||_2}, \qquad i, j = 0, \ldots, k-1, \tag{4.16}$$

where $||\cdot||_2$ is the vector 2-norm.

We end this section with brief a note about Fourier expansion of functions in terms of orthonormal polynomials. If a given function $g(y)$ is square integrable with respect to π, then g admits a mean-squared convergent Fourier series in terms of the orthonormal polynomials,

$$g(y) \;=\; \sum_{i=0}^{\infty} g_i\,\phi_i(y), \tag{4.17}$$

for y in the support of π and where equality is in the $L_2(\pi)$ sense. By orthogonality of the polynomials, the Fourier coefficients g_i are

$$g_i \;=\; (g, \phi_i)_\pi\,. \tag{4.18}$$

This polynomial approximation plays an important role in the algorithms introduced in Section 4.3.

4.2.2 Gauss-Christoffel quadrature

We next discuss the Gauss-Christoffel quadrature for numerical integration and show its connection to orthonormal polynomials [83, Chap. 3]. Given a measure π and an integrable function $g(y)$, a k-point quadrature rule approximates the integral of g with respect to π by a weighted sum of g evaluated at k input values,

$$\int g(y)\,\mathrm{d}\pi(y) \;=\; \sum_{i=0}^{k-1} \omega_i\, g(\lambda_i) + r_k. \tag{4.19}$$

The λ_i's are the quadrature nodes and the ω_i's are the associated quadrature weights. The k-point quadrature approximation error is contained in the residual term r_k. We can minimize $|r_k|$ by choosing the quadrature nodes and weights appropriately. The nodes and weights of the Gauss-Christoffel quadrature maximize the polynomial degree of exactness, which refers to the highest degree polynomial that the quadrature rule exactly integrates—i.e., $r_k = 0$. The k-point Gauss-Christoffel quadrature rule has polynomial degree of exactness $2k-1$. Furthermore, the Gauss-Christoffel quadrature has been shown to converge exponentially at a rate ρ^{-k} for integrals defined on a compact domain when the integrand is analytic [120, Chap. 19]. The base $\rho > 1$ relates to the size of the function's domain of analytical continuability. For functions with $p-1$ continuous derivatives on a compact domain, the Gauss-Christoffel quadrature converges like $k^{-(2p+1)}$.

The Gauss-Christoffel quadrature nodes and weights depend on the given measure π. They can be obtained through the eigendecomposition of $\boldsymbol{J}$ [65]. Recall from (4.14) that $\boldsymbol{J}$ is the matrix of recurrence coefficients resulting from k steps of the Stieltjes procedure. The eigenvalues of $\boldsymbol{J}$ are the zeros of the k-degree orthonormal polynomial $\phi_k(y)$. These zeros are the nodes of the k-point Gauss-Christoffel quadrature rule with respect to π—i.e., $\phi_k(\lambda_i) = 0$ for $i = 0, \ldots, k-1$. The associated weights are the squares of the first entry of each normalized eigenvector,

$$\omega_i \;=\; (\boldsymbol{Q})^2_{1,i+1} \;=\; \frac{1}{||\boldsymbol{\phi}(\lambda_i)||_2^2}, \quad i = 0, \ldots, k-1. \tag{4.20}$$

The Stieltjes procedure employs the inner product from (4.8) to define the recurrence coefficients α_i, β_i. In the next section, we consider the Lanczos algorithm and explore conditions under which it be may considered a discrete analog to the Stieltjes procedure in Section 4.2.1. Before we make this connection, we define the discrete inner product. Let λ_i, ω_i define an N-point numerical integration rule with respect to π. For example, these could be the Gauss-Christoffel quadrature nodes and weights, though they need not be. For example, we could use Monte Carlo as a randomized numerical integration method, where the λ_i's drawn randomly according to π and $\omega_i = 1/N$ for all i. The discrete inner product is the numerical approximation of the continuous inner product,

$$(\phi, \psi)_{\pi^{(N)}} \;=\; \sum_{i=0}^{N-1} \omega_i \, \phi(\lambda_i) \, \psi(\lambda_i) \;\approx\; (\phi, \psi)_{\pi}\,, \tag{4.21}$$

where the "N" in $\pi^{(N)}$ denotes the number of points in the numerical integration rule used. The discrete norm is $||\phi||_{\pi^{(N)}} = \sqrt{(\phi, \phi)_{\pi^{(N)}}}$.

The pseudospectral expansion approximates the Fourier expansion from (4.17) by truncating the series after k terms and approximating the Fourier coefficients in (4.18) using the discrete inner product [25]. We write this series for a given square-integrable function $g(y)$ as

$$g(y) \;\approx\; \hat{g}(y) \;=\; \sum_{i=0}^{k-1} \hat{g}_i \, \phi_i(y), \tag{4.22}$$

where the pseudospectral coefficients are

$$\hat{g}_i \;=\; (g, \phi_i)_{\pi^{(N)}}\,. \tag{4.23}$$

Note that the approximation of $g(y)$ by $\hat{g}(y)$ depends on two factors: (i) the approximation accuracy of the first k pseudospectral coefficients and (ii) the magnitude of the trailing pseudospectral coefficients omitted due to truncation. We can improve (i) by using a higher-order integration rule or by increasing N. We improve (ii) by including more terms in the truncated series—that is, increasing k. The pseudospectral approximation and this two-level convergence play an important role in the new algorithms proposed in Section 4.3.

4.2.3 The Lanczos algorithm

The Lanczos algorithm was originally introduced as an iterative scheme for approximating eigenvalues and eigenvectors of linear differential operators [77]. Given a symmetric $N \times N$ matrix

$\boldsymbol{A}$, it constructs a symmetric, tridiagonal $k \times k$ matrix $\boldsymbol{T}$ whose eigenvalues approximate those of $\boldsymbol{A}$. Additionally, it produces an $N \times k$ matrix, denoted by $\boldsymbol{V} = [\,\mathbf{v}_0\,,\,\mathbf{v}_1\,,\ldots,\,\mathbf{v}_{k-1}\,]$ where the $\mathbf{v}_i$'s are the Lanczos vectors. These vectors transform the eigenvectors of $\boldsymbol{T}$ into approximate eigenvectors of $\boldsymbol{A}$. Algorithm 4 contains the steps of the Lanczos algorithm. Note that the inner products and norms in Algorithm 4 are the given by

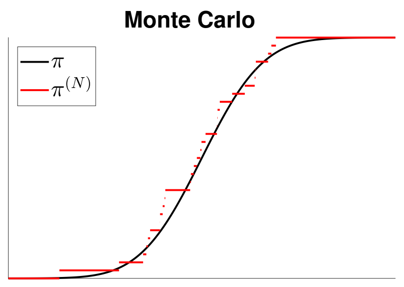
$$(\mathbf{w}, \mathbf{u}) \;=\; \mathbf{w}^\top \mathbf{u}, \qquad ||\mathbf{w}|| \;=\; \sqrt{(\mathbf{w}, \mathbf{w})}, \tag{4.24}$$

for vectors $\mathbf{w}, \mathbf{u} \in \mathbb{R}^N$. These relate to the discrete inner products introduced in Section 4.2.2, and we make precise connections later in this section.

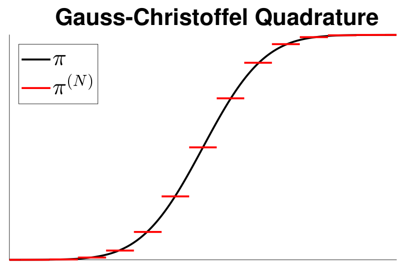
After k iterations, the Lanczos algorithm yields the relationship

$$\boldsymbol{AV} \;=\; \boldsymbol{VT} + \beta_k \mathbf{v}_k \mathbf{e}_k^\top, \tag{4.27}$$

where $\mathbf{e}_k \in \mathbb{R}^k$ is the vector of zeros with a one in the kth entry and $\boldsymbol{V}$ and $\boldsymbol{T}$ are as in (4.25) and (4.26), respectively. The matrix $\boldsymbol{T}$, similar to $\boldsymbol{J}$ in (4.14), is the Jacobi matrix [54]. The relationship between the matrices $\boldsymbol{J}$ and $\boldsymbol{T}$ has been studied extensively [53, 50, 40].



(a)



(b)

Figure 4.1: The distribution functions associated with the probability measure π and the discrete approximation $\pi^{(N)}$. Figure 4.1a constructs $\pi^{(N)}$ using a Monte Carlo integration rule with respect to π while Figure 4.1b uses Gauss-Christoffel quadrature to construct $\pi^{(N)}$.

We are interested in the use of the Lanczos algorithm as a discrete approximation to the Stieltjes procedure. Recall that Algorithm 3 (the Stieltjes procedure) assumes a given measure π. Let λ_i, ω_i be the nodes and weights for some N-point numerical integration rule with respect to π.

Algorithm 4 Lanczos algorithm [54, Section 3.1.7.1]

Given: An $N \times N$ symmetric matrix $\boldsymbol{A}$.

Assumptions: Let $\mathbf{v}_{-1} = \mathbf{0} \in \mathbb{R}^N$ and $\tilde{\mathbf{v}}_0$ be an arbitrary nonzero vector of length N.

```
(1) For $i = 0, 1 \ldots, k-1$,
      (i) $\beta_i = ||\tilde{\mathbf{v}}_i||$
      (ii) $\mathbf{v}_i = \tilde{\mathbf{v}}_i \,/\, \beta_i$
      (iii) $\alpha_i = (\boldsymbol{A}\mathbf{v}_i, \mathbf{v}_i)$
      (iv) $\tilde{\mathbf{v}}_{i+1} = (\boldsymbol{A} - \alpha_i \boldsymbol{I})\mathbf{v}_i - \beta_{i-1}\mathbf{v}_{i-1}$
(2) Define
```

$$\boldsymbol{V} = \begin{bmatrix} | & | & & | \\ \mathbf{v}_0 & \mathbf{v}_1 & \ldots & \mathbf{v}_{k-1} \\ | & | & & | \end{bmatrix} \tag{4.25}$$

and

$$\boldsymbol{T} = \begin{bmatrix} \alpha_0 & \beta_1 & & & \\ \beta_1 & \alpha_1 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{k-2} & \alpha_{k-2} & \beta_{k-1} \\ & & & \beta_{k-1} & \alpha_{k-1} \end{bmatrix}. \tag{4.26}$$

Output: The matrix of Lanczos vectors $\boldsymbol{V}$ and the matrix of recurrence coefficients $\boldsymbol{T}$.

This integration rule defines a discrete approximation of π, which we denote by $\pi^{(N)}$ (see Figure 4.1). Inner products with respect to $\pi^{(N)}$ take the form of the discrete inner product in (4.21). If we perform the Lanczos algorithm with

$$\boldsymbol{A} \;=\; \begin{bmatrix} \lambda_0 & & \\ & \ddots & \\ & & \lambda_{N-1} \end{bmatrix}, \qquad \tilde{\mathbf{v}}_0 \;=\; \begin{bmatrix} \sqrt{\omega_0} \\ \vdots \\ \sqrt{\omega_{N-1}} \end{bmatrix}, \tag{4.28}$$

then the result is equivalent to running the Stieltjes procedure using the discrete inner product with respect to $\pi^{(N)}$. Increasing N improves the approximation of $\pi^{(N)}$ to π. It can be shown that the recurrence coefficients in the resulting Jacobi matrix will converge to the recurrence coefficients related to the Stieltjes procedure with respect to π as N increases [54, Section 2.2]. In the next section, we show how this relationship between Stieltjes and Lanczos can be used to approximate composite functions, which is essential to understand the underpinnings of LSIR and LSAVE in Section 4.3.

4.2.4 Composite function approximation

The connection between Algorithms 3 and 4 can be exploited for polynomial approximation of composite functions [26]. Consider a function of the form

$$h(x) \;=\; g(f(x)), \qquad x \in \mathcal{X} \subseteq \mathbb{R} \tag{4.29}$$

where

$$\begin{aligned} f &: \mathcal{X} \to \mathcal{F} \subseteq \mathbb{R}, \\ g &: \mathcal{F} \to \mathcal{G} \subseteq \mathbb{R}. \end{aligned} \tag{4.30}$$

Assume the input space $\mathcal{X}$ is weighted with a given probability measure π_x. This measure and f induce a measure on $\mathcal{F}$ that we denote by π_y. Note that the methodology described in this section can be extended to multivariate inputs—i.e., $\mathcal{X} \subseteq \mathbb{R}^m$—through tensor product constructions and to multivariate outputs—i.e., $\mathcal{G} \subseteq \mathbb{R}^n$—by considering each output individually. We need both of these extensions in Section 4.3; however, we consider the scalar case here for clarity.

The goal is to construct a pseudospectral expansion of g using orthonormal polynomials with respect to π_y and a Gauss-Christoffel quadrature rule defined over $\mathcal{F}$. In Section 4.2.1, we examined the Stieltjes procedure which constructs a sequence of orthonormal polynomials with respect to a measure—π_y in this context. In Section 4.2.2, we showed this algorithm also produces the nodes and weights of the Gauss-Christoffel quadrature rule with respect to π_y. In Section 4.2.3, we saw how the Lanczos algorithm can be used to produce similar results for a discrete approximation to the measure π_y. All of this suggests a methodology for constructing a pseudospectral approximation of g. However, we constructed the discrete approximation of π_y in Section 4.2.3 using a numerical integration rule. We cannot do this here since π_y is unknown. We can construct an N-point numerical integration rule on $\mathcal{X}$ since π_x is known. Let x_i, ν_i denote the nodes and weights for our integration rule of choice with respect to π_x. This rule defines a discrete approximation of π_x, which we write as $\pi_x^{(N)}$. We approximate π_y by the discrete measure $\pi_y^{(N)}$ by evaluating $f_i = f(x_i)$. We then perform the Lanczos algorithm on

$$\boldsymbol{A} \;=\; \begin{bmatrix} f_0 & & \\ & \ddots & \\ & & f_{N-1} \end{bmatrix}, \qquad \tilde{\mathbf{v}}_0 \;=\; \begin{bmatrix} \sqrt{\nu_0} \\ \sqrt{\nu_1} \\ \vdots \\ \sqrt{\nu_{N-1}} \end{bmatrix} \tag{4.31}$$

to obtain the system $\boldsymbol{AV} = \boldsymbol{VT} + \beta_k \mathbf{v}_k \mathbf{e}_k^\top$.

Let the eigendecomposition of the resulting Jacobi matrix be

$$\boldsymbol{T} \;=\; \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top. \tag{4.32}$$

The eigenvalues of $\boldsymbol{T}$ define the k-point Gauss-Christoffel quadrature nodes relative to $\pi_y^{(N)}$. We denote these quadrature nodes by $\lambda_i^{(N)}$, $i = 0, \ldots, k-1$, where the superscript indicates that these nodes are relative to the discrete measure $\pi_y^{(N)}$. From (4.16), the normalized eigenvectors of $\boldsymbol{T}$ have the form

$$(\boldsymbol{Q})_{i+1} \;=\; \frac{\boldsymbol{\phi}^{(N)}(\lambda_i^{(N)})}{||\boldsymbol{\phi}^{(N)}(\lambda_i^{(N)})||_2}, \qquad i = 0, \ldots, k-1, \tag{4.33}$$

where $\boldsymbol{\phi}^{(N)}(y) = [\,\phi_0^{(N)}(y)\,,\,\phi_1^{(N)}(y)\,,\,\ldots\,,\,\phi_{k-1}^{(N)}(y)\,]^\top$ and $\phi_i^{(N)}(y)$ denotes the ith orthonormal polynomial relative to $\pi_y^{(N)}$. The quadrature weight associated with $\lambda_i^{(N)}$ is given by the square of the first element of the ith normalized eigenvector,

$$\omega_i^{(N)} \;=\; (\boldsymbol{Q})_{1,i+1}^2, \qquad i = 0, \ldots, k-1. \tag{4.34}$$

From Section 4.2.3 we have convergence of the quadrature nodes $\lambda_i^{(N)}$ and weights $\omega_i^{(N)}$ to λ_i and ω_i, respectively, as N increases. In this sense, we consider these quantities to be approximations of the quadrature nodes and weights relative to π_y,

$$\lambda_i^{(N)} \;\approx\; \lambda_i \quad \text{and} \quad \omega_i^{(N)} \;\approx\; \omega_i. \tag{4.35}$$

In Section 4.4, we numerically study this approximation.

An alternative perspective on this k-point Gauss-Christoffel quadrature rule is that of a second discrete measure $\pi_y^{(N,k)}$ that approximates the measure $\pi_y^{(N)}$. That is,

$$\pi_y^{(N,k)} \;\approx\; \pi_y^{(N)} \;\approx\; \pi_y. \tag{4.36}$$

By taking more Lanczos iterations, we improve the leftmost approximation, and for $k = N$, we have $\pi_y^{(N,N)} = \pi_y^{(N)}$ (in exact arithmetic assuming that $f_0, \ldots, f_{N-1}$ from (4.31) are distinct). By increasing N (the number of points in the numerical integration rule with respect to π_x), we improve the rightmost approximation. This may be viewed as convergence of the discrete Lanczos algorithm to the continuous Stieltjes procedure. This mirrors the two-level approximation from (4.7). Both contain approximation over $\mathcal{X}$ by a chosen integration rule and approximation over $\mathcal{F}$ by an integration rule resulting from the different algorithms. The key differences is in the quality of those integration rules over $\mathcal{F}$—Gauss-Christoffel quadrature in (4.36) versus Riemann sums in (4.7).

The Lanczos vectors resulting from performing the Lanczos algorithm on (4.31) also contain useful information [26]. The Lanczos vectors are of the form

$$(\boldsymbol{V})_{i+1,j+1} \;\approx\; \sqrt{\nu_i}\,\phi_j(f_i), \qquad \begin{array}{l} i = 0, \ldots, N-1, \\ j = 0, \ldots, k-1, \end{array} \tag{4.37}$$

where $f_i = f(x_i)$, ν_i is the weight associated with the node x_i, and ϕ_j is the jth-degree orthonormal polynomial with respect to π_y. The approximation in (4.37) is due to the approximation of π_y by $\pi_y^{(N)}$ and is in the same vein as the approximation in (4.35). We also numerically study this approximation in Section 4.4.

The approximation method in [26] suggests evaluating g at the $k \ll N$ quadrature nodes obtained from the Jacobi matrix and using these evaluations to construct a pseudospectral approximation. This allows for accurate estimation of g while placing a majority of the computational cost on evaluating f instead of both f and g in (4.29). Such an approach is valuable when g is difficult to compute relative to f. In the next section, we explain how this methodology can be used to construct approximations to $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$ from (4.3).

4.3 Lanczos-Stieltjes methods for inverse regression

In this section, we use the tools reviewed in Section 4.2 to develop a new Lanczos-Stieltjes approach to inverse regression methods—specifically to approximate the matrices $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$. This approach avoids approximating $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$ by Riemann sums (or slicing) as discussed in Section 4.1. Instead, we use orthonormal polynomials and quadrature approximations to build more accurate estimates of these matrices. Note that for these algorithms, we assume standardized inputs (see Equation (1.2)) as we have been assuming throughout. We also assume that $\pi_{\mathbf{x}}$ is such that the inputs have finite fourth moments.

4.3.1 Lanczos-Stieltjes inverse regression (LSIR)

In Section 4.1, we showed that the SIR algorithm approximates the matrix

$$\boldsymbol{C}_{\text{IR}} = \int \boldsymbol{\mu}(y)\, \boldsymbol{\mu}(y)^\top \, \mathrm{d}\pi_y(y) \tag{4.38}$$

using a sliced mapping of the output—i.e., a Riemann sum approximation. We wish to approximate $\boldsymbol{C}_{\text{IR}}$ without such slicing; however, the structure of $\boldsymbol{\mu}(y)$ makes this difficult. Recall that the conditional expectation is the average of the inverse image of $f(\mathbf{x})$ for a fixed value of y,

$$\boldsymbol{\mu}(y) \;=\; \int \mathbf{x}\, \mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}). \tag{4.39}$$

Approximating $\boldsymbol{\mu}(y)$ requires knowledge of the conditional measure $\pi_{\mathbf{x}|y}$, which may not be available if f is complex. However, using the tools from Section 4.2, we can exploit composite structure in $\boldsymbol{\mu}(y)$.

The conditional expectation $\boldsymbol{\mu}(y)$ is a function that maps values of y to values in $\mathbb{R}^m$. Furthermore, y is itself a function of $\mathbf{x}$; see (4.1). Thus, the conditional expectation has composite structure. That is, we can define the function

$$\boldsymbol{\mu}_{\mathbf{x}}(\mathbf{x}) \;=\; \boldsymbol{\mu}(f(\mathbf{x})), \tag{4.40}$$

where (using the notation from (4.30))

$$\begin{aligned} f &: (\mathcal{X} \subseteq \mathbb{R}^m) \to (\mathcal{F} \subseteq \mathbb{R}), \\ \boldsymbol{\mu} &: (\mathcal{F} \subseteq \mathbb{R}) \to (\mathcal{G} \subseteq \mathbb{R}^m). \end{aligned} \tag{4.41}$$

Using the techniques from Section 4.2.4, we seek to construct a pseudospectral expansion of $\boldsymbol{\mu}$ using the orthogonal polynomials and Gauss-Christoffel quadrature with respect to π_y.

By Jensen's inequality,

$$\boldsymbol{\mu}(y)^\top \boldsymbol{\mu}(y) \;\leq\; \int \mathbf{x}^\top \mathbf{x}\, \mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}), \tag{4.42}$$

Integrating both sides of (4.42) with respect to π_y,

$$\begin{aligned} \int \boldsymbol{\mu}(y)^\top \boldsymbol{\mu}(y)\, \mathrm{d}\pi_y(y) \;&\leq\; \iint \mathbf{x}^\top \mathbf{x}\, \mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x})\, \mathrm{d}\pi_y(y) \\ &=\; \int \mathbf{x}^\top \mathbf{x}\, \mathrm{d}\pi_{\mathbf{x}}(\mathbf{x}). \end{aligned} \tag{4.43}$$

Since we assume the inputs probability measure has finite fourth moments, the right-hand side of (4.43) is finite. This guarantees that each component of $\boldsymbol{\mu}(y)$ is square integrable with respect to π_y, ensuring that $\boldsymbol{\mu}(y)$ has a Fourier expansion in orthonormal polynomials with respect to π_y,

$$\boldsymbol{\mu}(y) \;=\; \sum_{i=0}^{\infty} \boldsymbol{\mu}_i\, \phi_i(y) \tag{4.44}$$

with equality in the $L_2(\pi_y)$ sense. The Fourier coefficients in (4.44) are

$$\boldsymbol{\mu}_i \;=\; \int \boldsymbol{\mu}(y)\,\phi_i(y)\,\mathrm{d}\pi_y(y). \tag{4.45}$$

Plugging (4.44) into (4.38),

$$\begin{aligned} \boldsymbol{C}_{\mathrm{IR}} \;&=\; \int \boldsymbol{\mu}(y)\,\boldsymbol{\mu}(y)^\top\,\mathrm{d}\pi_y(y) \\ &=\; \int \left[\sum_{i=0}^{\infty} \boldsymbol{\mu}_i\,\phi_i(y)\right]\left[\sum_{j=0}^{\infty} \boldsymbol{\mu}_j\,\phi_j(y)\right]^\top \mathrm{d}\pi_y(y) \\ &=\; \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \boldsymbol{\mu}_i\,\boldsymbol{\mu}_j^\top \left[\int \phi_i(y)\,\phi_j(y)\,\mathrm{d}\pi(y)\right] \\ &=\; \sum_{i=0}^{\infty} \boldsymbol{\mu}_i\,\boldsymbol{\mu}_i^\top. \end{aligned} \tag{4.46}$$

Thus, the $\boldsymbol{C}_{\mathrm{IR}}$ matrix can be computed as the sum of the outer products of Fourier coefficients from (4.45).

We cannot compute the Fourier coefficients directly since they require knowledge of $\boldsymbol{\mu}(y)$. However, we can rewrite (4.45) as

$$\begin{aligned} \boldsymbol{\mu}_i \;&=\; \int \boldsymbol{\mu}(y)\,\phi_i(y)\,\mathrm{d}\pi_y(y) \\ &=\; \int \left[\int \mathbf{x}\,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x})\right] \phi_i(y)\,\mathrm{d}\pi_y(y) \\ &=\; \iint \mathbf{x}\,\phi_i(y)\,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x})\,\mathrm{d}\pi_y(y) \\ &=\; \int \mathbf{x}\,\phi_i(f(\mathbf{x}))\,\mathrm{d}\pi_{\mathbf{x}}(\mathbf{x}). \end{aligned} \tag{4.47}$$

This form of the Fourier coefficients is more amenable to numerical approximation. Since $\pi_{\mathbf{x}}$ is known, we can obtain an N-point integration rule with nodes $\mathbf{x}_j$ and weights ν_j. We then define the pseudospectral coefficients with respect to the N-point multivariate integration rule,

$$\hat{\boldsymbol{\mu}}_i \;=\; \sum_{j=0}^{N-1} \nu_j\,\mathbf{x}_j\,\phi_i(f(\mathbf{x}_j)). \tag{4.48}$$

To evaluate ϕ_i at $f(\mathbf{x}_j)$, we use the Lanczos vectors in $\boldsymbol{V}$. Recall from (4.37) that these vectors approximate the orthonormal polynomials from Stieltjes at f evaluated at the nodes scaled by the

Algorithm 5 Lanczos-Stieltjes inverse regression (LSIR)

Given: The function $f : \mathbb{R}^m \to \mathbb{R}$ and input probability measure $\pi_{\mathbf{x}}$.

Assumptions: The input probability measure $\pi_{\mathbf{x}}$ is such (1.2) holds and it have finite fourth moments.

(1) Obtain the nodes $\mathbf{x}_i$ and weights ν_i for an N-point integration rule with respect to $\pi_{\mathbf{x}}$.

(2) Evaluate $f_i = f(\mathbf{x}_i)$ for $i = 0, \dots, N-1$.

(3) Perform k iterations of Algorithm 4 on

$$\boldsymbol{A} = \begin{bmatrix} f_0 & & \\ & \ddots & \\ & & f_{N-1} \end{bmatrix}, \qquad \tilde{\mathbf{v}}_0 = \begin{bmatrix} \sqrt{\nu_0} \\ \sqrt{\nu_1} \\ \vdots \\ \sqrt{\nu_{N-1}} \end{bmatrix} \tag{4.49}$$

to obtain $\boldsymbol{AV} = \boldsymbol{VT} + \eta_k \mathbf{v}_k \mathbf{e}_k^\top$.

(4) For $i = 0, \dots, m-1$, $\ell = 0, \dots, k-1$,
Compute the ith component of the ℓth pseudospectral coefficient

$$(\hat{\boldsymbol{\mu}}_\ell)_{i+1} = \sum_{p=0}^{N-1} \sqrt{\nu_p}\, (\mathbf{x}_p)_{i+1}\, (\boldsymbol{V})_{p+1,\ell+1}\,. \tag{4.50}$$

(5) For $i, j = 0, \dots, m-1$,
Compute the i,jth component of $\hat{\boldsymbol{C}}_{\text{IR}}$

$$\left(\hat{\boldsymbol{C}}_{\text{IR}}\right)_{i+1,j+1} = \sum_{\ell=0}^{k-1} (\hat{\boldsymbol{\mu}}_\ell)_{i+1}\, (\hat{\boldsymbol{\mu}}_\ell)_{j+1}\,. \tag{4.51}$$

Output: The matrix $\hat{\boldsymbol{C}}_{\text{IR}}$.

square root of the associated weights. Algorithm 5 formalizes this process as the Lanczos-Stieltjes inverse regression (LSIR) algorithm.

Algorithm 5 depends on two levels of approximation: (i) approximation due to the numerical integration rule on $\mathcal{X}$ and (ii) approximation due to truncating the polynomial expansion of $\boldsymbol{\mu}(y)$. The former depends on the number N of points in the numerical integration rule over $\mathcal{X}$ while the latter depends on the number k of Lanczos iterations. Performing more Lanczos iterations includes more terms in the approximation of $\boldsymbol{C}_{\text{IR}}$; however, the additional terms correspond to integrals against higher degree polynomials in (4.45). For fixed N, the quality of the pseudospectral approximation deteriorates as the degree of polynomial increases. Therefore, sufficiently many points are needed to ensure quality estimates of the orthonormal polynomials of high degree. We explore this phonomenon in Section 4.4.

To compare the computational cost of the LSIR algorithm to its slice-based counterpart, we consider the approximate costs of the Lanczos method and the slicing procedure. Due to its origins as an iterative procedure for approximating eigenvalues, the computational costs of the Lanczos algorithm have been well-studied. For the diagonal matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$, performing k iterations the Lanczos algorithm requires $\mathcal{O}(kN)$ operations [64, Chap. 9]. The costs associated with the SIR algorithm arise from sorting the outputs in order to define the slices. Sorting algorithms are known to take an average of $\mathcal{O}(N \log(N))$ operations [3]. However, in practice the most significant cost is typically the cost of the evaluations $f_i = f(\mathbf{x}_i)$—a necessary step for both the Lanczos-Stieltjes and the slice-based approaches.

4.3.2 Lanczos-Stieltjes average variance estimation (LSAVE)

In this section, we apply the Lanczos-Stieltjes approach from Section 4.3.1 to the $\boldsymbol{C}_{\text{AVE}}$ matrix to construct an alternative algorithm to the slice-based SAVE. Recall from (4.3) that

$$\boldsymbol{C}_{\text{AVE}} \;=\; \int \left(\boldsymbol{I} - \boldsymbol{\Sigma}(y)\right)^2 \, \mathrm{d}\pi_y(y), \tag{4.52}$$

where the conditional covariance of the inverse image of $f(\mathbf{x})$ for a fixed value of y is

$$\boldsymbol{\Sigma}(y) \;=\; \int \left(\mathbf{x} - \boldsymbol{\mu}(y)\right)\left(\mathbf{x} - \boldsymbol{\mu}(y)\right)^\top \, \mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}). \tag{4.53}$$

Similar to the conditional expectation, $\boldsymbol{\Sigma}(y)$ has composite structure due to the relationship $y = f(\mathbf{x})$ such that we can define

$$\boldsymbol{\Sigma}_{\mathbf{x}}(\mathbf{x}) \;=\; \boldsymbol{\Sigma}(f(\mathbf{x})), \tag{4.54}$$

where

$$\begin{aligned} &f : (\mathcal{X} \subseteq \mathbb{R}^m) \rightarrow (\mathcal{F} \subseteq \mathbb{R}), \\ &\boldsymbol{\Sigma} : (\mathcal{F} \subseteq \mathbb{R}) \rightarrow \left(\mathcal{G} \subseteq \mathbb{R}^{m \times m}\right). \end{aligned} \tag{4.55}$$

We want to build a pseudospectral expansion of $\mathbf{\Sigma}$ with respect to π_y similar to $\boldsymbol{\mu}$ in Section 4.3.1.

By Jensen's inequality,

$$\|\mathbf{\Sigma}(y)\|_F^2 \;\le\; \int \left\| \left(\mathbf{x}-\boldsymbol{\mu}(y)\right)\left(\mathbf{x}-\boldsymbol{\mu}(y)\right)^\top \right\|_F^2 \,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}). \tag{4.56}$$

Integrating each side with respect to π_y,

$$\begin{aligned}
&\int \|\mathbf{\Sigma}(y)\|_F^2 \,\mathrm{d}\pi_y(y) \\
&\quad \le \iint \left\| \left(\mathbf{x}-\boldsymbol{\mu}(y)\right)\left(\mathbf{x}-\boldsymbol{\mu}(y)\right)^\top \right\|_F^2 \,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x})\,\mathrm{d}\pi_y(y) \\
&\quad \le \int \left\| \left(\mathbf{x}-\boldsymbol{\mu}(f(\mathbf{x}))\right)\left(\mathbf{x}-\boldsymbol{\mu}(f(\mathbf{x}))\right)^\top \right\|_F^2 \,\mathrm{d}\pi_{\mathbf{x}}(\mathbf{x}).
\end{aligned} \tag{4.57}$$

Expanding the integrand on the right-hand side produces sums and products of fourth and lower conditional moments of the inputs. The assumption of finite fourth moments guarantees that all of these conditional moments are finite. Thus,

$$\int \|\mathbf{\Sigma}(y)\|_F^2 \,\mathrm{d}\pi_y(y) \;<\; \infty, \tag{4.58}$$

which implies that each component of $\mathbf{\Sigma}(y)$ is square integrable with respect to π_y; therefore it has a convergent Fourier expansion in terms of orthonormal polynomials with respect to π_y,

$$\mathbf{\Sigma}(y) \;=\; \sum_{i=0}^{\infty} \mathbf{\Sigma}_i\,\phi_i(y), \tag{4.59}$$

where equality is in the $L_2(\pi_y)$ sense. The coefficients in (4.59) are

$$\mathbf{\Sigma}_i \;=\; \int \mathbf{\Sigma}(y)\,\phi_i(y)\,\mathrm{d}\pi_y(y). \tag{4.60}$$

Plugging (4.59) into (4.52)

$$\begin{aligned}
\boldsymbol{C}_{\mathrm{AVE}} &= \int \left(\boldsymbol{I}-\mathbf{\Sigma}(y)\right)^2 \,\mathrm{d}\pi_y(y) \\
&= \int \left(\boldsymbol{I}-\sum_{i=0}^{\infty}\mathbf{\Sigma}_i\,\phi_i(y)\right)^2 \,\mathrm{d}\pi_y(y) \\
&= \int \boldsymbol{I}\,\mathrm{d}\pi_y(y) - 2\sum_{i=0}^{\infty}\mathbf{\Sigma}_i \int \phi_i(y)\,\mathrm{d}\pi_y(y) \\
&\qquad + \sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\mathbf{\Sigma}_i\,\mathbf{\Sigma}_j \int \phi_i(y)\,\phi_j(y)\,\mathrm{d}\pi_y(y) \\
&= \boldsymbol{I} - 2\,\mathbf{\Sigma}_0 + \sum_{i=0}^{\infty}\mathbf{\Sigma}_i^2.
\end{aligned} \tag{4.61}$$

Therefore, we can compute $\boldsymbol{C}_{\mathrm{AVE}}$ using the Fourier coefficients of $\boldsymbol{\Sigma}(y)$. To simplify the computation of $\boldsymbol{\Sigma}_i$, we rewrite 4.60 as

$$\begin{aligned}\boldsymbol{\Sigma}_i &= \int \boldsymbol{\Sigma}(y)\,\phi_i(y)\,\mathrm{d}\pi_y(y)\\ &= \int\int (\mathbf{x}-\boldsymbol{\mu}(y))\,(\mathbf{x}-\boldsymbol{\mu}(y))^\top\,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x})\,\phi_i(y)\,\mathrm{d}\pi_y(y)\\ &= \int\int (\mathbf{x}-\boldsymbol{\mu}(y))\,(\mathbf{x}-\boldsymbol{\mu}(y))^\top\,\phi_i(y)\,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x})\,\mathrm{d}\pi_y(y)\\ &= \int (\mathbf{x}-\boldsymbol{\mu}(f(\mathbf{x})))\,(\mathbf{x}-\boldsymbol{\mu}(f(\mathbf{x})))^\top\,\phi_i(f(\mathbf{x}))\,\mathrm{d}\pi_{\mathbf{x}}(\mathbf{x}).\end{aligned} \tag{4.62}$$

We approximate this integral using the N-point numerical integration rule with respect to $\pi_{\mathbf{x}}$ to obtain

$$\hat{\boldsymbol{\Sigma}}_i = \sum_{j=0}^{N-1} \nu_j\,(\mathbf{x}_j-\boldsymbol{\mu}(f(\mathbf{x}_j)))\,(\mathbf{x}_j-\boldsymbol{\mu}(f(\mathbf{x}_j)))^\top\,\phi_i(f(\mathbf{x}_j)). \tag{4.63}$$

We again approximate $\phi_i(f(\mathbf{x}_j))$ using the Lanczos vectors similar to LSIR; see (4.37). Notice that (4.63) also depends on $\boldsymbol{\mu}(f(\mathbf{x}_j))$. To obtain these values, we compute the pseudospectral coefficients of $\boldsymbol{\mu}(f(\mathbf{x}))$ from (4.48) and construct its pseudospectral expansion at each $\mathbf{x}_j$. Algorithm 6 provides an outline for Lanczos-Stieltjes average variance estimation (LSAVE).

Algorithm 6 contains the same two-level approximation as the LSIR algorithm. As such, it also requires a sufficiently high-order integration rule to accurately approximate the high degree polynomials resulting from k Lanczos iterations. In the next section, we provide numerical studies of the LSIR and LSAVE algorithms on several test problems as well as comparisons to the traditional SIR and SAVE algorithms.

4.4 Numerical results

In this section, we numerically study the LSIR and LSAVE algorithms. The error analysis for estimating of the ridge directions in Chapter 3 depends on how well $\hat{\boldsymbol{C}}_{\mathrm{SIR}}$ and $\hat{\boldsymbol{C}}_{\mathrm{SAVE}}$ approximate their respective population matrices. Therefore, in this section, we focus on the approximation the population matrices by $\hat{\boldsymbol{C}}_{\mathrm{IR}}$ and $\hat{\boldsymbol{C}}_{\mathrm{AVE}}$—i.e., the matrices constructed by the LSIR and LSAVE

Algorithm 6 Lanczos-Stieltjes average variance estimation (LSAVE)

Given: The function $f : \mathbb{R}^m \to \mathbb{R}$ and input probability measure $\pi_{\mathbf{x}}$.
Assumptions: The input probability measure $\pi_{\mathbf{x}}$ is such (1.2) holds and it have finite fourth moments.

(1) Obtain the nodes $\mathbf{x}_i$ and weights ν_i for an N-point integration rule with respect to $\pi_{\mathbf{x}}$.

(2) Evaluate $f_i = f(\mathbf{x}_i)$ for $i = 0, \dots, N-1$.

(3) Perform k iterations of Algorithm 4 on

$$\boldsymbol{A} = \begin{bmatrix} f_0 & & \\ & \ddots & \\ & & f_{N-1} \end{bmatrix}, \qquad \tilde{\mathbf{v}}_0 = \begin{bmatrix} \sqrt{\nu_0} \\ \sqrt{\nu_1} \\ \vdots \\ \sqrt{\nu_{N-1}} \end{bmatrix} \tag{4.64}$$

to obtain $\boldsymbol{AV} = \boldsymbol{VT} + \eta_k \mathbf{v}_k \mathbf{e}_k^\top$.

(4) For $i = 0, \dots, m-1$, $\ell = 0, \dots, k-1$,
Compute the ith component of the ℓth pseudospectral coefficient of $\boldsymbol{\mu}(y)$

$$(\hat{\boldsymbol{\mu}}_\ell)_{i+1} = \sum_{p=0}^{N-1} \sqrt{\nu_p}\, (\mathbf{x}_p)_{i+1} (\boldsymbol{V})_{p+1,\ell+1}\,. \tag{4.65}$$

(5) For $i = 0, \dots, m-1$,
Compute the ith component of the pseudospectral expansion of $\boldsymbol{\mu}(f(\mathbf{x}_p))$

$$(\hat{\boldsymbol{\mu}}(f(\mathbf{x}_p)))_{i+1} = \sum_{\ell=0}^{k-1} \frac{1}{\sqrt{\nu_p}} (\hat{\boldsymbol{\mu}}_\ell)_{i+1} (\boldsymbol{V})_{p+1,\ell+1}\,. \tag{4.66}$$

(6) For $i, j = 0, \dots, m-1$, $\ell = 0, \dots, k-1$,
Compute the i,jth component of the ℓth pseudospectral coefficient of $\boldsymbol{\Sigma}(y)$

$$\left(\hat{\boldsymbol{\Sigma}}_\ell\right)_{i+1,j+1} = \sum_{p=0}^{N-1} \sqrt{\nu_p} \left((\mathbf{x}_p)_{i+1} - (\hat{\boldsymbol{\mu}}(f(\mathbf{x}_p)))_{i+1}\right) \left((\mathbf{x}_p)_{j+1} - (\hat{\boldsymbol{\mu}}(f(\mathbf{x}_p)))_{j+1}\right) (\boldsymbol{V})_{p+1,\ell+1}\,. \tag{4.67}$$

(7) For $i, j = 0, \dots, m-1$,
Compute the i,jth component of $\hat{\boldsymbol{C}}_{\text{AVE}}$

$$\left(\hat{\boldsymbol{C}}_{\text{AVE}}\right)_{i+1,j+1} = \delta_{i,j} - 2\left(\hat{\boldsymbol{\Sigma}}_0\right)_{i+1,j+1} + \sum_{\ell=0}^{k-1} \sum_{p=0}^{m-1} \left(\hat{\boldsymbol{\Sigma}}_\ell\right)_{i+1,p+1} \left(\hat{\boldsymbol{\Sigma}}_\ell\right)_{p+1,j+1}\,. \tag{4.68}$$

Output: The matrix $\hat{\boldsymbol{C}}_{\text{AVE}}$.

algorithms. We measure this error using the Frobenius norm

$$\|\boldsymbol{E}\|_F \;=\; \left(\sum_{i=1}^{m}\sum_{j=1}^{m}(\boldsymbol{E})_{i,j}^2\right)^{1/2}, \tag{4.69}$$

where $\boldsymbol{E}$ is the error in the approximated matrix.

4.4.1 Quadratic problem

This study examines the approximation of the Gauss-Christoffel quadrature and orthonormal polynomials on the output space by the Lanczos algorithm as described in Section 4.2.4. These components are central the Algorithms 5 and 6 so it is important to understand their convergence properties. For this study, we consider the function

$$y \;=\; f(\mathbf{x}) \;=\; \mathbf{g}^\top\mathbf{x} + \mathbf{x}^\top\boldsymbol{H}\mathbf{x}, \qquad \mathbf{x}\in[-1,1]^3\subset\mathbb{R}^3, \tag{4.70}$$

where $\mathbf{g}\in\mathbb{R}^3$ is a constant vector and $\boldsymbol{H}\in\mathbb{R}^{3\times 3}$ is a constant matrix. We assume the inputs are weighted by the uniform density over the input space $\mathcal{X}=[-1,1]^3$,

$$\mathrm{d}\pi_{\mathbf{x}}(\mathbf{x}) \;=\; \begin{cases}\frac{1}{2^3}\,\mathrm{d}\mathbf{x} & \text{if } ||\mathbf{x}||_\infty \le 1,\\ 0\,\mathrm{d}\mathbf{x} & \text{otherwise.}\end{cases} \tag{4.71}$$

Recall from Section 4.2.4 that we can use the Lanczos algorithm to obtain a k-point Gauss-Christoffel quadrature rule and the first k orthonormal polynomials relative to the discrete measure $\pi_y^{(N)}$. We treat these as approximations to the quadrature rule and orthonormal polynomials relative to the continuous measure π_y (see (4.35) and (4.37)). In this section, we study the behavior of these approximations for (4.70).

We use two different integration rules for this study: a tensor product Clenshaw-Curtis quadrature rule [14] and simple Monte Carlo [94]. We do this to emphasize that approximation accuracy of the Gauss-Christoffel quadrature rule and the orthonormal polynomials with respect to π_y depend on the quality of the integration rule chosen with respect to $\pi_{\mathbf{x}}$.

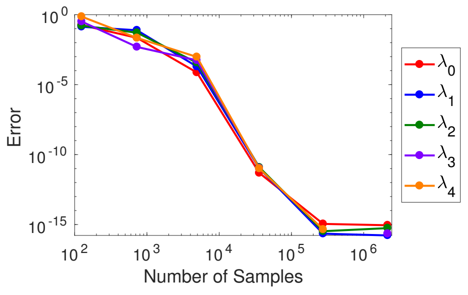
Recall from (4.37) that the Lanczos vectors contain evaluations of the first k (corresponding to the number of Lanczos iterations performed) orthonormal polynomials with respect to $\pi_y^{(N)}$ at the points $f_i = f(\mathbf{x}_i)$ weighted by the square root of the associated weights, $\sqrt{\nu_i}$. To compare the approximations for increasing numbers of samples, we must ensure that our integration rules are nested. That is, the $\mathbf{x}_i$'s of the N_j-point integration rule must be included in the N_{j+1}-point integration rule, where N_j and N_{j+1} denote subsequently increasing numbers of points. We use the Clenshaw-Curtis quadrature rule here because it is a nested quadrature rule. For Monte Carlo, we append new independent random samples to our current set to ensure the nested structure.

First, we study convergence of the Gauss-Christoffel quadrature rule produced on the output space $\mathcal{F}$ by the Lanczos algorithm. The convergence of the Jacobi matrix (4.26) to (4.14) as the discrete approximation $\pi_y^{(N)}$ approaches π_y has been explore previous [54]; however, we are specifically interested in the quadrature rule resulting from the eigendecomposition of the Jacobi matrix. Recall from (4.35) that $\lambda_i^{(N)}$ and $\omega_i^{(N)}$ denote the ith Gauss-Christoffel quadrature node and weight with respect to $\pi_y^{(N)}$. Figure 4.2 shows the differences in the 5-point quadrature rules with increasing samples over $\mathcal{X}$,

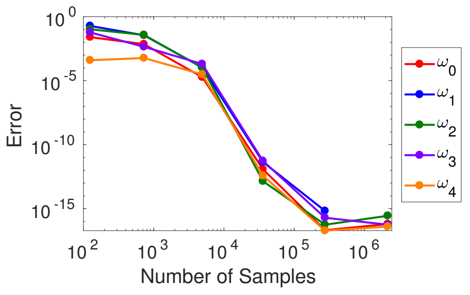
$$\left|\hat{\lambda}_i^{(N_{j+1})} - \hat{\lambda}_i^{(N_j)}\right| \quad \text{and} \quad \left|\hat{\omega}_i^{(N_{j+1})} - \hat{\omega}_i^{(N_j)}\right|, \tag{4.72}$$

for $i = 0, \dots, 4$. Using Clenshaw-Curtis quadrature rules over the input space (Figures 4.2a and 4.2b), the Gauss-Christoffel quadrature rule with respect to π_y converges exponentially. In Figures 4.2c and 4.2d, we see the expected $N^{-1/2}$ convergence in the Gauss-Christoffel quadrature rule when using Monte Carlo to approximate $\pi_{\mathbf{x}}$. Quality estimates of the quadrature rule over the output space $\mathcal{F}$ are required to produce good approximations of the $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$ using the Lanczos-Stieltjes approach.

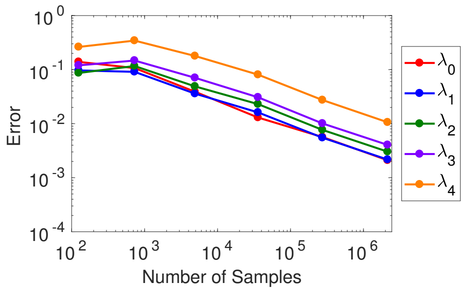
Next, we examine convergence of the Lanczos vectors to the orthonormal polynomials with respect to π_y. Let $\boldsymbol{V}^{(N_j)}$ be the matrix of k Lanczos vectors resulting from performing Lanczos with an N_j-point integration rule on the input space $\mathcal{X}$ with respect to $\pi_{\mathbf{x}}$. Define the matrix $\boldsymbol{W}_\nu = \operatorname{diag}\left(\left[\sqrt{\nu_0} \; \dots \; \sqrt{\nu_{N_j-1}}\right]\right)$, and let $\tilde{\boldsymbol{V}}^{(N_j)} = \boldsymbol{W}_\nu^{-1}\boldsymbol{V}^{(N_j)}$ be the matrix of orthonormal polynomials evaluated at the f_i's (no longer scaled by the $\sqrt{\nu_i}$'s). Due to the nestedness of the integration rules, $\tilde{\boldsymbol{V}}^{(N_j)}$ contains evaluations of the same k orthonormal polynomials at nested sets of f_i's as we



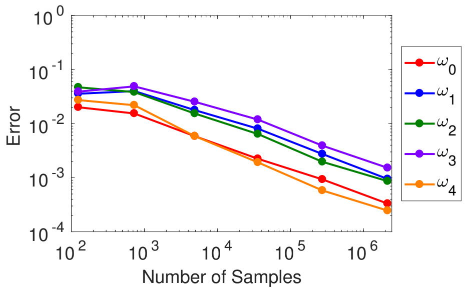
(a) Node convergence with CC



(b) Weight convergence with CC



(c) Node convergence with MC



(d) Weight convergence with MC

Figure 4.2: Differences in the 5-point Gauss-Christoffel quadrature nodes λ_i and weights ω_i resulting the Lanczos algorithm applied to (4.70). Figures 4.2a and 4.2b contain the differences using a tensor product Clenshaw-Curtis quadrature rules used over $\mathcal{X}$, and Figures 4.2c and 4.2d use Monte Carlo samples.

increase the number of samples. Let $\boldsymbol{P} \in \mathbb{R}^{N_j \times N_{j+1}}$ be the matrix the removes the rows of $\tilde{\boldsymbol{V}}^{(N_{j+1})}$ that do not correspond to rows in $\tilde{\boldsymbol{V}}^{(N_j)}$. We write the maximum difference in the ith polynomial for subsequent orders of the quadrature rule as

$$\left|\left|\boldsymbol{P}\left(\tilde{\boldsymbol{V}}^{(N_{j+1})}\right)_i - \left(\tilde{\boldsymbol{V}}^{(N_j)}\right)_i\right|\right|_\infty, \tag{4.73}$$

where $(\cdot)_i$ denotes the ith column of the given matrix for $i = 0, \dots, k-1$.

Figures 4.3a and 4.3b contain plots of the maximum differences in (4.73) for increasing numbers of Clenshaw-Curtis points and Monte Carlo points, respectively. In both plots, higher degree polynomials require more samples to produce accurate approximations. This is not surprising, but it does highlight an important relationship between the numerical integration rule used with respect to $\pi_{\mathbf{x}}$ and the number of Lanczos iterations performed. Namely, as the number of Lanczos iterations increases, more samples are required to ensure accurate approximation of the orthonormal polynomials (and, in turn, $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$). Additionally, we see that the convergence rate of the orthonormal polynomials depends on the integration rule chosen over $\mathcal{X}$.

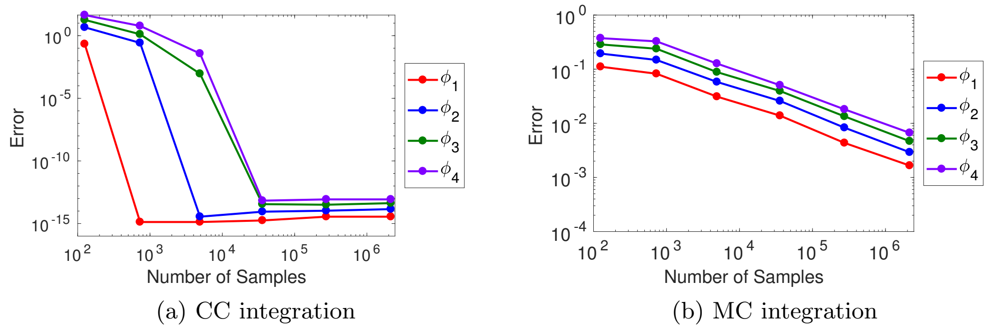
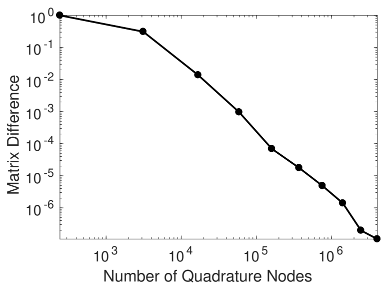


Figure 4.3: Maximum differences in the approximated orthonormal polynomials as the number of quadrature points on $\mathcal{X}$ increases. Figure 4.3a uses a tensor product Clenshaw-Curtis quadrature rule over the input space while Figure 4.3b uses Monte Carlo integration.

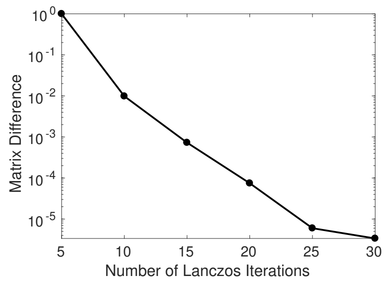
4.4.2 Hartmann problem

In this section, we study the convergence of the Lanczos-Stieltjes algorithms in terms of the number of samples and the number of Lanczos iterations. Additionally, we compare the Lanczos-Stieltjes approach to its slice-based counterpart. For these studies, we use the physically-motivated Hartmann problem from Section 2.3.1. Recall that this problem has two relevant quantities of interest: (i) the average flow velocity and (ii) the induced magnetic field. We first examine the induced magnetic field. The plots are comparable studies on the average flow velocity are included at the end of this section as well. Their behavior and the interpretation of the results is similar to that of the flow velocity.

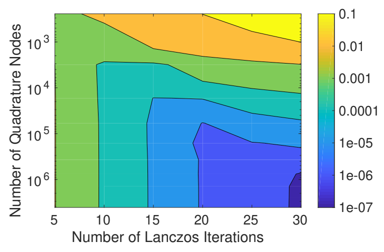
For our first study, we use tensor product Gauss-Christoffel quadrature rules on the input space. The number of points in a tensor product rule grows exponentially with dimension, so they are not appropriate for more than a handful of inputs. The point of this study is to emphasize and demonstrate how the LSIR and LSAVE algorithms remove the approximation bottleneck caused by the Riemann sums in SIR and SAVE and place the burden of accuracy on the numerical integration rule over the input space. The results for this study are contained in Figure 4.4.



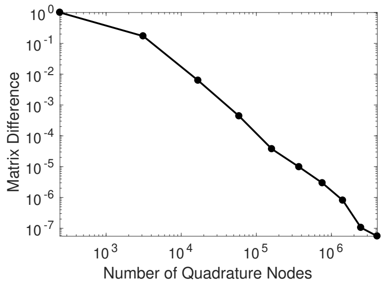
(a)



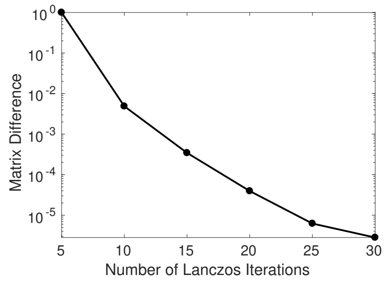
(b)



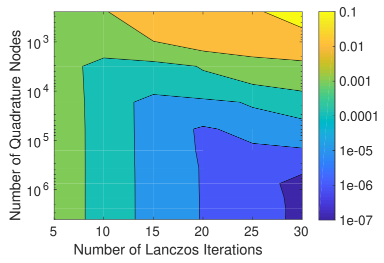
(c)



(d)



(e)



(f)

Figure 4.4: Convergence studies for the LSIR algorithm (Figures 4.4a, 4.4b, and 4.4c) and the LSAVE algorithm (Figures 4.4d, 4.4e, and 4.4f) on the induced magnetic field from the Hartmann problem (see (2.37)).

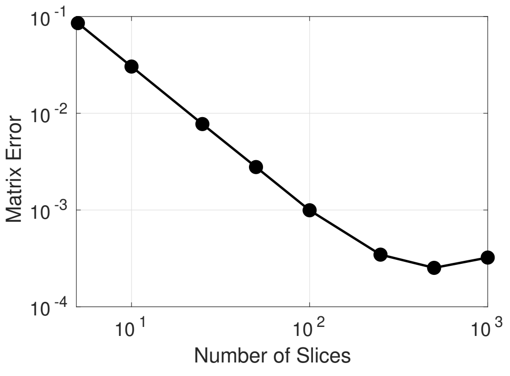
Figure 4.4a demonstrates the convergence of the LSIR algorithm in terms of the number N of Gauss-Christoffel quadrature nodes on the input space, and Figure 4.4b shows convergence in terms of the number k of Lanczos iterations performed. These plots show the relative matrix error using the Frobenius norm of the matrix differences between subsequent Lanczos-Stieltjes approximations of $\boldsymbol{C}_{\text{IR}}$ computed using increasing numbers of quadrature nodes (with $k = 35$ fixed) and Lanczos iterations (with $N = 23^5 = 6{,}436{,}343$ fixed), respectively. We see a decay in these differences as we increase N and k suggesting that the LSIR algorithm is converging.

For Figure 4.4c, we perform Algorithm 5 for $N = 23^5 = 6{,}436{,}343$ and $k = 35$ and treat the resulting matrix as the "true" value of $\boldsymbol{C}_{\text{IR}}$. We then compute errors relative to this matrix for various values of N and k. Figure 4.4c shows the relative error decaying as we increase both the number of quadrature nodes and Lanczos iterations—i.e., as we move down and to the right. Consider this plot for increasing N with k fixed—i.e., moving downward at a fixed point along the horizontal axis. The error decays up to a point at which it remains constant. This decay corresponds to more accurate computation of the pseudospectral coefficients $\hat{\boldsymbol{\mu}}_i$ from (4.48) by taking more quadrature nodes. The leveling off corresponds to the point at which errors in the coefficients are smaller than errors due to truncating the pseudospectral expansion at k (the number of Lanczos iterations). Conversely, if we fix N and study the error as we increase k—i.e., fix a point along the vertical axis and move right—we see the error decay until a point at which it begins to grow again. This behavior agrees with the results from Section 4.4.1 that suggest that sufficiently many quadrature nodes are needed to accurately estimate the high-degree polynomials associated with large values of k. As we move right in Figure 4.4c, we are approximating higher-degree polynomials using the Lanczos algorithm. Poor approximation of these polynomials results in an inaccurate estimates of $\boldsymbol{C}_{\text{IR}}$.

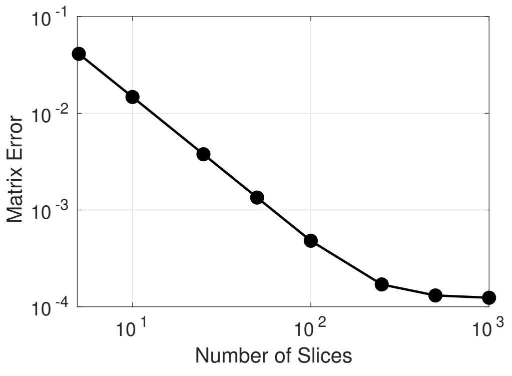
Figures 4.4d and 4.4f contain the results of the same studies as above but performed on the LSAVE algorithm (Algorithm 6). The results and interpretations are similar to those for the LSIR algorithm.

Next, we examine how the approximated SIR and SAVE matrices from Algorithms 1 and 2, respectively, compare to the LSIR and LSAVE approximations. Recall $\hat{\boldsymbol{C}}_{\text{SIR}}$ and $\hat{\boldsymbol{C}}_{\text{SAVE}}$ contain two levels of approximation—one due to the number of samples N and one due to the number of terms in the Riemann sum—i.e., the number of slices—R over the output space. We first focus on convergence in terms of Riemann sums. Figure 4.5 compares the approximated SIR and SAVE matrices for increasing R to their Lanczos-Stieltjes counterparts. For the Lanczos-Stieltjes approximations—$\hat{\boldsymbol{C}}_{\text{IR}}$ and $\hat{\boldsymbol{C}}_{\text{AVE}}$—we use $N = 23^5 = 6{,}436{,}343$ Gauss-Christoffel quadrature nodes and $k = 35$ Lanczos iterations as this produces sufficiently converged matrices; see the previous numerical study. For the SIR and SAVE algorithms, we use $N = 10^8$ samples randomly drawn according to $\pi_{\mathbf{x}}$. The sliced approximations converge to their Lanczos-Stieltjes counterparts at a rate R^{-1} as expected for Riemann sums [39, Chap. 2].

Lastly, we compare the slice-based algorithms SIR and SAVE to their Lanczos-Stieltjes counterparts LSIR and LSAVE with Monte Carlo integration on the input space. This comparison is the most appropriate for practical models with several input parameters, where tensor product



(a)

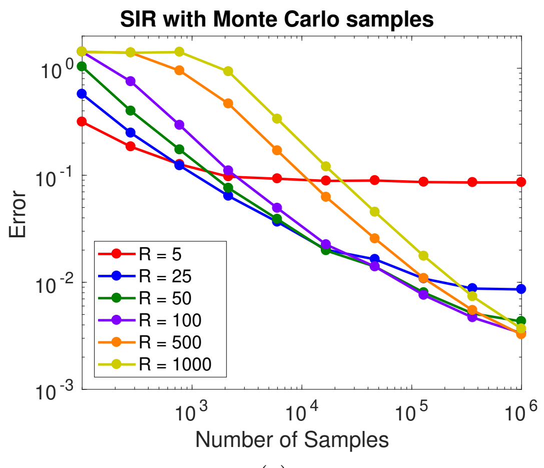


(b)

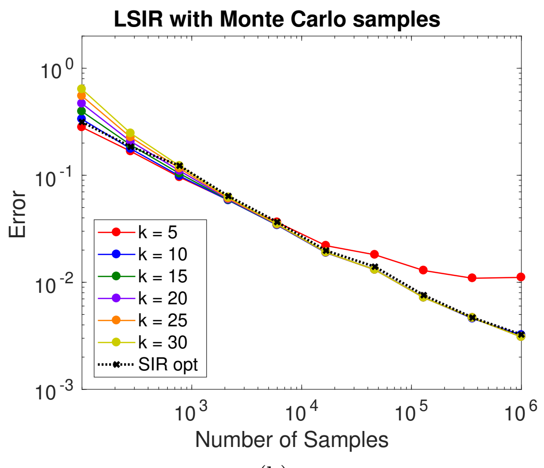
Figure 4.5: A comparison of the Riemann sum approximation and the Lanczos-Stieltjes approximation of $\boldsymbol{C}_{\mathrm{IR}}$ (Figure 4.5a) and $\boldsymbol{C}_{\mathrm{AVE}}$ (Figure 4.5b) for increasing number of Riemann sums (or slices) for the induced magnetic field. For the Lanczos-Stieltjes approximations, we used $N = 23^5 =$ 6,436,343 Gauss-Christoffel quadrature nodes on $\mathcal{X}$ and $k = 35$ Lanczos iterations. For the slice-based approximations, we used $N = 10^8$ Monte Carlo samples.

quadrature on the input space is infeasible.

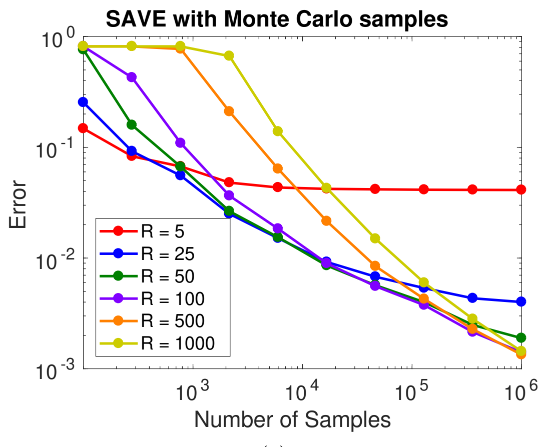
We again use the Lanczos-Stieltjes algorithms with $N = 23^5 = 6{,}436{,}343$ quadrature nodes and $k = 35$ Lanczos iterations as the "true" values of $\boldsymbol{C}_{\mathrm{IR}}$ and $\boldsymbol{C}_{\mathrm{AVE}}$ for computation of the relative matrix errors. Figures 4.6a and 4.6b compare the SIR and LSIR algorithms (Algorithms 1 and 5, respectively) for increasing numbers of Monte Carlo samples. Additionally, we perform this comparison for various values of R (the number of terms in the Riemann sum or slices) and k (the number of Lanczos iterations) for each of the methods. We notice less variance in the LSIR plot as a function of k than in the SIR plot as a function of R. The Lanczos-Stieltjes approach refines the approximation over the output space such that the final approximation depends most strongly on the chosen integration rule over the input space. That is, the issue of choosing how to slice up the output space does not exist in the Lanczos-Stieltjes approach as the method automatically chooses the best integration rule over $\mathcal{F}$. Figure 4.6b also includes the best case error from the SIR plot. This is the minimum error among all of the tested values of R for each value of N. The LSIR algorithm performs approximately as well as the best case in SIR, regardless of the value of k chosen. Figures 4.6c and 4.6d perform the same study as above comparing the SAVE and LSAVE algorithms (Algorithms 2 and 6, respectively). The results and interpretations are similar to those for the SIR/LSIR study.



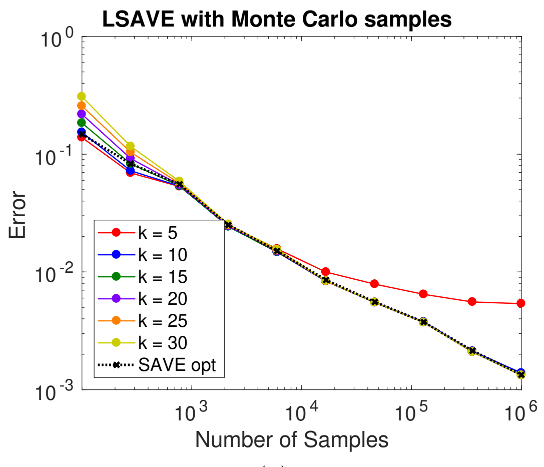
(a)



(b)



(c)



(d)

Figure 4.6: A comparison of the SIR/SAVE and LSIR/LSAVE algorithms for the induced magnetic field using Monte Carlo integration on the input space. Figures 4.6a and 4.6c show the relative matrix errors of the SIR and SAVE algorithms, respectively, as a function of the number of samples for various values of R (the number of slices). Figures 4.6b and 4.6d show the relative matrix errors of the LSIR and LSAVE algorithms, respectively, as a function of the number of samples for various values of k (the number of Lanczos iterations). These plots also show the best case results from their slice-based counterparts for reference.

For completeness, we include the analogous plots from Figures 4.4, 4.5, and 4.6 applied to the average flow velocity from (2.37). Qualitatively, the results of these studies are similar to those for the induced magnetic field and the understanding and interpretation of this is the same.

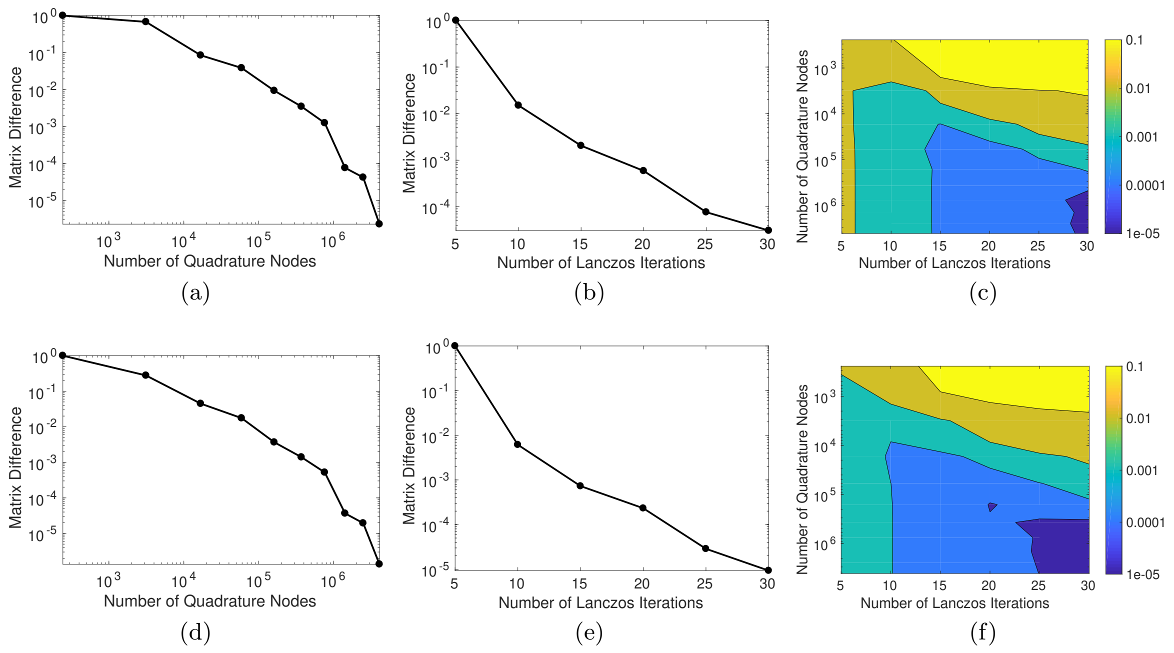


Figure 4.7: Convergence studies for the LSIR algorithm (Figures 4.7a, 4.7b, and 4.7c) and the LSAVE algorithm (Figures 4.7d, 4.7e, and 4.7f) on the average flow velocity from the Hartmann problem (see (2.36)).

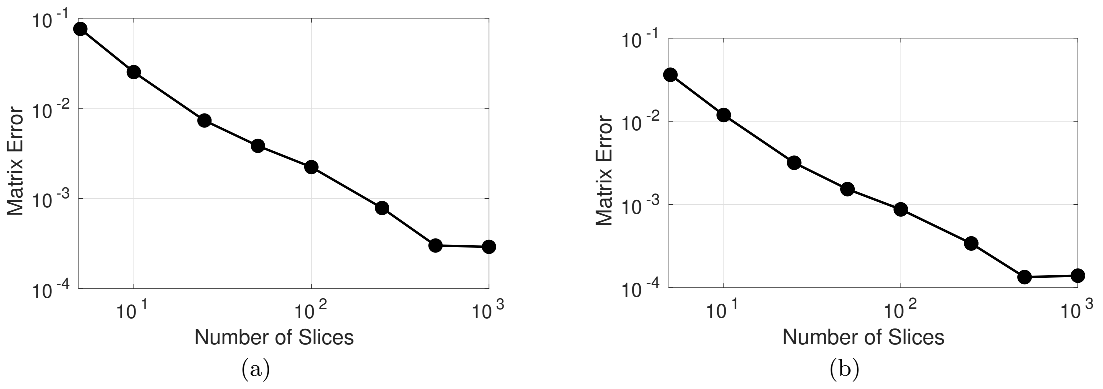
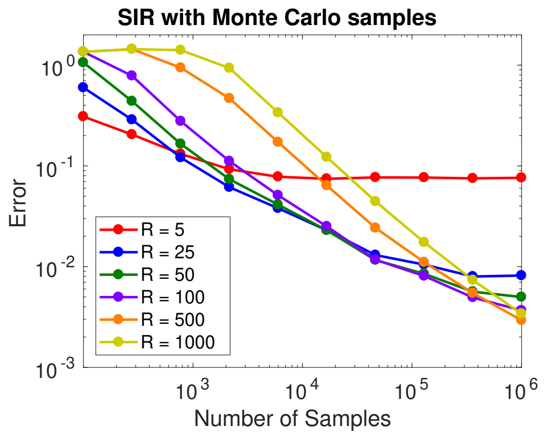
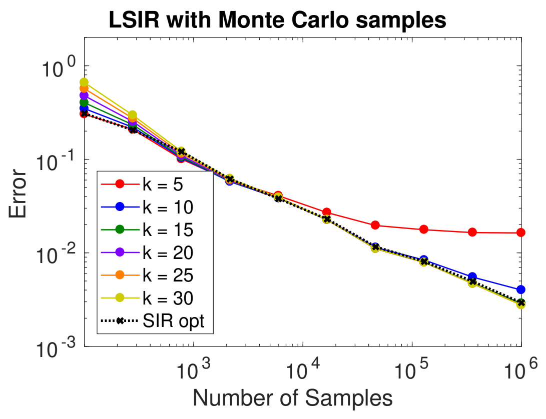


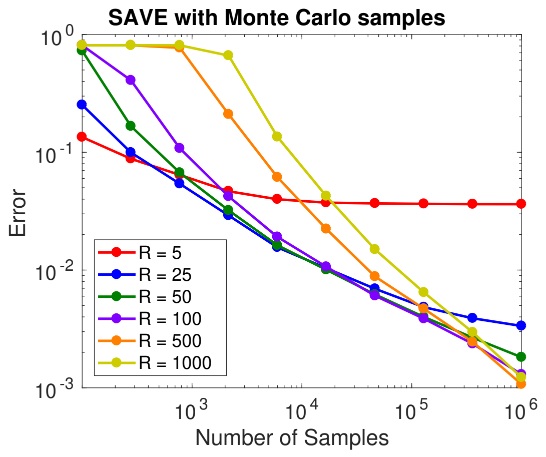
Figure 4.8: A comparison of the Riemann sum approximation and the Lanczos-Stieltjes approximation of $\boldsymbol{C}_{\text{IR}}$ (Figure 4.8a) and $\boldsymbol{C}_{\text{AVE}}$ (Figure 4.8b) for increasing number of Riemann sums (or slices) for the average flow velocity. For the Lanczos-Stieltjes approximations, we used $N = 23^5 = 6{,}436{,}343$ Gauss-Christoffel quadrature nodes on $\mathcal{X}$ and $k = 35$ Lanczos iterations. For the slice-based approximations, we used $N = 10^8$ Monte Carlo samples.



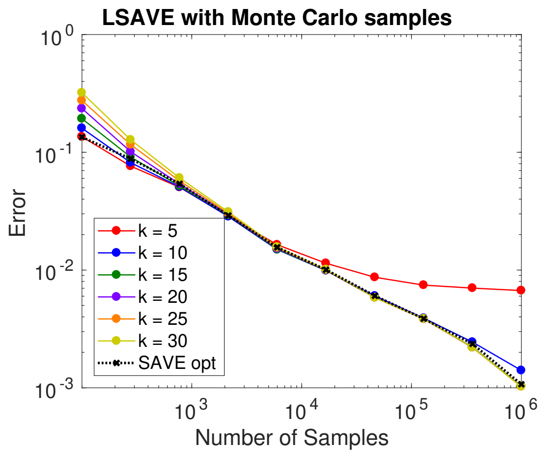
(a)



(b)



(c)



(d)

Figure 4.9: A comparison of the SIR/SAVE and LSIR/LSAVE algorithms for the average flow velocity using Monte Carlo integration on the input space. Figures 4.9a and 4.9c show the relative matrix errors of the SIR and SAVE algorithms, respectively, as a function of the number of samples for various values of R (the number of slices). Figures 4.9b and 4.9d show the relative matrix errors of the LSIR and LSAVE algorithms, respectively, as a function of the number of samples for various values of k (the number of Lanczos iterations). These plots also show the best case results from their slice-based counterparts for reference.

4.5 Summary

In this chapter, we propose alternative approaches to the sliced inverse regression (SIR) and sliced average variance estimation (SAVE) algorithms for approximating $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$. The traditional methods approximate these matrices by applying a partitioning—i.e., slicing—to the range of output values. In the context of deterministic functions, this slice-based approach can be interpreted as a Riemann sum approximation of the integrals in (4.3). The proposed algorithms use orthonormal polynomials and Gauss-Christoffel quadrature to produce high-order approximations of $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$. We call the new algorithms Lanczos-Stieltjes inverse regression (LSIR) and Lanczos-Stieltjes average variance estimation (LSAVE).

We use two numerical test problems to study convergence of the Lanczos-Stieltjes algorithms with respect to the algorithm parameters. We first examine the convergence of the approximate quadrature and orthonormal polynomial components resulting from the Lanczos method's discrete approximation of the Stieltjes procedure. This study highlights the interplay between the number of quadrature nodes on the input space and the number of Lanczos iterations. More Lanczos iterations correspond to higher degree polynomials, which require more samples for the same accuracy. Poor approximations of these polynomials lead to poor approximations of $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$. We then compare the Lanczos-Stieltjes approximations of $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$ to their slice-based counterparts. These numerical studies emphasize a key characteristic of the Lanczos-Stieltjes approaches. Due to the composite structure of $\boldsymbol{C}_{\text{IR}}$ and $\boldsymbol{C}_{\text{AVE}}$, both the slicing approach and Lanczos-Stieltjes contain two levels of approximation: (i) numerical integration on the input space $\mathcal{X}$ and (ii) approximation on the output space $\mathcal{F}$. There is a trade-off between (i) and (ii) in terms of which approximation is the dominant source of numerical error for various choices of N and R. The Lanczos-Stieltjes approach significantly reduces the errors due to approximation over $\mathcal{F}$, placing the burden of accuracy on the approximation over $\mathcal{X}$. This enables Gauss-Christoffel quadrature on $\mathcal{X}$ to produce high-order accuracy when such integration rules are appropriate—e.g., when the number of inputs is sufficiently small. When tensor product quadrature rules are infeasible, the Lanczos-Stieltjes approach allows Monte Carlo integration to perform as expected without significant dependence on the approximation on the output space $\mathcal{F}$.

Chapter 5

One-dimensional ridge function approximation and integration

The focus of this thesis so far has been on addressing the ridge recovery and ridge approximation problem—i.e., finding the ridge directions $\boldsymbol{A}$ for a given function. In this chapter, we introduce an approach for constructing a polynomial approximation of the ridge profile g for a one-dimension ridge functions. In this scenario, a single direction captures all (or most) of the variation in the output. Such an assumption may appear restrictive; however, exploitable near-one-dimensional structure has been identified in models for lithium ion batteries [21], car aerodynamics [93], integrated hydrologic models [71], hypersonic scramjet designs [16], among others [27, 57, 34]. Recall Figure 1.5, which contains several one-dimensional shadow plots of such structure.

For this work, we assume that the one important direction—i.e., the coefficients of the linear combination—is known, having been computed by some method such as active subspaces from Chapter 2 or sufficient dimension reduction from Chapters 3 and 4. By exploiting known one-dimensional structure, we can build an approximation using exponentially fewer function evaluations (which are assumed to be very expensive) than would be necessary for a comparable polynomial approximation on the full-dimensional input space. Figure 5.1 illustrates this point. This figure shows L^2 approximation errors for a five-dimensional function that is a one-dimensional ridge function. In terms of the total polynomial degree, there is no difference in performance between building the polynomial surrogate on the full five-dimensional input space versus the one-dimensional reduced input space. However, in terms of the number of function evaluations required to build each approximation, exploiting the one-dimension structure in the function results in an exponential savings in the computational costs.

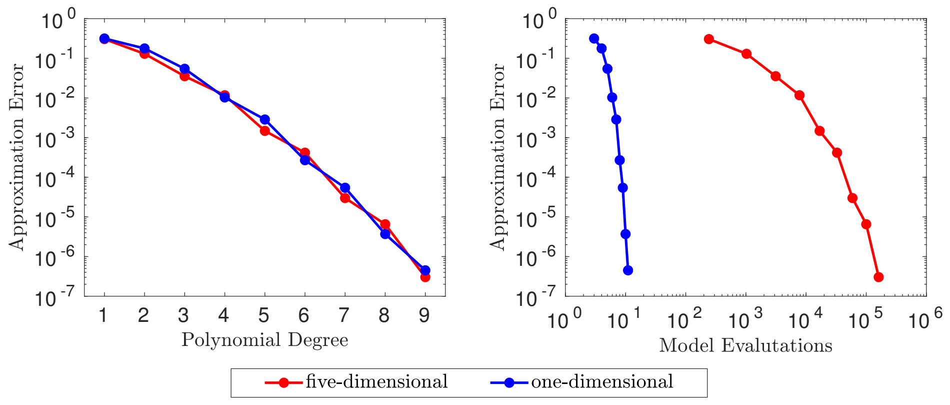


Figure 5.1: A comparison of the cost and accuracy of the five-dimensional (in red) and the one-dimensional (in blue) polynomial approximations for a given function.

One current approach for exploiting this structure assumes Gaussian process priors on the one-dimensional subspace (referred to as a single-index model) and updates the posterior using MCMC [67]. However, such approximations may converge slowly or struggle from issues related to poor mixing. Another method for approximating one-dimensional ridge functions transforms the reduced inputs through an empirical CDF to enable the use of Legendre polynomials, but this mapping results in poor approximations near the boundaries of the domain [122]. The methodology introduced in this chapter first approximates the induced density function on the reduced input space using convolutions. We then use the Lanczos iterative method as a discrete approximation of the Stieltjes procedure for constructing orthonormal polynomials and Gauss-Christoffel quadrature rules with respect to arbitrary densities. This allows us to accurately fit polynomial surrogates on the one-dimensional ridge subspace.

5.1 Background

As discussed in Chapter 1, we consider functions of the form

$$y \;=\; f(\mathbf{x}), \qquad y \in \mathbb{R}, \quad \mathbf{x} \in \mathbb{R}^m, \tag{5.1}$$

and assume the input space is weighted by a given probability measure $\pi_{\mathbf{x}}$. This measure plays an important role in the work presented in this chapter. Often, a relatively non-information input measure is given, such as a multivariate Gaussian or a uniform density over the hypercube. As we discuss later, the work in this chapter is uninteresting in the context of Gaussian distributions. For this reason, we assume that the input space is weighted by a uniform density over the $[-1,1]^m$ hypercube. This density function is

$$p(\mathbf{x}) \;=\; \begin{cases} \frac{1}{2^m} & \text{if } ||\mathbf{x}||_\infty \leq 1, \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

Note that the methodologies presented here extend to any input probability measure, provided that the components are independent.

As the title of this chapter suggests, we are interested in the specific case of one-dimension ridge functions. That is

$$y \;=\; f(\mathbf{x}) \;=\; g(u) \tag{5.3}$$

or

$$y \;=\; f(\mathbf{x}) \;\approx\; g(u), \tag{5.4}$$

where $u = \mathbf{a}^\top \mathbf{x}$ for $\mathbf{a} \in \mathbb{R}^m$ and $g : \mathbb{R} \to \mathbb{R}$.

5.1.1 Polynomial approximation

Assume that f is square-integrable with respect to the input measure. Then, we may express f as a Fourier expansion in terms of orthogonal polynomials with respect to $\pi_{\mathbf{x}}$,

$$y \;=\; f(\mathbf{x}) \;=\; \sum_{|\boldsymbol{\alpha}|=0}^{\infty} f_{\boldsymbol{\alpha}}\, \psi_{\boldsymbol{\alpha}}(\mathbf{x}), \tag{5.5}$$

where equality is denoted in the L^2 sense [54]. The multivariate orthogonal polynomials $\psi_{\boldsymbol{\alpha}}(\mathbf{x})$ are indexed by the multi-index $\boldsymbol{\alpha} \in \mathbb{N}_0^m$ which denotes the degree of the polynomial with respect to each of the components of $\mathbf{x}$. Given the assumption $\mathbf{x} \sim \mathcal{U}([-1,1]^m)$, we can write these multivariate polynomials as

$$\psi_{\boldsymbol{\alpha}}(\mathbf{x}) \;=\; \prod_{i=1}^{m} \psi_{\alpha_i}(x_i), \tag{5.6}$$

where each ψ_{α_i} is the univariate Legendre polynomial of degree α_i. Without loss of generality, we also assume that the $\psi_{\boldsymbol{\alpha}}$ are normalized so that the coefficients in (5.5) are the inner product of f with the appropriate polynomial,

$$f_{\boldsymbol{\alpha}} \;=\; \int f(\mathbf{x})\, \psi_{\boldsymbol{\alpha}}(\mathbf{x})\, p(\mathbf{x})\, \mathrm{d}\mathbf{x}. \tag{5.7}$$

This method of approximation by orthogonal polynomials also appears in the uncertainty quantification literature under the name polynomial chaos [128, 56, 132].

In practice, we may choose to compute the pseudospectral expansion of $f(\mathbf{x})$ [25]. We do this by truncating (5.5) to include only polynomials of total degree d or less and approximating the integral in (5.7) numerically,

$$y \;=\; f(\mathbf{x}) \;\approx\; \sum_{|\boldsymbol{\alpha}|\leq d} \hat{f}_{\boldsymbol{\alpha}}\, \psi_{\boldsymbol{\alpha}}(\mathbf{x}), \quad \text{where} \quad f_{\boldsymbol{\alpha}} \;\approx\; \hat{f}_{\boldsymbol{\alpha}} \;=\; \sum_{j=0}^{M-1} \omega_j\, f(\boldsymbol{\xi}_j)\, \psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi}_j), \tag{5.8}$$

where $\boldsymbol{\xi}_j$, ω_j denote the nodes and weights of an M-point numerical integration rule—e.g., tensor product Gauss quadrature—with respect to p.

Pseudospectral polynomial approximations can serve as quick-to-evaluate surrogates for the original function. However, as the input dimension m grows, the cost of constructing (5.8) can quickly increase—a total degree d polynomial in m dimensions has $\binom{m+d}{d}$ coefficients. Recall that we assume $f(\mathbf{x})$ is (or can be well-approximated by) a one-dimensional ridge function. We can exploit this structure to enable pseudospectral polynomial approximation.

To construct an orthogonal polynomial expansion of $g(u)$ similar to (5.5), we must first understand the transformed input space. Recall that the full-dimensional input space is weighted with the input probability density function $p(\mathbf{x})$. The linear transform $\mathbf{a}^\top\mathbf{x}$ induces a new probability measure with density function $q(u)$. Figure 5.2 shows different rotations and projections defined by different vectors $\mathbf{a}$ of the three-dimensional cube $[-1,1]^3$ and the resulting one-dimensional probability densities $q(u)$.

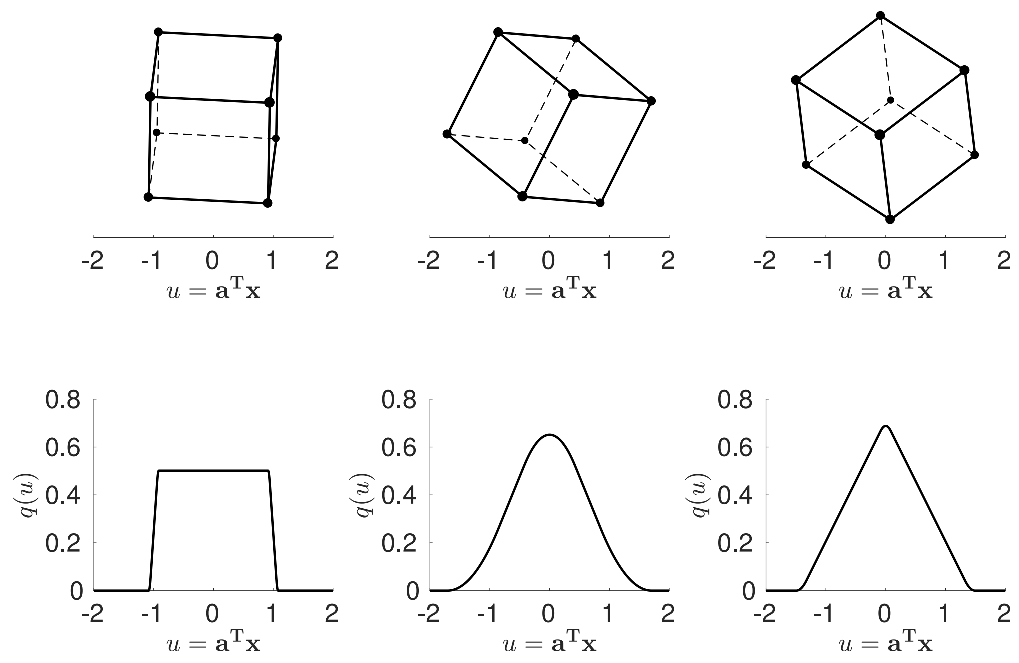


Figure 5.2: Different density functions $q(u)$ induced by different vectors $\mathbf{a} \in \mathbb{R}^3$

We address the computation $q(u)$ in Section 5.1.2. For now, assume $q(u)$ is known. We write the polynomial expansion of $g(u)$ as

$$y \;=\; g(u) \;=\; \sum_{i=0}^{\infty} g_i\,\phi_i(u), \quad \text{where} \quad g_i \;=\; \int g(u)\,\phi_i(u)\,q(u)\,du, \tag{5.9}$$

where the polynomials ϕ_i are orthonormal with respect to q. By truncating the expansion and numerically approximating the coefficients, we obtain the pseudospectral approximation of the ridge profile,

$$y \;=\; g(u) \;\approx\; \sum_{i=0}^{d} \hat{g}_i\,\phi_i(u), \quad \text{where} \quad \hat{g}_i \;=\; \sum_{j=0}^{M-1} \nu_j\,g(\lambda_j)\,\phi_i(\lambda_j), \tag{5.10}$$

where λ_j, ν_j define a numerical integration rule with respect to q.

Constructing the pseudospectral polynomial expansion of the one-dimensional ridge profile $g(u)$ significantly reduces the number of function evaluations required compared to working in the m-dimensional space. However, (5.10) requires knowledge of the orthonormal polynomials ϕ_i and an integration rule with respect to q. In Section 4.2, we showed how the Stieltjes algorithm can be used to construct a sequence of orthonormal polynomials with respect to arbitrary q. Furthermore, we showed how the Lanczos algorithm can be used a discrete approximation of Stieltjes. We employ these tools again here.

Let u_j, υ_j be the nodes and weights of an N-point numerical integration rule with respect to $q(u)$. These nodes and weights define a discrete approximation of $q(u)$, which we denote by $q^{(N)}(u)$. Performing Algorithm 4 on

$$\boldsymbol{A} = \begin{bmatrix} u_0 & & \\ & \ddots & \\ & & u_{N-1} \end{bmatrix}, \quad \tilde{\mathbf{v}}_0 = \begin{bmatrix} \sqrt{\upsilon_0} \\ \vdots \\ \sqrt{\upsilon_{N-1}} \end{bmatrix}, \tag{5.11}$$

is equivalent to performing Algorithm 3 on the discrete density function $q^{(N)}(u)$ [26]. The recurrence coefficients in $\boldsymbol{T}$ from (4.26) converge to those in $\boldsymbol{J}$ from (4.14) as N goes to infinity [54]. Therefore, we can use these recurrence coefficients to produce approximations to the orthonormal polynomials $\{\phi_0(u), \phi_1(u), \phi_2(u), \dots\}$. Additionally, the eigendecomposition of $\boldsymbol{T}$ provides us with an approximate Gauss-Christoffel quadrature rule with respect to $q(u)$.

5.1.2 Convolution of probability densities

To obtain a numerical integration rule with respect to $q(u)$, we must first be able to approximate it. This is done using convolution of probability densities [9, Chap. 4]. Consider two independent random variables x_1 and x_2 with density functions p_1 and p_2, respectively. The density function of $u = x_1 + x_2$ is

$$q(u) \;=\; (p_1 * p_2)(u) \;=\; \int p_1(t)\, p_2(u-t)\, dt. \tag{5.12}$$

Equation (5.12) is referred to as the convolution of p_1 and p_2.

Recall from (5.2) that we assume the input space is weighted by a uniform density over the hypercube, $\mathbf{x} \sim \mathcal{U}([-1,1]^m)$. By independence, we have that $p(\mathbf{x}) = p_1(x_1)\, p_2(x_2)\, \dots\, p_m(x_m)$ with each

$$p_i(x_i) \;=\; \begin{cases} \frac{1}{2} & \text{if } |x_i| \le 1, \\ 0 & \text{otherwise.} \end{cases} \tag{5.13}$$

We can write the linear transform $u = \mathbf{a}^\top \mathbf{x} = a_1 x_1 + \cdots + a_m x_m$ and recognize that $a_i x_i \sim \mathcal{U}([-a_i, a_i])$. Thus, we can obtain $q(u)$ by iteratively applying convolutions to each $a_i x_i$. In practice, we approximate the integral in (5.12) using a trapezoidal rule. However, this integral is one-dimensional and does not require us to evaluate the computational model—i.e., $f(\mathbf{x})$ from (5.1), so we can use a high density of points to approximate the convolution.

Algorithm 7 details the process of approximating $q(u)$ using iterative convolutions. Note that this algorithm uses the sign$\,(\cdot)$ function, which returns a vector of ± 1 based on the sign of each entry of the given vector. If $a_i = 0$ for some $i = 1, \ldots, m$, then the corresponding x_i has no influence on the model output. For our purposes, we may assume that $\text{sign}\,(0) = 1$. Also note that we define N equally-spaced points along the one-dimensional interval, where N is odd. This requirement is an artifact of the numerical approximation of (5.12).

5.2 Methodology

The pseudospectral expansion of the ridge profile from (5.10) requires us to evaluate $g(\lambda_j)$, but the ridge profile is unknown. Consider the linear transformation $\boldsymbol{\xi}_j = \lambda_j\, \mathbf{a}$. If $f(\mathbf{x})$ is an exact ridge function, then

$$f(\boldsymbol{\xi}_j) \;=\; f(\lambda_j\, \mathbf{a}) \;=\; g(\mathbf{a}^\top(\lambda_j\, \mathbf{a})) \;=\; g(\lambda_j\, \mathbf{a}^\top \mathbf{a}) \;=\; g(\lambda_j), \tag{5.20}$$

since $\mathbf{a}$ is assumed to be normalized. Thus, we can evaluate $f(\boldsymbol{\xi}_j)$ in place of $g(\lambda_j)$, provided that $\boldsymbol{\xi}_j \in [-1, 1]^m$. To ensure we find a valid point at which to evaluate f, consider the projection of the m-dimensional hypercube down to a one-dimensional domain via $u = \mathbf{a}^\top \mathbf{x}$. In general, the endpoints of this one-dimensional interval are defined by two opposing corners of the hypercube. The endpoints of the one-dimensional interval are

$$u_\ell \;=\; \mathbf{a}^\top \text{sign}\,(-\mathbf{a}) \quad \text{and} \quad u_r \;=\; \mathbf{a}^\top \text{sign}\,(\mathbf{a})\,, \tag{5.21}$$

and the corresponding corners of the hypercube are

$$\mathbf{x}_\ell \;=\; \text{sign}\,(-\mathbf{a}) \quad \text{and} \quad \mathbf{x}_r \;=\; \text{sign}\,(\mathbf{a})\,. \tag{5.22}$$

Algorithm 7 Discrete convolution of densities

```
Given: a vector $\mathbf{a} \in \mathbb{R}^m$
Assumptions: $\mathbf{x} \sim \mathcal{U}([-1,1]^m)$
  (1) Find the inputs of the one-dimensional interval
        $u_\ell = \mathbf{a}^\top \text{sign}(-\mathbf{a}), \quad u_r = \mathbf{a}^\top \text{sign}(\mathbf{a}),$   (5.14)
      and define $N$ (where $N$ is odd) equally-spaced points along the interval
        $u_j = u_\ell + j\,\Delta u, \quad j = 0,\dots,N-1,$   (5.15)
      where $\Delta u = (u_r - u_\ell)/(N-1)$.
  (2) Initialize the vector $\mathbf{q} = \begin{bmatrix} q(u_0) & \dots & q(u_{N-1}) \end{bmatrix}^\top$ where
        $q(u_j) = \begin{cases} 1/(2\,a_1) & \text{if } |u_j| \le a_1, \\ 0 & \text{otherwise.} \end{cases}$   (5.16)
  (3) for $i = 1,\dots,m-1$
      (i) Define $\mathbf{p} = \begin{bmatrix} p(u_0) & \dots & p(u_{N-1}) \end{bmatrix}^\top$ where
            $p(u_j) = \begin{cases} 1/(2\,a_{i+1}) & \text{if } |u_j| \le a_{i+1}, \\ 0 & \text{otherwise.} \end{cases}$   (5.17)
      (ii) For $j = 0,\dots,N-1$, define
            $k_0 = \max\left\{1, j - \frac{N-1}{2} + 1\right\},$
            $k_1 = \min\left\{\frac{N-1}{2} + j + 1, N\right\},$   (5.18)
          and compute
            $q_j = \sum_{k=k_0}^{k_1} q_k\, p_{\frac{N-1}{2} - j + k}$   (5.19)
NOTE: skip any $i$ for which $a_i = 0$
Output: $\mathbf{q} = \begin{bmatrix} q(u_0) & \dots & q(u_{N-1}) \end{bmatrix}^\top$
```

A line in m-dimensional space that connects these two corners of the hypercube is guaranteed to be contained within the hypercube. By projecting the one-dimensional quadrature points λ_j onto that line, we ensure that $\boldsymbol{\xi}_j \in [-1,1]^m$. We do this by

$$\boldsymbol{\xi}_j = (1 - \gamma_j)\,\mathbf{x}_\ell + \gamma_j\,\mathbf{x}_r, \tag{5.23}$$

where $\gamma_j = (\lambda_j - u_\ell)/(u_r - u_\ell)$. Figure 5.3 illustrates this process for a three-dimensional cube.

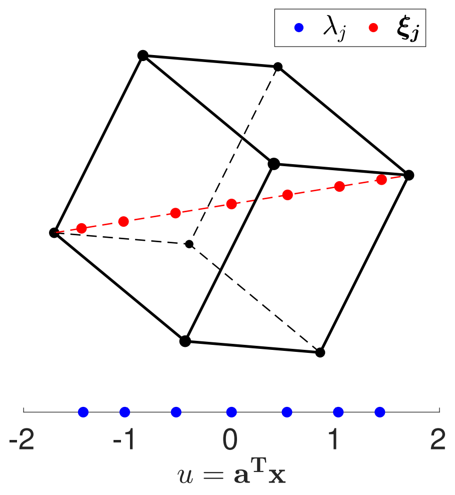

● λ_j ● $\boldsymbol{\xi}_j$
-2
-1
0
1
2
$u = \mathbf{a}^{\mathbf{T}}\mathbf{x}$

Figure 5.3: The projection of the one-dimensional quadrature nodes λ_j into m-dimensional space.

Algorithm 8 outlines the steps for building a pseudospectral expansion of an exact one-dimensional ridge function. To approximate the integral of (5.3), we recognize that

$$\hat{g}_0 \;=\; \sum_{j=0}^{d} \nu_j \, g(\lambda_j) \;\approx\; \int g(u)\, q(u)\, du \;=\; \int f(\mathbf{x})\, p(\mathbf{x})\, \mathrm{d}\mathbf{x}. \tag{5.24}$$

Algorithm 8 contains two levels of approximation: (i) the discrete approximation of $q(u)$ using an N-point trapezoidal rule and (ii) the number $d+1$ of Lanczos iterations performed. The number of Lanczos iterations corresponds to the number of terms in the polynomial expansion of g and the number of Gauss-Christoffel quadrature nodes. The latter is important as this is the number of model evaluations required to compute the pseudospectral coefficients. In general, we should choose $N \gg d$ since the approximation of q at the N trapezoidal points does not require any function evaluations, which are typically the most expensive step.

In the case where $f(\mathbf{x})$ is well-approximated by a ridge function (see (5.4)), the best L^2 approximation of f by g is the expected value of the output conditioned on $u = \mathbf{a}^\top\mathbf{x}$ [96, Chap. 8]. Thus, we want to compute $g(\lambda_j) = \mathbb{E}\left[y = f(\mathbf{x}) | \lambda_j = \mathbf{a}^\top\mathbf{x}\right]$. We write the sample approximation of this conditional expectation as

$$g(\lambda_j) \;\approx\; \hat{g}(\lambda_j) \;=\; \frac{1}{M_j} \sum_{i=0}^{M_j - 1} f(\boldsymbol{\xi}_{j,i}), \tag{5.30}$$

Algorithm 8 Lanczos-Stieljes one-dimensional ridge function approximation

Given: function $f : \mathbb{R}^m \to \mathbb{R}$ and unit vector $\mathbf{a} \in \mathbb{R}^m$ such that

$$y \;=\; f(\mathbf{x}) \;=\; g(\mathbf{a}^\top \mathbf{x})$$

for some unknown $g : \mathbb{R} \to \mathbb{R}$

Assumptions: $\mathbf{x} \sim \mathcal{U}([-1,1]^m)$ and f is square-integrable with respect to the input density $p(\mathbf{x})$

(1) Perform Algorithm 7 to obtain $\mathbf{q} = \begin{bmatrix} q(u_0) & \dots & q(u_{N-1}) \end{bmatrix}^\top$.

(2) Perform Algorithm 4 on

$$\boldsymbol{A} \;=\; \begin{bmatrix} u_0 & & \\ & \ddots & \\ & & u_{N-1} \end{bmatrix}, \quad \tilde{\mathbf{v}}_0 \;=\; \begin{bmatrix} \sqrt{q(u_0)} \\ \vdots \\ \sqrt{q(u_{N-1})} \end{bmatrix}, \tag{5.25}$$

to obtain the Jacobi matrix $\boldsymbol{T}$.

(3) Take the eigendecomposition of $\boldsymbol{T}$,

$$\boldsymbol{T} \;=\; \boldsymbol{Q}\,\boldsymbol{\Lambda}\,\boldsymbol{Q}^\top, \tag{5.26}$$

where the eigenvalues are $\{\lambda_0, \lambda_1, \dots, \lambda_d\}$ and the eigenvectors are normalized. Define $\nu_j = (\boldsymbol{Q})^2_{1,j+1}$ for $j = 0, \dots, d$.

(4) For $j = 0, \dots, d$, compute

$$\boldsymbol{\xi}_j \;=\; (1-\gamma_j)\,\mathbf{x}_\ell + \gamma_j\,\mathbf{x}_r, \quad \text{for} \quad \gamma_j = (\lambda_j - u_\ell)\,/\,(u_r - u_\ell), \tag{5.27}$$

where $\mathbf{x}_\ell = \operatorname{sign}(-\mathbf{a})$, $\mathbf{x}_r = \operatorname{sign}(\mathbf{a})$, $u_\ell = \mathbf{a}^\top \operatorname{sign}(-\mathbf{a})$, and $u_r = \mathbf{a}^\top \operatorname{sign}(\mathbf{a})$.

(5) Compute the pseudospectral coefficients

$$\hat{g}_i \;=\; \sum_{j=0}^{d} \nu_j\, f(\boldsymbol{\xi}_j)\, \phi_i(\lambda_j). \tag{5.28}$$

(6) Build the pseudospectral expansion

$$y \;=\; g(u) \;\approx\; \sum_{i=0}^{d} \hat{g}_i\, \phi_i(u). \tag{5.29}$$

where the M_j input values $\boldsymbol{\xi}_{j,i} \in [-1,1]^m$ are sampled uniformly conditioned on $\mathbf{a}^\top \boldsymbol{\xi}_{j,i} = \lambda_j$. To compute (5.30), we use a hit-and-run type of sampling algorithm. Given some $\boldsymbol{\xi}_{j,i}$, choose a random (unit) direction $\mathbf{w} \in \mathbb{R}^m$ that is orthogonal to the ridge direction $\mathbf{a}$. We then pick a step size $t \in [-\sqrt{m}, \sqrt{m}]$, where this range is used to ensure that the step size covers the maximum possible range of the rotated $[-1,1]^m$ hypercube. To obtain the $(i+1)$th conditional sample, we step from the previously drawn sample, $\boldsymbol{\xi}_{j,i+1} = \boldsymbol{\xi}_{j,i} + t\,\mathbf{w}$, provided that $\boldsymbol{\xi}_{j,i+1} \in [-1,1]^m$. If this is not the case, then we choose a new random step size until a valid conditional sample is obtained.

An issue with approximating a near-1D ridge function is that the sample approximation of the conditional expectation in (5.30) results in noisy estimates of $g(\lambda_j)$. For this reason, constructing an interpolating polynomial, as is constructed by Algorithm 8, may not be the best approach given a restricted computational budget. We recommend truncating the pseudospectral polynomial expansion further to avoid overfitting the approximation. For each Gauss-Christoffel quadrature node λ_j, we have $\boldsymbol{\xi}_{j,i}$ for $i = 0, \ldots, M_j - 1$. Estimate the standard error in each approximation of the conditional expectation, $s_j = \hat{\sigma}_j / \sqrt{M_j}$, where $\hat{\sigma}_j$ is the standard deviation of the $f(\boldsymbol{\xi}_{j,i})$, $i = 0, \ldots, M_j - 1$. Assuming that g can be expressed as a polynomial expansion, we expect a decay in the coefficients g_i from (5.9) for sufficiently large i. Recall that the pseudospectral coefficients, $\hat{g}_i$ from (5.10), approximate the true coefficients. We suggest truncating the expansion at a degree $\tilde{d} <= d$ polynomial, where $|\hat{g}_i| < \sum_{j=0}^{d} s_j/(d+1)$ for all $i > \tilde{d}$. This heuristic removes terms whose contribution to the expansion is smaller than the noise in the sample approximations $\hat{g}(\lambda_j)$. Algorithm 9 formalizes the process described here for building a pseudospectral approximation of a near-1D ridge function.

5.3 Numerical results

In this section, we numerically study the behavior of Algorithms 8 and 9 for approximating the ridge profile. We consider three problems here. The first is an exact one-dimensional ridge function and the other two are approximate one-dimensional ridge functions. The last problem is the physically-motivated Hartmann problem from Section 2.3.1.

Algorithm 9 Lanczos-Stieljes near-1D ridge function approximation

Given: function $f : \mathbb{R}^m \to \mathbb{R}$ and unit vector $\mathbf{a} \in \mathbb{R}^m$ such that

$$y \;=\; f(\mathbf{x}) \;\approx\; g(\mathbf{a}^\top \mathbf{x})$$

for some unknown $g : \mathbb{R} \to \mathbb{R}$

Assumptions: $\mathbf{x} \sim \mathcal{U}([-1,1]^m)$ and f is square-integrable with respect to the input density $p(\mathbf{x})$

```
(1) Perform Steps 1-3 of Algorithm 8 to obtain the Gauss-Christoffel quadrature nodes $\lambda_j$ and
    weights $\nu_j$ with respect to $q(u)$.

(2) For $j = 0, \dots, d$,
    (i) Compute

        $$\boldsymbol{\xi}_{j,0} \;=\; (1-\gamma_j)\,\mathbf{x}_\ell + \gamma_j\,\mathbf{x}_r, \quad \text{for} \quad \gamma_j = (\lambda_j - u_\ell)\,/\,(u_r - u_\ell), \qquad (5.31)$$

        where $\mathbf{x}_\ell = \text{sign}\,(-\mathbf{a})$, $\mathbf{x}_r = \text{sign}\,(\mathbf{a})$, $u_\ell = \mathbf{a}^\top\text{sign}\,(-\mathbf{a})$, and
        $u_r = \mathbf{a}^\top\text{sign}\,(\mathbf{a})$.
    (ii) Evaluate $f(\boldsymbol{\xi}_{j,0})$.
    (iii) For $i = 1, \dots, M_j - 1$,
        (a) Choose a random vector $\mathbf{w} \in \mathbb{R}^m$ with $\mathbf{w}^\top\mathbf{a} = 0$ and $||\mathbf{w}||_2 = 1$.
        (b) Choose a random value $t \in [-\sqrt{m}, \sqrt{m}]$.
        (c) If $||\boldsymbol{\xi}_{j,i} + t\,\mathbf{w}||_\infty <= 1$, then set $\boldsymbol{\xi}_{j,i+1} = \boldsymbol{\xi}_{j,i} + t\,\mathbf{w}$. Otherwise, repeat Step (b).
        (d) Evaluate $f(\boldsymbol{\xi}_{j,i+1})$.
    (iv) Approximate the conditional expectation at the quadrature point,

        $$g(\lambda_j) \;\approx\; \hat{g}(\lambda_j) \;=\; \frac{1}{M_j}\sum_{i=0}^{M_j-1} f(\boldsymbol{\xi}_{j,i}) \qquad (5.32)$$

        and the standard error $s_j \;=\; \hat{\sigma}_j/\sqrt{M_j}$, where $\hat{\sigma}_j$ is the standard deviation of the
        $f(\boldsymbol{\xi}_{j,i})$ for $i = 0, \dots, M_j - 1$.

(3) Compute the pseudospectral coefficients

    $$\hat{g}_i \;=\; \sum_{j=0}^{d} \nu_j\,\hat{g}(\lambda_j)\,\phi_i(\lambda_j). \qquad (5.33)$$

(4) Define $\tilde{d}$ to be the largest index such that $|\hat{g}_i| < \sum_{j=0}^{d} s_j/(d+1)$ for all $i > \tilde{d}$.

(5) Build the pseudospectral expansion

    $$y \;=\; g(u) \;\approx\; \sum_{i=0}^{\tilde{d}} \hat{g}_i\,\phi_i(u). \qquad (5.34)$$
```

5.3.1 Exact ridge function

In this section, we numerically study the behavior of Algorithm 8 for

$$y = \sin\left(2\,\pi\,(\mathbf{a}^\top \mathbf{x})\right) + \cos\left(\frac{\pi}{2}(\mathbf{a}^\top \mathbf{x})\right), \qquad \mathbf{x} \in \mathbb{R}^{25}. \tag{5.35}$$

We assume $\mathbf{x} \sim \mathcal{U}([-1,1]^{25})$. Notice that (5.35) is an exact one-dimensional ridge function.

Figure 5.4 contains the results from Algorithm 8 for $N = 10,000$ and $d = 50$. The first plot shows the one-dimensional ridge profile of (5.35) in black with the pseudospectral approximation from Algorithm 8 given by the blue dashed line. The inverse CDF method using Gauss-Legendre quadrature from [122] is shown in green for comparison. The absolute error of each method at each point is shown in the second plot, and the density $q(u)$ is in the third plot. Visually, both methods appear to perform reasonably well in the center of the domain, where the induced density function is relatively large. However, by examining the absolute errors of each method, we see that the Algorithm 8 significantly outperforms the pseudospectral approximation constructed using Gauss-Legendre quadrature. Near the endpoints, the errors in each approximation method begin to increase, but $q(u)$ in these regions is many orders of magnitude smaller than in the middle of the domain.

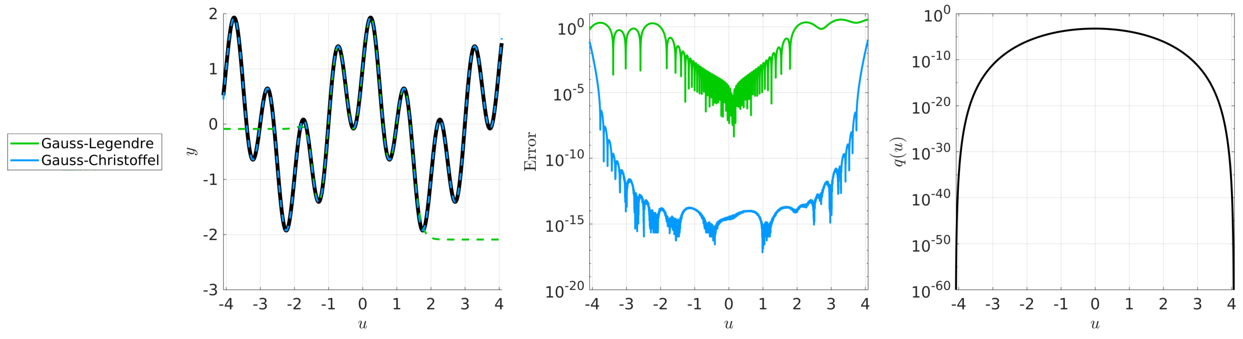


Figure 5.4: The results of Algorithm 8 applied to (5.35). The first plot shows the true ridge profile (in black) and the $d = 50$ Gauss-Christoffel (in blue) and Gauss-Legendre (in green) pseudospectral approximations. The second plot contains the absolute error of the approximations, and the third plot shows the density $q(u)$.

Next, we examine the two levels of approximation in Algorithm 8 Recall that these two levels are (i) the number N of trapezoid points used to construct the discrete approximation of $q(u)$ and

(ii) the number d of Lanczos iterations performed. The latter corresponds to the degree of the polynomial expansion as well as the number of Gauss-Christoffel quadrature nodes.

Figure 5.5 contains the results of this study. The first plot shows the approximated L^2 norm of the error between f and the pseudospectral expansions for varying values of N and d. The error appears to depend strongly on d. This is because a high-degree polynomial is required to fit the highly-oscillatory nature of $f(\mathbf{x})$. The second plot contains approximations of the integral of $f(\mathbf{x})$ using the first coefficient in the pseudospectral expansion. Here, we see a strong dependence on N. This is because integration errors in the Gauss-Christoffel decay quickly with d. To improve the approximation, we must improve our discrete approximation of the density $q(u)$.

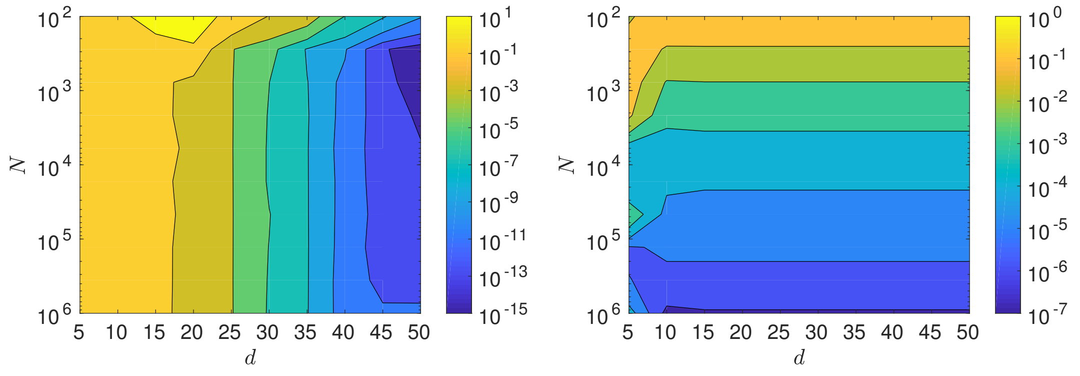


Figure 5.5: Results from studies of the two levels of approximation in Algorithm 8 applied to (5.35). The left plot contains L^2 errors in the polynomial approximation for various values of N and d. The right plot shows errors in the approximate integral of (5.35).

5.3.2 Approximate ridge function

In this section, we numerically study the behavior of Algorithm 9 for approximating nearly one-dimensional ridge functions. We consider the function

$$y = \sin\left(\frac{\pi}{5}(\mathbf{a}^\top \mathbf{x})\right) + \frac{1}{5}\cos\left(\frac{4\pi}{5}(\mathbf{a}^\top \mathbf{x})\right) + \frac{1}{40}\mathbf{x}^\top \boldsymbol{B}\,\mathbf{1} \qquad \mathbf{x} \in \mathbb{R}^{\mathbf{25}}, \tag{5.36}$$

where $\mathbf{a} \in \mathbb{R}^5$ defines the ridge-like structure, $\boldsymbol{B} \in \mathbb{R}^{5\times 24}$ contains an orthonormal basis for the subspace orthogonal to $\mathbf{a}$, and $\mathbf{1} \in \mathbb{R}^{\mathbf{24}}$ is a vector of ones. We assume $\mathbf{x} \sim \mathcal{U}([-1,1]^{25})$.

Figure 5.6 shows the results of using the extension of Algorithm 8 discussed in the previous section to (5.36). The plot on the left is a shadow plot of evaluations of (5.36) against $u = \mathbf{a}^\top \mathbf{x}$. The spread in the plot is due to variations in the 24 directions orthogonal to $\mathbf{a}$. The red line is the polynomial approximation of the ridge profile—$g(u) = \mathbb{E}\left[y = f(\mathbf{x}) | u = \mathbf{a}^\top \mathbf{x}\right]$. This is computed using $d = 11$ and $M = 50$ total functions evaluations distributed among the 12 Gauss-Christoffel quadrature nodes. The polynomial expansion is truncated at $\tilde{d} = 6$ to avoid overfitting the noise in the approximation of $g(\lambda_j)$. Note that fitting a polynomial of total degree 6 in 25 dimensions would require at least $M = \binom{25+6}{6} = 736,281$ to have a well-posed problem.

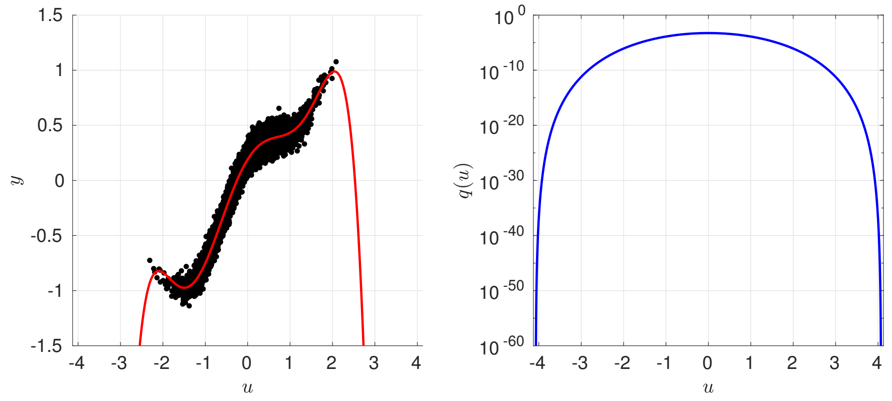


Figure 5.6: The results of the extension to Algorithm 8 applied to the approximate ridge function (5.36). The left plot contains a show plot of $f(\mathbf{x})$ along the $u = \mathbf{a}^\top \mathbf{x}$ axis with the $d = 11$ polynomial approximation on top of it. The right plot shows the density $q(u)$

5.3.3 Hartmann problem

In this section, we consider the physically-motivated Hartmann problem from magnetohydrodynamics. This problem is described in detail in Section 2.3.1, and we use the same problem setup here.

Figure 5.7 contains the results of applying Algorithm 9 to the induced magnetic field from (2.37). The top plot is a shadow plot of B_{ind} against $u = \mathbf{a}^\top \mathbf{x}$ overlaid with a polynomial approximation of the ridge profile. This approximation was constructed using $d = 4$ with a total computational budget of 100 function evaluations (20 for each of the 5 quadrature nodes). The bottom plots show the approximated relative L^2 errors in polynomial approximations constructed on the full five-dimensional inputs space (on the left) and the one-dimensional ridge subspace (on the right). On the full input space, we use uniformly-sampled points from the $[-1,1]^5$ hypercube and construct the least-squares polynomial approximation using Legendre polynomials with L^2 regularization. These approximations perform poorly when the computational budget is limited and restrict our optimal choice of polynomial degree. This issue grows exponentially as the dimension of the given function increases. The one-dimensional approximation constructed using Algorithm 9 achieves its optimal performance with very few function evaluations. For the restricted computational budget studied, the one-dimensional approximation outperforms its five-dimensional counterpart. Given a larger computational budget, we would expect the full polynomial approximation to perform better. This because the one-dimensional approximation is limited by the approximation of B_{ind} by a one-dimensional ridge function. Recall from Table 2.3 containing approximations of $\mathbb{E}\left[\text{Var}\left[f(\mathbf{x})\,\middle|\mathbf{u}=\boldsymbol{A}^\top\mathbf{x}\right]\right]$, which we use as a global measure of the ridge structure in $f(\mathbf{x})$. From Table 2.3, the expected conditional variance in the one-dimensional ridge approximation of B_{ind} is approximately 5.50×10^{-2}. This corresponds to the leveling off in the accuracy of the one-dimensional polynomial approximation. Figure 5.8 contains the results for the same study applied to the average flow velocity u_{avg} from (2.36). Note that the expected conditional variance for u_{avg} from Table 2.3 is 1.29×10^{-2}.

5.4 Summary

In this chapter, we introduce a novel algorithm for constructing polynomial approximations of one-dimensional ridge functions using the Lanczos iterative method. In general, building a polynomial surrogate of an m-dimensional function suffers from the curse of dimensionality—an exponential increase in computational costs associated with increases in m. We also introduce an approach to extending this algorithm to functions that are well-approximated by a one-dimensional ridge function.

We numerically study the new algorithm on several test problems, including exact and approximate ridge functions. We show that exploiting low-dimensional structure can result in exponential

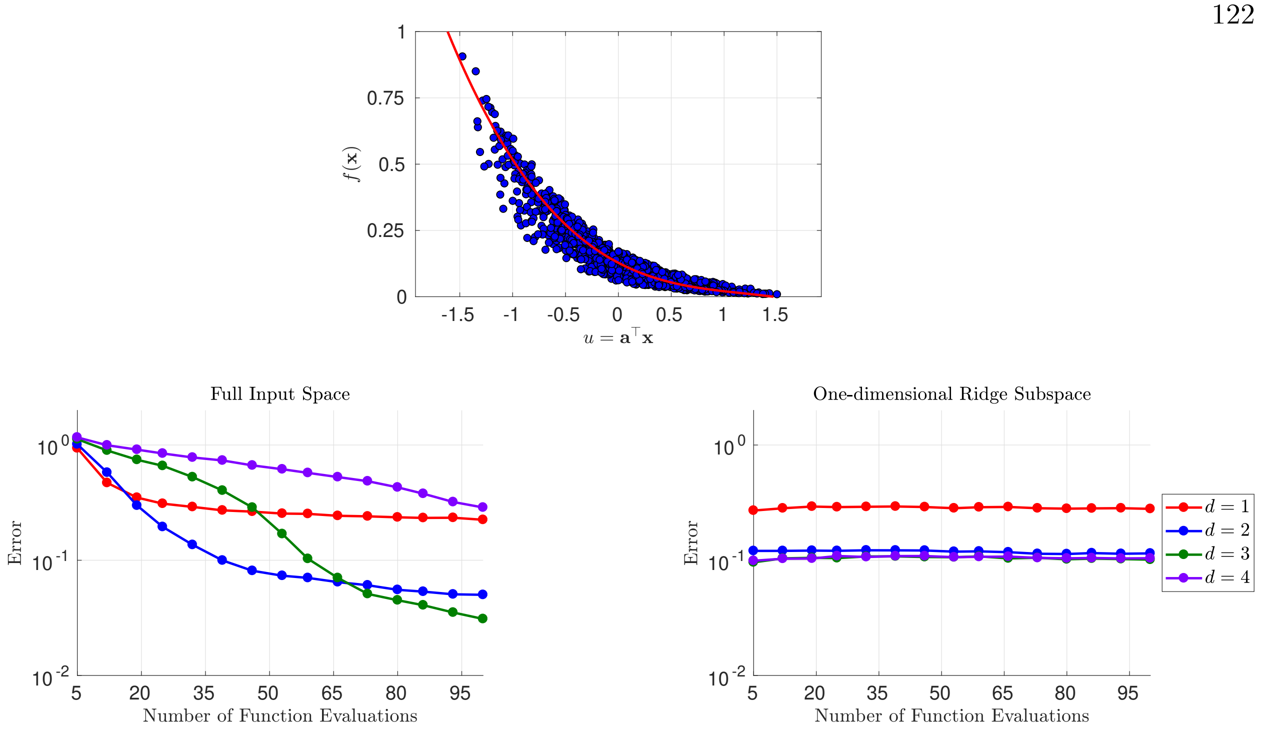


Figure 5.7: The results of applying the extension of Algorithm 8 to induced magnetic field. The top plot is a shadow plot of B_{ind} against $u = \mathbf{a}^\top \mathbf{x}$. The bottom plots show L^2 errors of polynomial approximations constructed on the full five-dimensional input space (on the left) and the one-dimensional ridge subspace (on the right).

savings while maintaining accuracy. Additionally, we study the two-level approximation behavior of the algorithm. The first level is a discrete approximation of the induced density function $q(u)$. The second level is the number Lanczos iterations performed. This corresponds to the degree of the polynomial approximation of the ridge function as well as the number of Gauss-Christoffel quadrature points. In studying the extension of the algorithm to nearly one-dimensional ridge functions, we show that we can quickly achieve the baseline error using very few function evaluations.

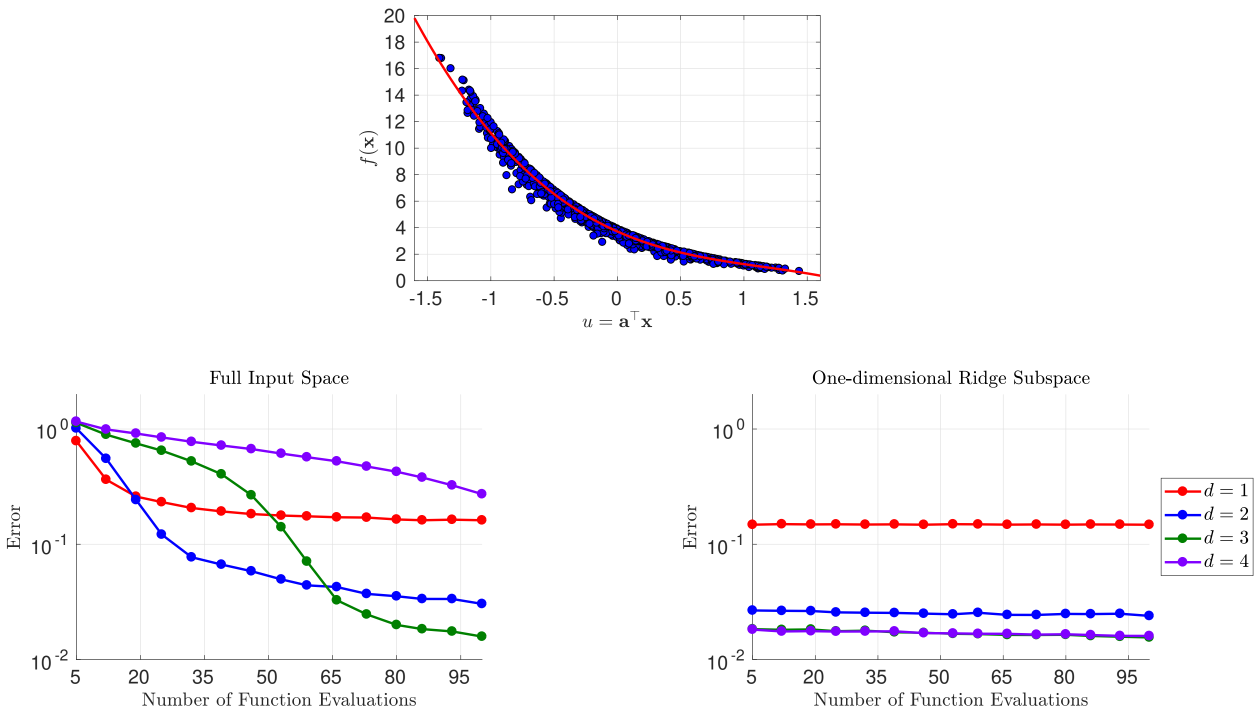


Figure 5.8: The results of applying the extension of Algorithm 8 to average flow velocity. The top plot is a shadow plot of u_{avg} against $u = \mathbf{a}^\top \mathbf{x}$. The bottom plots show L^2 errors of polynomial approximations constructed on the full five-dimensional input space (on the left) and the one-dimensional ridge subspace (on the right).

Chapter 6

Conclusion

In this thesis, we examine several research questions related to the use of ridge functions for dimension reduction. In general, approximation of high-dimensional functions suffers from the curse of dimensionality—i.e., an exponential increase in computational costs resulting from increasing input dimension. Supervised linear dimension reduction looks for directions within the input space that are relatively more (or less) important in terms of determining the function output. This idea is formalized by the concept of ridge functions, which reformulate the original m-dimensional function into a function that depends on $n < m$ linear combinations of the original inputs. The work presented here considers the interpretation, discovery, and exploitation of this type of low-dimensional ridge structure.

In Chapter 2, we relate dimension reduction via ridge functions to classical dimensional analysis. Dimensional analysis seeks to reduce the number of parameters in a physically-motivated system by examining the units of the various quantities. The Buckingham Pi theorem describes how unitless parameters are constructed via specific products of powers of the original parameters. The number of unitless parameters is bounded above the number of original parameters. In physically-motivated models, the linear combinations defined by the ridge directions relate to these products and powers of the dimensional quantities through a log transform. This provides some insight into why ridge structure seems to be prevalent in computational science models. We use magnetohydrodynamics (MHD) as the physical framework for the work in this chapter. In particular, we studied the Hartmann problem (a standard problem in MHD with closed form solutions) and a computational model from an MHD generator.

In addition to clarifying the interpretation, this work also suggests a method for inducing ridge structure in your model. By considering a log-transformed set of inputs, we are guaranteed a ridge function whose dimension is bounded above by the number of unitless quantities resulting from Buckingham Pi analysis. We also provide numerical evidence that this log transformation may work as a heuristic for improving performance in ridge approximation.

In Chapter 3, we review a theoretical framework for discovering low-dimensional structure in regression problems—sufficient dimension reduction (SDR). Methods for SDR, such as sliced inverse regression (SIR) and sliced average variance estimation (SAVE), compute a basis for the central subspace. This is done using a slice-based approximation of the inverse regression.

We show that, in the context of deterministic functions, the fundamental requirements of SDR are equal to those of ridge functions. In this way, this work connects these two previously disjoint fields of dimension reduction research. We explore SIR and SAVE as algorithms for ridge recovery. These algorithms are significantly cheaper than other gradient- or optimization-based algorithms for ridge recovery. We also examine situations where these methods may falsely indicate ridge structure that is not present in the function. Lastly, we perform rigorous convergence analysis of these algorithms.

Chapter 4 introduces novel algorithms for computing the underlying population matrices of SIR and SAVE that outperform the slice-based approaches studied in Chapter 3. The new algorithms replace the slicing of the output space by an orthonormal polynomial expansion and Gauss-Christoffel quadrature. They employ classical algorithms from numerical analysis in a novel way to accurately approximate the inverse conditional moments underlying the SIR and SAVE methods. We call these algorithms Lanczos-Stieltjes inverse regression (LSIR) and Lanczos-Stieltjes average variance estimation (LSAVE).

Numerical studies of the new LSIR and LSAVE algorithms show significant improvement over the slice-based approach. We show that the slicing can be interpreted as a Riemann sum approximation that acts as an accuracy bottleneck. After a point, increased sampling of the input space will not improve the accuracy of the methods. The new algorithms remove this bottleneck and enable the use of higher-order numerical integration techniques. In the case of Monte Carlo sampling (the only sampling that can be performed with the slice-based algorithms), the Lanczos-Stieltjes methods perform at least as well as the best case of the SIR and SAVE algorithms.

Lastly, in Chapter 5, we introduce a new approach to building a polynomial approximation of the ridge profile for one-dimensional ridge functions. The linear transformation defining a one-dimensional ridge function induces a probability measure on the ridge subspace. In general, this induced measure is arbitrary. We use discrete convolutions to approximate this measure and employ several of the same Lanczos-Stieltjes techniques from Chapter 4 to build orthonormal polynomials and high-order Gauss-Christoffel quadrature rules with respect to this measure. This methodology is extended to approximate ridge functions by approximating the conditional expectation. This is done using a hit-and run algorithm. Numerical studies of both exact and approximate one-dimensional ridge functions show that we can construct accurate polynomial approximations of the ridge profile with relatively few function evaluations

Bibliography

[1] S. Abbott, Understanding Analysis, Springer, New York, 2nd ed., 2001.

[2] K. P. Adragni and R. D. Cook, Sufficient dimension reduction and prediction in regression, Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 367 (2009), pp. 4385–4405.

[3] M. Ajtai, K. Janos, and E. Szemeredi, An $\mathcal{O}(n \log n)$ sorting network, in Fifteenth Annual ACM Symposium on Theory of Computing, 1983, pp. 1–9.

[4] M. C. Albrecht, C. J. Nachtsheim, T. A. Albrecht, and R. D. Cook, Experimental design for engineering dimensional analysis, Technometrics, 55 (2013), pp. 257–270.

[5] D. Allaire and K. Willcox, Surrogate modeling for uncertainty assessment with application to aviation environmental system models, AIAA Journal, 48 (2010), pp. 1791–1803.

[6] G. I. Barenblatt, Scaling, Self-similarity, and Intermediate Asymptotics, Cambridge University Press, Cambridge, 1996.

[7] R. E. Bellman, Adaptive Control Processes: A Guided Tour, Princeton University Press, 1961.

[8] K. Beyer, J. Goldstein, R. Ramakrishnan, and S. U, When is "nearest neighbor" meaningful?, in International Conference on Database Theory, 1999, pp. 217–235.

[9] P. Billingsley, Probability and Measure, Wiley, New York, 1986.

[10] I. Borg and P. J. F. Groenen, Modern Multidimensional Scaling, Theory and Applications, Springer, New York, 2005.

[11] D. Calvetti and E. Somersalo, Computational Mathematical Modeling: An Integrated Approach Across Scales, SIAM, Philadelphia, 2013.

[12] P. Challenor, Using emulators to estimate uncertainty in complex models, in Uncertainty Quantification in Scientific Computing, A. M. Dienstfrey and R. F. Boisvert, eds., Berlin, Heidelberg, 2012, Springer Berlin Heidelberg, pp. 151–164.

[13] J. T. Chang and D. Pollard, Conditioning as disintegration, Statistica Neerlandica, 51 (1997), pp. 287–317.

[14] C. W. Clenshaw and A. R. Curtis, A method for numerical integration on an automatic computer, Numerische Mathematik, 2 (1960), pp. 197–205.

[15] A. Cohen, I. Daubechies, R. DeVore, G. Kerkyacharian, and D. Picard, Capturing ridge functions in high dimensions from point queries, Constructive Approximation, 35 (2012), pp. 225–243.

[16] P. Constantine, M. Emory, J. Larsson, and G. Iaccarino, Exploiting active subspaces to quantify uncertainty in the numerical simulation of the HyShot II scramjet, Journal of Computational Physics, 302 (2015), pp. 1–20.

[17] P. Constantine and D. Gleich, Computing active subspaces with Monte Carlo, arXiv:1408.0545v2, (2015).

[18] P. G. Constantine, Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies, SIAM, Philadelphia, 2015.

[19] P. G. Constantine, Project title. `https://github.com/paulcon/pypbvp`, 2018.

[20] P. G. Constantine, Z. del Rosario, and G. Iaccarino, Many physical laws are ridge functions, arXiv:1605.07974, (2016).

[21] P. G. Constantine and A. Doostan, Time-dependent global sensitivity analysis with active subspaces for a lithium ion battery model, Statistical Analysis and Data Mining, 10 (2017), pp. 243–262.

[22] P. G. Constantine, E. Dow, and Q. Wang, Active subspace methods in theory and practice: Applications to kriging surfaces, SIAM Journal on Scientific Computing, 36 (2014), pp. A1500–A1524.

[23] P. G. Constantine, A. Eftekhari, J. Hokanson, and R. Ward, A near-stationary subspace for ridge approximation, Computer Methods in Applied Mechanics and Engineering, 326 (2017), pp. 402–421.

[24] P. G. Constantine, A. Eftekhari, and M. B. Wakin, Computing active subspaces efficiently with gradient sketching, in 2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2015, pp. 353–356.

[25] P. G. Constantine, M. S. Eldred, and E. T. Phipps, Sparse pseudospectral approximation method, Computer Methods in Applied Mechanics and Engineering, 229 (2012), pp. 1–12.

[26] P. G. Constantine and E. T. Phipps, A Lanczos method for approximating composite functions, Applied Mathematics and Computation, 218 (2012), pp. 11751–11762.

[27] P. G. Constantine, B. Zaharatos, and M. Campanelli, Discovering an active subspace in a single-diode solar cell model, Statistical Analysis and Data Mining: The ASA Data Science Journal, 8 (2015), pp. 264–273.

[28] R. D. Cook, On the interpretation of regression plots, Journal of the American Statistical Association, 89 (1994), pp. 177–189.

[29] ——, Graphics for regressions with a binary response, Journal of the American Statistical Association, 91 (1996), pp. 983–992.

[30] R. D. Cook, Regression Graphics: Ideas for Studying Regression through Graphics, John Wiley & Sons, Inc, New York, 1998.

[31] R. D. Cook, SAVE: A method for dimension reduction and graphics in regression, Communications in Statistics - Theory and Methods, 29 (2000), pp. 2109–2121.

[32] R. D. Cook and L. Forzani, Likelihood-based sufficient dimension reduction, Journal of the American Statistical Association, 104 (2009), pp. 197–208.

[33] R. D. Cook and S. Weisberg, Sliced inverse regression for dimension reduction: comment, Journal of the American Statistical Association, 86 (1991), pp. 328–332.

[34] A. Cortesi, P. G. Constantine, T. Magin, and P. M. Congedo, Forward and backward uncertainty quantification with active subspaces: application to hypersonic flows around a cylinder, PhD thesis, INRIA Bordeaux, équipe CARDAMOM, 2017.

[35] R. Courant and D. Hilbert, Methods of Mathematical Physics, Vol. II, Interscience, New York, 1962.

[36] T. Cowling and R. B. Lindsay, Magnetohydrodynamics, Physics Today, 10 (1957), p. 40.

[37] J. P. Cunningham and Z. Ghahramani, Linear dimensionality reduction: Survey, insights, and generalizations, Journal of Machine Learning Research, 16 (2015), pp. 2859–2900.

[38] P. A. Davidson, An Introduction to Magnetohydrodynamics, Cambridge University Press, Cambridge, 2001.

[39] P. J. Davis and P. Rabinowitz, Methods of Numerical Integration, Academic Press, San Diego, 1984.

[40] C. de Boor and G. H. Golub, The numerically stable reconstruction of a Jacobi matrix from spectral data, Linear Algebra and its Applications, 21 (1978), pp. 245–260.

[41] P. Diaz and P. G. Constantine, Global sensitivity metrics from active subspaces, arXiv:1510.04361v1, (2015).

[42] P. Domingos, A few useful things to know about machine learning, Communications of the ACM, 55 (2012), pp. 78–87.

[43] D. L. Donoho, High-dimensional data analysis: The curses and blessings of dimensionality, in AMS Conference on Math Challenges of the 21st Century, 2000.

[44] M. L. Eaton, A characterization of spherical distributions, Journal of Multivariate Analysis, 20 (1986), pp. 272–27.

[45] B. Efron and R. J. Tibshirani, An Introduction to the Bootstrap, CRC press, 1994.

[46] M. Eldred, C. Webster, and P. Constantine, Evaluation of non-intrusive approaches for wiener-askey generalized polynomial chaos, in 49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 2008, p. 1892.

[47] G. Fasshauer and M. McCourt, Kernel-based Approximation Methods using MATLAB, World Scientific Publishing Company, Singapore, 2016.

[48] G. B. Folland, Real Analysis: Modern Techniques and Their Applications, John Wiley & Sons Ltd, New York, 2nd ed., 1999.

[49] M. Fornasier, K. Schnass, and J. Vybiral, Learning functions of few arbitrary linear parameters in high dimensions, Foundations of Computational Mathematics, 12 (2012), pp. 229–262.

[50] G. E. Forsythe, Generation and use of orthogonal polynomials for data-fitting with a digital computer, Journal of the Society for Industrial and Applied Mathematics, 5 (1957), pp. 74–88.

[51] J. H. Friedman and W. Stuetzle, Projection pursuit regression, Journal of the American Statistical Association, 76 (1980), pp. 817–823.

[52] K. Fukumizu and C. Leng, Gradient-based kernel dimension reduction for regression, Journal of the American Statistical Association, 109 (2014), pp. 359–370.

[53] W. Gautschi, The interplay between classical analysis and (numerical) linear algebra—a tribute to Gene H. Golub, Electronic Transactions on Numerical Analysis, 13 (2002), pp. 119–147.

[54] ——, Orthogonal Polynomials, Oxford Press, Oxford, 2004.

[55] R. Ghanem, D. Higdon, and H. Owhadi, Handbook of Uncertainty Quantification, Springer International Publishing, 2016.

[56] R. Ghanem and P. Spanos, Stochastic Finite Elements: A Spectral Approach, Springer-Verlag, New York, 1991.

[57] J. M. Gilbert, J. L. Jefferson, P. G. Constantine, and R. M. Maxwell, Global spatial sensitivity of runoff to subsurface permeability using the active subspace method, Advances in water resources, 92 (2016), pp. 30–42.

[58] A. Gittens and J. A. Tropp, Tail bounds for all eigenvalues of a sum of random matrices, arXiv:1104.4513, (2011).

[59] A. Glaws and P. G. Constantine, Gauss–Christoffel quadrature for inverse regression: applications to computer experiments, Statistics and Computing, (2018).

[60] ——, A Lanczos-Stieltjes method for one-dimensional ridge function approximation and integration, arXiv:1808.02095, (2018).

[61] A. Glaws, P. G. Constantine, and R. D. Cook, Inverse regression for ridge recovery: A data-driven approach for parameter space dimension reduction in computational science, arXiv:1702.02227, (2017).

[62] A. Glaws, P. G. Constantine, J. Shadid, and T. M. Wildey, Dimension reduction in magnetohydrodynamics power generation models: dimensional analysis and active subspaces, Statistical Analysis and Data Mining, 10 (2017), pp. 312–325.

[63] G. H. Golub and G. Meurant, Matrices, Moments, and Quadrature with Applications, Princeton University, Princeton, 2010.

[64] G. H. Golub and C. F. Van Loan, Matrix Computations, JHU Press, Baltimore, 2nd ed., 1996.

[65] G. H. Golub and J. H. Welsch, Calculation of Gauss quadrature rules, Mathematics of Computation, 23 (1969), pp. 221–230.

[66] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, Cambridge, 2016.

[67] R. B. Gramacy and H. Lian, Gaussian process single-index models as emulators for computer experiments, Technometrics, 54 (2012), pp. 30–41.

[68] W. K. Härdle, M. Müller, S. Sperlich, and A. Werwatz, Nonparametric and Semiparametric Models, Springer-Verlag, Berlin, 2004.

[69] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer, New York, 2nd ed., 2009.

[70] J. M. Hokanson and P. G. Constantine, Data-driven polynomial ridge approximation using variable projection, SIAM Journal on Scientific Computing, 40 (2017), pp. A1566–A1589.

[71] J. L. Jefferson, J. M. Gilbert, P. G. Constantine, and R. M. Maxwell, Active subspaces for sensitivity analysis and dimension reduction of an integrated hydrologic model, Computers & Geosciences, 83 (2015), pp. 127–138.

[72] F. John, Plane Waves and Spherical Means Applied to Partial Differential Equations, Interscience, New York, 1955.

[73] M. E. Johnson, Multivariate Statistical Simulation, Wiley, Hoboken, 1987.

[74] I. T. Jolliffe, Principal Component Analysis, Springer Science+Business Media, LLC, 1986.

[75] D. R. Jones, A taxonomy of global optimization methods based on response surfaces, Journal of Global Optimization, 21 (2001), pp. 345–383.

[76] J. R. Koehler and A. B. Owen, Computer experiments, Handbook of Statistics, 13 (1996), pp. 261–308.

[77] C. Lanczos, An iteration method for the solution of the eigenvalue problem of linear differential and integral operators, Journal of Research of the National Bureau of Standards, 45 (1950), pp. 255–282.

[78] J. A. Lee and M. Verleysen, Nonlinear Dimensionality Reduction, Springer Science+Business Media, LLC, New York, 2007.

[79] B. Li, Sufficient Dimension Reduction: Methods and Applications with R, CRC Press, Philadelphia, 2018.

[80] K. C. Li, Sliced inverse regression for dimension reduction, Journal of the American Statistical Association, 86 (1991), pp. 316–327.

[81] ——, On principal hessian directions for data visualization and dimension reduction: Another application of stein's lemma, Journal of the American Statistical Association, 87 (1992), pp. 1025–1039.

[82] K.-C. Li and N. Duan, Regression analysis under link violation, The Annals of Statistics, 17 (1989), pp. 1009–1052.

[83] J. Liesen and Z. Strakoš, Krylov Subspace Methods: Principles and Analysis, Oxford Press, Oxford, 2013.

[84] D. K. J. Lin and W. Shen, Comment: Some statistical concerns on dimensional analysis, Technometrics, 55 (2013), pp. 281–285.

[85] P. T. Lin, J. N. Shadid, R. S. Tuminaro, M. Sala, G. L. Hennigan, and R. P. Pawlowski, A parallel fully coupled algebraic multilevel preconditioner applied to multiphysics PDE applications: Drift-diffusion, flow/transport/reaction, resistive MHD, International Journal for Numerical Methods in Fluids, 64 (2010), pp. 1148–1179.

[86] B. F. Logan and L. A. Shepp, Optimal reconstruction of a function from its projections, Duke Mathematical Journal, 42 (1975), pp. 645–659.

[87] G. Meurant, The Lanczos and Conjugate Gradient Algorithms: From Theory to Finite Precision Computations, SIAM, Philadelphia, 2006.

[88] M. D. Morris, Factorial sampling plans for preliminary computational experiments, Technometrics, 33 (1991), pp. 161–174.

[89] S. Mukherjee, Q. Wu, and D.-X. Zhou, Learning gradients on manifolds, Bernoulli, 16 (2010), pp. 181–207.

[90] R. H. Myers and D. C. Montgomery, Response Surface Methodology: Process and Product Optimization Using Designed Experiments, John Wiley & Sons, New York, 1995.

[91] National Institute of Standards and Technology, International system of units (si). `http://physics.nist.gov/cuu/Units/units.html`, 2016. Accessed: 2016-06-11.

[92] NETL, Magnetohydrodynamic power generation workshop summary report, tech. rep., National Energy Technology Laboratory, 2014.

[93] C. Othmer, T. Lukaczyk, P. G. Constantine, and J. J. Alonso, On active subspaces in car aerodynamics, American Institute of Aeronautics and Astronautics, (2016).

[94] A. B. Owen, Monte Carlo Theory, Methods and Examples, 2013.

[95] R. Pawlowski, J. Shadid, T. Smith, E. Cyr, and P. Weber, Drekar::cfd-a turbulent fluid-flow and conjugate heat transfer code: Theory manual version 1.0, Tech. Rep. SAND2012-2697, Sandia National Laboratories, 2012.

[96] A. Pinkus, Ridge Functions, Cambridge University Press, 2015.

[97] M. M. Rao and R. J. Swift, Probability Theory with Applications, Springer, New York, 2006.

[98] S. Razavi, B. A. Tolson, and D. H. Burn, Review of surrogate modeling in water resources, Water Resources Research, 48 (2012), p. W07401.

[99] R. H. Sabersky, A. J. Acosta, E. G. Hauptmann, and E. M. Gates, Fluid Flow: A First Course in Fluid Mechanics, Prentice Hall, New Jersey, 4th ed., 1999.

[100] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, Design and analysis of computer experiments, Statistical Science, 4 (1989), pp. 409–423.

[101] A. Saltelli, M. Ratto, T. Andres, J. C. Francesca Campolongo, D. Galtelli, M. Saisana, and S. Tarantola, Global Sensitivity Analysis. The Primer, John Wiley & Sons Ltd, England, 2008.

[102] T. J. Santner, B. J. Williams, and W. I. Notz, The Design and Analysis of Computer Experiments, Springer Science+Businuess Media New York, 2003.

[103] D. W. Scott, Multivariate Density Estimation: Theory, Practice, and Visualization, John Wiley & Sons, New York, 2015.

[104] J. Shadid, R. Pawlowski, J. Banks, L. Chacn, P. Lin, and R. Tuminaro, Towards a scalable fully-implicit fully-coupled resistive MHD formulation with stabilized FE methods, Journal of Computational Physics, 229 (2010), pp. 7649–7671.

[105] J. Shadid, R. Pawlowski, E. Cyr, R. Tuminaro, L. Chacón, and P. Weber, Scalable implicit incompressible resistive MHD with stabilized FE and fully-coupled Newton-Krylov-AMG, Computer Methods in Applied Mechanics and Engineering, 304 (2016), pp. 1–25.

[106] J. Shadid, T. Smith, E. Cyr, T. Wildey, and R. Pawlowski, Stabilized FE simulation of prototype thermal-hydraulics problems with integrated adjoint-based capabilities, Journal of Computational Physics, (2016). In press.

[107] W. Shen, Dimensional Analysis in Statistics: Theories, Methodologies and Applications, PhD thesis, The Pennsylvania State University, 2015.

[108] W. Shen, T. Davis, D. K. J. Lin, and C. J. Nachtsheim, Dimensional analysis and its applications in statistics, Journal of Quality Technology, 46 (2014), pp. 185–198.

[109] R. C. Smith, Uncertainty Quantification: Theory, Implementation, and Applications, SIAM, Philadelphia, 2013.

[110] R. L. Smith, Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions, Operations Research, 32 (1984), pp. 1296–1308.

[111] I. M. Sobol, Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates, Mathematics and computers in simulation, 55 (2001), pp. 271–280.

[112] D. Sondak, J. Shadid, A. Oberai, R. Pawlowski, E. Cyr, and T. Smith, A new class of finite element variational multiscale turbulence models for incompressible magnetohydrodynamics, Journal of Computational Physics, 295 (2015), pp. 596–616.

[113] G. W. Stewart, Error and perturbation bounds for subspaces associated with certain eigenvalue problems, SIAM Review, 15 (1973), pp. 727–764.

[114] G. W. Stewart, Stochastic perturbation theory, SIAM Review, 4 (1990), pp. 579–610.

[115] T. J. Stieltjes, Quelques recherches sur la thorie des quadratures dites mcaniques, Annales scientifiques de l'cole Normale Suprieure, 1 (1884), pp. 409–426.

[116] H. Strange and R. Zwiggelaar, Open Problems in Spectral Dimensionality Reduction, Springer, 2014.

[117] T. Sullivan, Introduction to Uncertainty Quantification, Springer, New York, 2015.

[118] R. Tipireddy and R. Ghanem, Basis adaptation in homogeneous chaos spaces, Journal of Computational Physics, 259 (2014), pp. 304–317.

[119] J. F. Traub and A. G. Werschulz, Complexity and Information, Cambridge University Press, Cambridge, 1998.

[120] L. N. Trefethen, Approximation Theory and Approximation Practice, SIAM, Philadelphia, 2013.

[121] J. A. Tropp, User-friendly tail bounds for sums of random matrices, Foundations of Computational Mathematics, 12 (2012), pp. 389–434.

[122] P. A. Tsilifis, Gradient-informed basis adaptation for Legendre chaos expansions, Journal of Verification, Validation, and Uncertainty Quantification, 3 (2018), p. 011005.

[123] H. Tyagi and V. Cevher, Learning non-parametric basis independent models from point queries via low-rank methods, Applied and Computational Harmonic Analysis, 37 (2014), pp. 389–412.

[124] USGAO, FOSSIL FUELS: The Department of Energy's Magnetohydrodynamics Development Program, Tech. Rep. RCED-93-174, U.S. Government Accountability Office, 1993.

[125] G. Wahba, Spline Models for Observational Data, SIAM, Philadelphia, 1990.

[126] G. G. Wang and S. Shan, Review of metamodeling techniques in support of engineering design optimization, Journal of Mechanical Design, 129 (2006), pp. 370–380.

[127] R. L. Wheeden and A. Zygmund, Measure and Integral: An Introduction to Real Analysis, CRC, Boca Raton, 1977.

[128] N. Wiener, The homogeneous chaos, American Journal of Mathematics, 60 (1938), pp. 897–936.

[129] H. Wozniakowski, Tractability and strong tractability of linear multivariate problems, Journal of Complexity, 10 (1994), pp. 96–128.

[130] Q. Wu, J. Guinney, M. Maggioni, and S. Mukherjee, Learning gradients: Predictive models that infer geometry and statistical dependence, Journal of Machine Learning Research, 11 (2010), pp. 2175–2198.

[131] D. Xiu and J. S. Hesthaven, High-order collocation methods for differential equations with random inputs, SIAM Journal on Scientific Computing, 27 (2005), pp. 1118–1139.

[132] D. Xiu and G. E. Karniadakis, The Wiener–Askey polynomial chaos for stochastic differential equations, SIAM Journal on Scientific Computing, 24 (2002), pp. 619–644.

[133] X. Yin, B. Li, and R. D. Cook, Successive direction extraction for estimating the central subspace in a multiple-index regression, Journal of Multivariate Analysis, 99 (2008), pp. 1733–1757.

Appendix A

Proofs

In what follows, we prove the various theorems presented in Section 3.2. We begin by introducing a lemma that will simplify this task. This lemma enables quick analysis of the asymptotic behavior of multiple summations over complicated multi-index sets. Define the multi-index set

$$\mathcal{K}^{p,q}(r) = \left\{ \mathbf{k} \in \mathbb{N}^{p+q} \;\middle|\; \begin{array}{c} k_1, \dots k_p \in \{1, \dots, N\} \\ \text{and } k_{p+1}, \dots, k_{p+q} \in \{1, \dots, N_r\} \end{array} \right\} \tag{A.1}$$

where $N_r \leq N$. Let $\mathbf{b} \in \mathbb{R}^{p+q}$ be a random vector and let $\left\{\mathbf{b}^k\right\}$, $k = 1, \dots, N$, denote N independent realizations of $\mathbf{b}$ where N is from (A.1). Define the $(p+q)$-dimensional tensor $\boldsymbol{B}^{p,q}(r)$ whose elements are given by

$$B^{p,q}_{\mathbf{k}}(r) = \mathbb{E}\left[b_1^{k_1} \dots b_p^{k_p} b_{p+1}^{k_{p+1}} \dots b_{p+q}^{k_{p+q}} \right] \tag{A.2}$$

for $\mathbf{k} \in \mathcal{K}^{p,q}(r)$. Note that the k_i's above do not indicate powers of b_j, but rather indices. That is, b_j^k denotes the jth entry of $\mathbf{b}^k$ which is the kth realization of the random vector $\mathbf{b}$. Thus,

$$\boldsymbol{B}^{p,q}(r) \in \mathbb{R}^{\overbrace{N \times \cdots \times N}^{p} \times \overbrace{N_r \times \cdots \times N_r}^{q}}. \tag{A.3}$$

The following lemma plays an important role in upcoming proofs.

Lemma 1. *Let $\mathbf{b} \in \mathbb{R}^{p+q}$ be a random vector and let $\left\{\mathbf{b}^k\right\}$, $k = 1, \dots, N$, denote N independent realizations of $\mathbf{b}$. Define $N_r \in \mathbb{N}$ such that $p + q \leq N_r \leq N$. Let $\mathcal{K}^{p,q}(r)$ and $\boldsymbol{B}^{p,q}(r)$ be defined* *according to* (A.1) *and* (A.2)*, respectively. Then,*

$$\sum_{\mathbf{k}\in\mathcal{K}^{p,q}(r)} B_{\mathbf{k}}^{p,q}(r) \;=\; N^p N_r^q\, \mathbb{E}\left[b_1\right]\ldots\mathbb{E}\left[b_p\right]\mathbb{E}\left[b_{p+1}\right]\ldots\mathbb{E}\left[b_{p+q}\right] + \mathcal{O}(N^p N_r^{q-1} + N^{p-1}N_r^q). \tag{A.4}$$

Proof. Define the following subsets of $\mathcal{K}^{p,q}(r)$:

$$\begin{aligned}\mathcal{K}_1^{p,q}(r) &= \left\{\mathbf{k}\in\mathcal{K}^{p,q}(r)\,|k_i \neq k_j \text{ for } i,j=1,\ldots,p+q \text{ and } i\neq j\right\},\\ \mathcal{K}_2^{p,q}(r) &= \mathcal{K}^{p,q}(r)\setminus\mathcal{K}_1^{p,q}(r).\end{aligned} \tag{A.5}$$

Asymptotically, the subset $\mathcal{K}_1^{p,q}(r)$ has

$$\left(\prod_{i=0}^{q-1}(N_r-i)\right)\left(\prod_{j=q}^{p+q-1}(N-j)\right) \;=\; N^pN_r^q + \mathcal{O}(N^pN_r^{q-1}+N^{p-1}N_r^q) \tag{A.6}$$

elements. Since all of the indices are unique for any $\mathbf{k}\in\mathcal{K}_1^{p,q}(r)$,

$$B_{\mathbf{k}}^{p,q}(r) \;=\; \mathbb{E}\left[b_1\right]\ldots\mathbb{E}\left[b_p\right]\mathbb{E}\left[b_{p+1}\right]\ldots\mathbb{E}\left[b_{p+q}\right]. \tag{A.7}$$

Consider the number of elements in $\mathcal{K}_2^{p,q}(r)$. By construction, any $\mathbf{k}\in\mathcal{K}_2^{p,q}(r)$ must have at least two identical indices. As N and N_r tend towards infinity, the largest subset of elements in $\mathcal{K}_2^{p,q}(r)$ will be those with only two identical elements. There are three such cases:

Case 1:

Two of the k_i's which range from $1,\ldots,N$ are identical and all other k_i's are unique. There are

$$\left(\prod_{i=0}^{q-1}(N_r-i)\right)\left(\prod_{j=q}^{p+q-2}(N-j)\right) \;=\; \mathcal{O}(N^{p-1}N_r^q) \tag{A.8}$$

such elements.

Case 2:

Two of the k_i's which range from $1,\ldots,N_r$ are identical and all other k_i's are unique. There are

$$\left(\prod_{i=0}^{q-2}(N_r-i)\right)\left(\prod_{j=q-1}^{p+q-2}(N-j)\right) \;=\; \mathcal{O}(N^pN_r^{q-1}) \tag{A.9}$$

such elements.

Case 3:

One k_i which range from $1, \ldots, N$ and one k_i which ranges from $1, \ldots, N_r$ are identical and all other k_i's are unique. There are

$$\left(\prod_{i=0}^{q-1}(N_r - i)\right)\left(\prod_{j=q}^{p+q-2}(N-j)\right) = \mathcal{O}(N^{p-1}N_r^q) \tag{A.10}$$

such elements. Thus, there are $\mathcal{O}(N^pN_r^{q-1} + N^{p-1}N_r^q)$ elements in $\mathcal{K}_2^{p,q}(r)$.

Therefore,

$$\begin{aligned}\sum_{\mathbf{k}\in\mathcal{K}^{p,q}(r)} B_{\mathbf{k}}^{p,q}(r) &= \sum_{\mathbf{k}\in\mathcal{K}_1^{p,q}(r)} B_{\mathbf{k}}^{p,q}(r) + \sum_{\mathbf{k}\in\mathcal{K}_2^{p,q}(r)} B_{\mathbf{k}}^{p,q}(r) \\ &= \left(N^pN_r^q + \mathcal{O}(N^pN_r^{q-1} + N^{p-1}N_r^q)\right)\mathbb{E}\left[b_1\right]\ldots\mathbb{E}\left[b_p\right]\mathbb{E}\left[b_{p+1}\right]\ldots\mathbb{E}\left[b_{p+q}\right] \\ &\qquad + \mathcal{O}(N^pN_r^{q-1} + N^{p-1}N_r^q) \\ &= N^pN_r^q\,\mathbb{E}\left[b_1\right]\ldots\mathbb{E}\left[b_p\right]\mathbb{E}\left[b_{p+1}\right]\ldots\mathbb{E}\left[b_{p+q}\right] + \mathcal{O}(N^pN_r^{q-1} + N^{p-1}N_r^q)\end{aligned} \tag{A.11}$$

□

A.1 Proof of Theorem 2

Theorem 2. *Let (Ω, Σ, P) be a probability triple. Suppose that $\mathbf{x} : \Omega \to \mathbb{R}^m$ and $y : \Omega \to \mathbb{R}$ are random variables related by a measurable function $f : \mathbb{R}^m \to \mathbb{R}$ so that $y = f(\mathbf{x})$. Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ be a constant matrix. Then $y \perp\!\!\!\perp \mathbf{x}|\boldsymbol{A}^\top\mathbf{x}$ if and only if $y = g(\boldsymbol{A}^\top\mathbf{x})$ where $g : \mathbb{R}^n \to \mathbb{R}$ is a measurable function.*

Proof. Assume that $y = f(\mathbf{x}) = g(\boldsymbol{A}^\top\mathbf{x})$ for some $\boldsymbol{A}$ and g, then conditional independence follows since the value of $\boldsymbol{A}^\top\mathbf{x}$ fully determines y.

Next, assume that $y \perp\!\!\!\perp \mathbf{x}|\boldsymbol{A}^\top\mathbf{x}$, and note that $\boldsymbol{A}^\top\mathbf{x} : \Omega \to \mathbb{R}^n$ is a random vector. Consider conditional dependence in terms of the generated σ-algebras,

$$\sigma(y) \perp\!\!\!\perp \sigma(\mathbf{x})|\sigma(\boldsymbol{A}^\top\mathbf{x}) \tag{A.12}$$

where

$$\begin{aligned}\sigma(y) &= \left\{y^{-1}(B) \,:\, B \in \mathcal{B}(\mathbb{R})\right\}, \\ \sigma(\mathbf{x}) &= \left\{\mathbf{x}^{-1}(B) \,:\, B \in \mathcal{B}(\mathbb{R}^m)\right\}, \\ \sigma(\boldsymbol{A}^\top\mathbf{x}) &= \left\{(\boldsymbol{A}^\top\mathbf{x})^{-1}(B) \,:\, B \in \mathcal{B}(\mathbb{R}^n)\right\},\end{aligned} \tag{A.13}$$

and $\mathcal{B}(\cdot)$ is the Borel σ-algebra over the indicated domain. By (A.12),

$$\mathbb{E}\left[z \,\middle|\, \sigma(\boldsymbol{A}^\top \mathbf{x})\right] = \mathbb{E}\left[z \,\middle|\, \sigma\left(\sigma(\boldsymbol{A}^\top \mathbf{x}) \cup \sigma(\mathbf{x})\right)\right] \tag{A.14}$$

where $z : \Omega \to \mathbb{R}^+$ is any $\sigma(y)$-measurable random variable [97, Proposition 13, Chapter 3].

Define the partition $y = y^+ - y^-$ where

$$\begin{aligned} y^+ &= f^+(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{if } f(\mathbf{x}) > 0 \\ 0 & \text{otherwise} \end{cases}, \\ y^- &= f^-(\mathbf{x}) = \begin{cases} -f(\mathbf{x}) & \text{if } f(\mathbf{x}) < 0 \\ 0 & \text{otherwise} \end{cases}. \end{aligned} \tag{A.15}$$

Since y is measurable, $y^+ : \Omega \to \mathbb{R}^+$ is $\sigma(y)$-measurable [127]. Applying (A.14) to y^+,

$$\mathbb{E}\left[y^+ \,\middle|\, \sigma(\boldsymbol{A}^\top \mathbf{x})\right] = \mathbb{E}\left[y^+ \,\middle|\, \sigma\left(\sigma(\boldsymbol{A}^\top \mathbf{x}) \cup \sigma(\mathbf{x})\right)\right]. \tag{A.16}$$

The transformation $\boldsymbol{A}^\top \mathbf{x}$ is $\mathcal{B}(\mathbb{R}^n)$-measurable since it is linear. Therefore, $\sigma(\boldsymbol{A}^\top \mathbf{x}) \subseteq \sigma(\mathbf{x})$, and (A.16) becomes

$$\mathbb{E}\left[y^+ \,\middle|\, \sigma(\boldsymbol{A}^\top \mathbf{x})\right] = \mathbb{E}\left[y^+ \,|\, \sigma(\mathbf{x})\right]. \tag{A.17}$$

By the Doob-Dynkin Lemma [97, Proposition 3, Chapter 1], y^+ is $\sigma(\mathbf{x})$-measurable since $y^+ = f^+(\mathbf{x})$. Combining this with (A.17),

$$y^+ = f^+(\mathbf{x}) = \mathbb{E}\left[y^+ \,|\sigma(\mathbf{x})\right] = \mathbb{E}\left[y^+ \,\middle|\, \sigma(\boldsymbol{A}^\top \mathbf{x})\right]. \tag{A.18}$$

This argument also applies for $y^- : \Omega \to \mathbb{R}^+$ such that

$$y^- = f^-(\mathbf{x}) = \mathbb{E}\left[y^- \,|\, \sigma(\mathbf{x})\right] = \mathbb{E}\left[y^- \,\middle|\, \sigma(\boldsymbol{A}^\top \mathbf{x})\right]. \tag{A.19}$$

Thus,

$$\begin{aligned} y &= y^+ - y^- \\ &= \mathbb{E}\left[y^+ \,\middle|\, \sigma(\boldsymbol{A}^\top \mathbf{x})\right] - \mathbb{E}\left[y^- \,\middle|\, \sigma(\boldsymbol{A}^\top \mathbf{x})\right] \\ &= \mathbb{E}\left[y^+ - y^- \,\middle|\, \sigma(\boldsymbol{A}^\top \mathbf{x})\right] \\ &= \mathbb{E}\left[y \,\middle|\, \sigma(\boldsymbol{A}^\top \mathbf{x})\right]. \end{aligned} \tag{A.20}$$

Therefore, y is $\sigma(\boldsymbol{A}^\top \mathbf{x})$-measurable, and by the Doob-Dynkin Lemma [97, Proposition 3, Chapter 1], is a function of $\boldsymbol{A}^\top \mathbf{x}$. That is,

$$y \;=\; g(\boldsymbol{A}^\top \mathbf{x}) \tag{A.21}$$

where $g : \mathbb{R}^n \to \mathbb{R}$ is a deterministic measurable function. $\square$

A.2 Proof of Theorem 3

Theorem 3. *Let $f : \mathbb{R}^m \to \mathbb{R}$ with input probability measure $\pi_{\mathbf{x}}$ admit a central subspace $\mathcal{S}_{f,\pi_{\mathbf{x}}}$, and assume $\pi_{\mathbf{x}}$ admits an elliptically symmetric and standardized density function. Then,* $\operatorname{colspan}(\boldsymbol{C}_{\mathrm{IR}}) \subseteq \mathcal{S}_{f,\pi_{\mathbf{x}}}$.

Proof. Consider the orthogonal $m \times m$ matrix $\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \end{bmatrix}$ where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{m\times(m-n)}$ contain bases for $\mathcal{S}_{f,\pi_{\mathbf{x}}}$ and the orthogonal complement of $\mathcal{S}_{f,\pi_{\mathbf{x}}}$, respectively. Decompose $\mathbf{x}$ into

$$\mathbf{x} \;=\; \boldsymbol{A}\mathbf{u} + \boldsymbol{B}\mathbf{v} \tag{A.22}$$

where $\mathbf{u} = \boldsymbol{A}^\top \mathbf{x} \in \mathbb{R}^n$ and $\mathbf{v} = \boldsymbol{B}^\top \mathbf{x} \in \mathbb{R}^{m-n}$.

Rewrite the conditional expectation in (3.27) as

$$\boldsymbol{\mu}(y) \;=\; \int \left[\int \mathbf{x}\, \mathrm{d}\pi_{\mathbf{x}|\mathbf{u}}(\mathbf{x}) \right] \mathrm{d}\pi_{\mathbf{u}|y}(\mathbf{u}), \tag{A.23}$$

where $\pi_{\mathbf{x}|\mathbf{u}}$ and $\pi_{\mathbf{u}|y}$ are the conditional probability measures over $\mathcal{X}(\mathbf{u}) = \{\mathbf{x} \in \mathbb{R}^m \,:\, \mathbf{u} = \boldsymbol{A}^\top \mathbf{x}\}$ and $g^{-1}(y) = \{\mathbf{u} \in \mathbb{R}^n \,:\, g(\mathbf{u}) = y\}$, respectively. By (A.22),

$$\mathcal{X}(\mathbf{u}) \;=\; \left\{ \mathbf{x} \in \mathbb{R}^m \,|\, \mathbf{x} = \boldsymbol{A}\mathbf{u} + \boldsymbol{B}\mathbf{v},\; \mathbf{v} \in \mathbb{R}^{m-n} \right\}. \tag{A.24}$$

Let $\pi_{\mathbf{v}|\mathbf{u}}$ denote the conditional probability measure induced by (A.22) and $\pi_{\mathbf{x}}$. Since $\mathbf{x}$ is standardized with an elliptically symmetric density and $\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \end{bmatrix}$ is an orthogonal matrix, $\pi_{\mathbf{v}|\mathbf{u}}$ is standardized and elliptically symmetric [73, Chapter 6]. Thus,

$$\begin{aligned} \int \mathbf{x}\, \mathrm{d}\pi_{\mathbf{x}|\mathbf{u}}(\mathbf{x}) \;&=\; \int (\boldsymbol{A}\mathbf{u} + \boldsymbol{B}\mathbf{v})\; \mathrm{d}\pi_{\mathbf{v}|\mathbf{u}}(\mathbf{v}) \\ &=\; \boldsymbol{A}\mathbf{u} \int \mathrm{d}\pi_{\mathbf{v}|\mathbf{u}}(\mathbf{v}) + \boldsymbol{B} \int \mathbf{v}\, \mathrm{d}\pi_{\mathbf{v}|\mathbf{u}}(\mathbf{v}) \\ &=\; \boldsymbol{A}\mathbf{u} \end{aligned} \tag{A.25}$$

Combining (A.23) and (A.25),

$$\boldsymbol{\mu}(y) \; = \; \boldsymbol{A} \int \mathbf{u} \, \mathrm{d}\mu_{\mathbf{u}|y}(\mathbf{u}). \tag{A.26}$$

Then, for $\mathbf{w} \in \operatorname{null}(\boldsymbol{A}^\top)$,

$$\boldsymbol{\mu}(y)^\top \mathbf{w} \; = \; \left(\boldsymbol{A} \int \mathbf{u} \, \mathrm{d}\mu_{\mathbf{u}|y}(\mathbf{u}) \right)^\top \mathbf{w} \; = \; \left(\int \mathbf{u}^\top \, \mathrm{d}\mu_{\mathbf{u}|y}(\mathbf{u}) \right) \left(\boldsymbol{A}^\top \mathbf{w} \right) \; = \; 0. \tag{A.27}$$

By (3.28), $\mathbf{w}^\top \boldsymbol{C}_{\mathrm{IR}} \mathbf{w} = 0$, which implies $\mathbf{w} \in \operatorname{null}(\boldsymbol{C}_{\mathrm{IR}})$. Therefore, $\operatorname{null}(\boldsymbol{A}^\top) \subseteq \operatorname{null}(\boldsymbol{C}_{\mathrm{IR}})$, which implies

$$\operatorname{colspan}(\boldsymbol{C}_{\mathrm{IR}}) \; \subseteq \; \operatorname{colspan}(\boldsymbol{A}) \; = \; \mathcal{S}_{f,\pi_{\mathbf{x}}} \tag{A.28}$$

as required. $\square$

A.3 Proof of Theorem 4

Theorem 4. *Assume that Algorithm 1 has been applied to the data set* $\{[\mathbf{x}_i^\top, y_i]\}$*, with* $i = 1, \ldots, N$*, where the* $\mathbf{x}_i$ *are drawn independently according to* $\pi_{\mathbf{x}}$ *and* $y_i = f(\mathbf{x}_i)$ *are point evaluations of* f*. Then, for* $k = 1, \ldots, m$*,*

$$\mathbb{E}\left[\left(\lambda_k(\boldsymbol{C}_{\mathrm{SIR}}) - \lambda_k(\hat{\boldsymbol{C}}_{\mathrm{SIR}}) \right)^2 \right] \; = \; \mathcal{O}(N_{r_{\min}}^{-1}) \tag{3.36}$$

where $\lambda_k(\cdot)$ *denotes the kth eigenvalue of the given matrix.*

Proof. Recall the SIR matrix from (3.32),

$$\boldsymbol{C}_{\mathrm{SIR}} \; = \; \sum_{r=1}^{R} \omega(r) \, \boldsymbol{\mu}_h(r) \, \boldsymbol{\mu}_h(r)^\top. \tag{A.29}$$

Algorithm 1 computes a sample approximation of $\boldsymbol{C}_{\mathrm{SIR}}$,

$$\hat{\boldsymbol{C}}_{\mathrm{SIR}} \; = \; \sum_{r=1}^{R} \hat{\omega}(r) \, \hat{\boldsymbol{\mu}}_h(r) \, \hat{\boldsymbol{\mu}}_h(r)^\top \tag{A.30}$$

where

$$\hat{\omega}(r) \; = \; \frac{N_r}{N} \qquad \text{and} \qquad \hat{\boldsymbol{\mu}}_h(r) \; = \; \frac{1}{N_r} \sum_{i \in \mathcal{I}_r} \mathbf{x}_i \tag{A.31}$$

where $\mathcal{I}_r$ is the set of indices for which $y_i \in J_r$ and N_r is the cardinality of $\mathcal{I}_r$. Rewrite $\hat{\omega}(r)$ as

$$\hat{\omega}(r) \;=\; \frac{1}{N}\sum_{i=1}^{N}\chi\left(y_i \in J_r\right) \tag{A.32}$$

where $\chi\left(y_i \in J_r\right)$ is an indicator function that is 1 when $y_i \in J_r$ and 0 otherwise.

Fix r and let

$$\omega \;=\; \omega(r), \quad \hat{\omega} \;=\; \hat{\omega}(r), \quad \chi_i \;=\; \chi\left(y_i \in J_r\right), \quad \boldsymbol{\mu} \;=\; \boldsymbol{\mu}_h(r), \quad \hat{\boldsymbol{\mu}} \;=\; \hat{\boldsymbol{\mu}}_h(r) \tag{A.33}$$

for convenience. Assume without loss of generality that $\mathcal{I}_r = \{1, \ldots, N_r\}$.

To compute the mean squared error, we focus on the computation of a single element in $\hat{\boldsymbol{C}}_{\mathrm{SIR}}$. To do this, we move the sample index to the superscript and let the subscript denote the vector or matrix element. That is, we let $\hat{x}_i^k$ denote the ith element of the vector $\hat{\mathbf{x}}^k$ which is the kth realization of the random vector $\mathbf{x}$.

For $1 \le i, j \le m$,

$$\begin{aligned}
\mathbb{E}\left[\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j\right] \;&=\; \mathbb{E}\left[\left(\frac{1}{N}\sum_{k_1=1}^{N}\chi^{k_1}\right)\left(\frac{1}{N_r}\sum_{k_2=1}^{N_r}x_i^{k_2}\right)\left(\frac{1}{N_r}\sum_{k_3=1}^{N_r}x_j^{k_3}\right)\right] \\
&=\; \frac{1}{N\,N_r^2}\sum_{k_1=1}^{N}\sum_{k_2,k_3=1}^{N_r}\mathbb{E}\left[\chi^{k_1}\,x_i^{k_2}\,x_j^{k_3}\right].
\end{aligned} \tag{A.34}$$

Using (A.1) and (A.2), write

$$\mathbb{E}\left[\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j\right] \;=\; \frac{1}{N\,N_r^2}\sum_{\mathbf{k}\in\mathcal{K}^{1,2}} B_{\mathbf{k}}^{1,2}. \tag{A.35}$$

Note that we drop the argument from $\mathcal{K}_{\mathbf{k}}^{1,2}(r)$ and $B_{\mathbf{k}}^{1,2}(r)$ since r is fixed.

By Lemma 1,

$$\begin{aligned}
\mathbb{E}\left[\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j\right] \;&=\; \frac{1}{N\,N_r^2}\sum_{\mathbf{k}\in\mathcal{K}^{1,2}} B_{\mathbf{k}}^{1,2} \\
&=\; \frac{1}{N\,N_r^2}\left[N\,N_r^2\,\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_i\right]\mathbb{E}\left[x_j\right] + \mathcal{O}(N\,N_r + N_r^2)\right] \\
&=\; \mathbb{E}\left[\chi\right]\mathbb{E}\left[x_i\right]\mathbb{E}\left[x_j\right] + \mathcal{O}(N_r^{-1}) \\
&=\; \omega\,\mu_i\,\mu_j + \mathcal{O}(N_r^{-1}).
\end{aligned} \tag{A.36}$$

Consider

$$\operatorname{Var}[\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j] \;=\; \mathbb{E}\left[(\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j)^2\right] - \mathbb{E}[\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j]^2 . \tag{A.37}$$

To compute $\mathbb{E}\left[(\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j)^2\right]$,

$$\begin{aligned}\mathbb{E}\left[(\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j)^2\right] \;&=\; \mathbb{E}\left[\left(\frac{1}{N}\sum_{k_1=1}^{N}\chi^{k_1}\right)^2\left(\frac{1}{N_r}\sum_{k_2=1}^{N_r}x_i^{k_2}\right)^2\left(\frac{1}{N_r}\sum_{k_3=1}^{N_r}x_j^{k_3}\right)^2\right]\\ &=\; \frac{1}{N^2N_r^4}\sum_{k_1,k_2=1}^{N}\;\sum_{k_3,k_4,k_5,k_6=1}^{N_r}\mathbb{E}\left[\chi^{k_1}\,\chi^{k_2}\,x_i^{k_3}\,x_i^{k_4}\,x_j^{k_5}\,x_j^{k_6}\right].\end{aligned} \tag{A.38}$$

Using (A.1) and (A.2),

$$\mathbb{E}\left[(\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j)^2\right] \;=\; \frac{1}{N^2N_r^4}\sum_{\mathbf{k}\in\mathcal{K}^{2,4}}B_{\mathbf{k}}^{2,4}. \tag{A.39}$$

By Lemma 1,

$$\begin{aligned}\mathbb{E}\left[(\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j)^2\right] \;&=\; \frac{1}{N^2N_r^4}\sum_{\mathbf{k}\in\mathcal{K}^{2,4}}B_{\mathbf{k}}^{2,4}\\ &=\; \frac{1}{N^2N_r^4}\left[N^2N_r^4\,\mathbb{E}[\chi]^2\;\mathbb{E}[x_i]^2\;\mathbb{E}[x_j]^2+\mathcal{O}(N^2N_r^3+NN_r^4)\right]\\ &=\; \mathbb{E}[\chi]^2\;\mathbb{E}[x_i]^2\;\mathbb{E}[x_j]^2+\mathcal{O}(N_r^{-1})\\ &=\; \omega^2\,\mu_i^2\,\mu_j^2+\mathcal{O}(N_r^{-1}).\end{aligned} \tag{A.40}$$

Thus,

$$\begin{aligned}\operatorname{Var}[\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j] \;&=\; \mathbb{E}\left[(\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j)^2\right]-\mathbb{E}[\hat{\omega}\,\hat{\mu}_i\,\hat{\mu}_j]^2\\ &=\; \left(\omega^2\,\mu_i^2\,\mu_j^2+\mathcal{O}(N_r^{-1})\right)-\left(\omega\,\mu_i\,\mu_j+\mathcal{O}(N_r^{-1})\right)^2\\ &=\; \mathcal{O}(N_r^{-1}).\end{aligned} \tag{A.41}$$

Since (A.36) and (A.41) hold for each $r=1,\ldots,R$,

$$\mathbb{E}\left[(\hat{C}_{\text{SIR}})_{ij}\right] \;=\; (C_{\text{SIR}})_{ij}+\mathcal{O}\left(N_{r_{\min}}^{-1}\right) \quad\text{and}\quad \operatorname{Var}\left[(\hat{C}_{\text{SIR}})_{ij}\right] \;=\; \mathcal{O}\left(N_{r_{\min}}^{-1}\right) \tag{A.42}$$

where $N_{r_{\min}}$ is the same from (3.35),

$$N_{r_{\min}} \;=\; \min_{1\le r\le R}N_r. \tag{A.43}$$

The mean squared error for each element of $\hat{\boldsymbol{C}}_{\text{SIR}}$ is

$$\begin{aligned}\text{MSE}\left[(\hat{C}_{\text{SIR}})_{ij}\right] &= \text{Bias}\left[(\hat{C}_{\text{SIR}})_{ij}\right]^2 + \text{Var}\left[(\hat{C}_{\text{SIR}})_{ij}\right] \\ &= \left(\mathbb{E}\left[(\hat{C}_{\text{SIR}})_{ij}\right] - (C_{\text{SIR}})_{ij}\right)^2 + \text{Var}\left[(\hat{C}_{\text{SIR}})_{ij}\right] \\ &= \left((C_{\text{SIR}})_{ij} + \mathcal{O}\left(N_{r_{\min}}^{-1}\right) - (C_{\text{SIR}})_{ij}\right)^2 + \mathcal{O}\left(N_{r_{\min}}^{-1}\right) \\ &= \mathcal{O}\left(N_{r_{\min}}^{-1}\right).\end{aligned} \tag{A.44}$$

Next, we examine how the element-wise mean squared error in (A.44) translates to errors in the eigenvalue estimates. By Corollary 8.1.6 in [64],

$$\left|\lambda_k(\boldsymbol{C}_{\text{SIR}}) - \lambda_k(\hat{\boldsymbol{C}}_{\text{SIR}})\right| \leq ||\boldsymbol{E}||_2 \tag{A.45}$$

where $\boldsymbol{E} = \boldsymbol{C}_{\text{SIR}} - \hat{\boldsymbol{C}}_{\text{SIR}}$. Since $||\cdot||_2 \leq ||\cdot||_F$,

$$\left|\lambda_k(\boldsymbol{C}_{\text{SIR}}) - \lambda_k(\hat{\boldsymbol{C}}_{\text{SIR}})\right| \leq ||\boldsymbol{E}||_F. \tag{A.46}$$

Squaring both sides and taking the expectation,

$$\mathbb{E}\left[\left(\lambda_k(\boldsymbol{C}_{\text{SIR}}) - \lambda_k(\hat{\boldsymbol{C}}_{\text{SIR}})\right)^2\right] \leq \mathbb{E}\left[||\boldsymbol{E}||_F^2\right]. \tag{A.47}$$

Consider

$$\mathbb{E}\left[||\boldsymbol{E}||_F^2\right] = \sum_{i,j=1}^{m} \mathbb{E}\left[E_{ij}^2\right] = \sum_{i,j=1}^{m} \mathbb{E}\left[\left((C_{\text{SIR}})_{ij} - (\hat{C}_{\text{SIR}})_{ij}\right)^2\right] = \sum_{i,j=1}^{m} \text{MSE}\left[(\hat{C}_{\text{SIR}})_{ij}\right]. \tag{A.48}$$

By (A.44),

$$\mathbb{E}\left[\left(\lambda_k(\boldsymbol{C}_{\text{SIR}}) - \lambda_k(\hat{\boldsymbol{C}}_{\text{SIR}})\right)^2\right] = \mathcal{O}(N_{r_{\min}}^{-1}) \tag{A.49}$$

as required. □

A.4 Proof of Theorem 5

Theorem 5. *Assume the same conditions from Theorem 4. Then, for sufficiently large N,*

$$\text{dist}\left(\text{ran}(\boldsymbol{A}), \text{ran}(\hat{\boldsymbol{A}})\right) = \frac{1}{\lambda_n(\boldsymbol{C}_{\text{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SIR}})}\,\mathcal{O}_p(N_{r_{\min}}^{-1/2}), \tag{3.38}$$

where $\mathcal{O}_p$ denotes convergence in probability.

Proof. Recall $\boldsymbol{A}, \hat{\boldsymbol{A}} \in \mathbb{R}^{m\times n}$ contain the first n eigenvectors of $\boldsymbol{C}_{\text{SIR}}$ and $\hat{\boldsymbol{C}}_{\text{SIR}}$, respectively. Let $\boldsymbol{B}, \hat{\boldsymbol{B}} \in \mathbb{R}^{m\times(m-n)}$ contain the last $m-n$ eigenvectors of each matrix.

By Corollary 8.1.11 in [64], if

$$||\boldsymbol{E}||_2 \ \leq \ \frac{\lambda_n(\boldsymbol{C}_{\text{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SIR}})}{5}, \tag{A.50}$$

then

$$\text{dist}\left(\text{ran}(\boldsymbol{A}),\, \text{ran}(\hat{\boldsymbol{A}})\right) \ \leq \ \frac{4}{\lambda_n(\boldsymbol{C}_{\text{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SIR}})}||\boldsymbol{E}_{21}||_2 \tag{A.51}$$

where $\boldsymbol{E} = \boldsymbol{C}_{\text{SIR}} - \hat{\boldsymbol{C}}_{\text{SIR}}$ and $\boldsymbol{E}_{21} = \boldsymbol{B}^\top \boldsymbol{E}\boldsymbol{A}$. In what follows, we show that (A.50) holds with high probability for sufficiently large N.

Theorem 2.6 from [114] states that for any $\tau > 0$

$$\mathbb{P}\left(\ ||\boldsymbol{E}||_F \leq \tau\sqrt{\mathbb{E}\left[||\boldsymbol{E}||_F^2\right]}\ \right) \ \geq \ 1 - \frac{1}{\tau^2}. \tag{A.52}$$

Choose τ_* to be large such that $1/\tau_*^2$ is arbitrarily close to zero, and recognize that $N_{r_{\min}} \to \infty$ as $N \to \infty$ since $\omega(r) > 0$ for each $r = 1, \dots, R$. By (A.48), there exists N_* such that

$$\tau_*\sqrt{\mathbb{E}\left[||\boldsymbol{E}||_F^2\right]} \ \leq \ \frac{\lambda_n(\boldsymbol{C}_{\text{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SIR}})}{5} \tag{A.53}$$

when $N > N_*$. Combining this with (A.52),

$$\mathbb{P}\left(\ ||\boldsymbol{E}||_F \leq \frac{\lambda_n(\boldsymbol{C}_{\text{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SIR}})}{5}\ \right) \ \geq \ 1 - \frac{1}{\tau_*^2}. \tag{A.54}$$

Since $||\cdot||_2 \leq ||\cdot||_F$,

$$\mathbb{P}\left(\ ||\boldsymbol{E}||_2 \leq \frac{\lambda_n(\boldsymbol{C}_{\text{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SIR}})}{5}\ \right) \geq 1 - \frac{1}{\tau_*^2}. \tag{A.55}$$

Thus, (A.50) is satisfied with probability $1 - 1/\tau_*^2$ when $N > N_*$. In this case,

$$\begin{aligned}\text{dist}\left(\text{ran}(\boldsymbol{A}),\, \text{ran}(\hat{\boldsymbol{A}})\right) &\leq \frac{4}{\lambda_n(\boldsymbol{C}_{\text{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SIR}})}||\boldsymbol{E}_{21}||_2 \\ &\leq \frac{4}{\lambda_n(\boldsymbol{C}_{\text{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SIR}})}||\boldsymbol{E}||_F \\ &\leq \frac{4}{\lambda_n(\boldsymbol{C}_{\text{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SIR}})}\tau_*\sqrt{\mathbb{E}\left[||\boldsymbol{E}||_F^2\right]},\end{aligned} \tag{A.56}$$

by Corollary 8.1.11 in [64]. Thus,

$$\operatorname{dist}\left(\operatorname{ran}(\boldsymbol{A}),\, \operatorname{ran}(\hat{\boldsymbol{A}})\right) \;=\; \frac{1}{\lambda_n(\boldsymbol{C}_{\text{SIR}}) - \lambda_{n+1}(\boldsymbol{C}_{\text{SIR}})}\mathcal{O}_p(N_{r_{\min}}^{-1/2}). \tag{A.57}$$

$\square$

A.5 Proof of Theorem 6

Theorem 6. *Let $f : \mathbb{R}^m \to \mathbb{R}$ with input probability measure $\pi_{\mathbf{x}}$ admit a central subspace $\mathcal{S}_{f,\pi_{\mathbf{x}}}$, and assume $\pi_{\mathbf{x}}$ admits an elliptically symmetric and standardized density function. Then,* $\operatorname{colspan}(\boldsymbol{C}_{\text{AVE}}) \subseteq \mathcal{S}_{f,\pi_{\mathbf{x}}}$.

Proof. Consider the orthogonal $m \times m$ matrix $\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \end{bmatrix}$ where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{m\times(m-n)}$ contain bases for $\mathcal{S}_{f,\pi_{\mathbf{x}}}$ and the orthogonal complement of $\mathcal{S}_{f,\pi_{\mathbf{x}}}$, respectively. Decompose $\mathbf{x}$ into

$$\mathbf{x} \;=\; \boldsymbol{A}\mathbf{u} + \boldsymbol{B}\mathbf{v} \tag{A.58}$$

where $\mathbf{u} = \boldsymbol{A}^\top\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{v} = \boldsymbol{B}^\top\mathbf{x} \in \mathbb{R}^{m-n}$.

Rewrite (3.40) as

$$\boldsymbol{\Sigma}(y) \;=\; \int \mathbf{x}\,\mathbf{x}^\top \, \mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}) - \boldsymbol{\mu}(y)\,\boldsymbol{\mu}(y)^\top, \tag{A.59}$$

and decompose the first term of (A.59) similar to (A.23)

$$\int \mathbf{x}\,\mathbf{x}^\top \, \mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}) \;=\; \int \left[\int \mathbf{x}\,\mathbf{x}^\top \, \mathrm{d}\pi_{\mathbf{x}|\mathbf{u}}(\mathbf{x})\right] \mathrm{d}\pi_{\mathbf{u}|y}(\mathbf{u}), \tag{A.60}$$

where $\pi_{\mathbf{x}|\mathbf{u}}$ and $\pi_{\mathbf{u}|y}$ are the conditional probability measures over $\mathcal{X}(\mathbf{u}) = \{\mathbf{x} \in \mathbb{R}^m \,:\, \mathbf{u} = \boldsymbol{A}^\top\mathbf{x}\}$ and $g^{-1}(y) = \{\mathbf{u} \in \mathbb{R}^n \,:\, g(\mathbf{u}) = y\}$, respectively. By (A.58),

$$\int \mathbf{x}\mathbf{x}^\top \, \mathrm{d}\pi_{\mathbf{x}|\mathbf{u}}(\mathbf{x}) \;=\; \int (\boldsymbol{A}\mathbf{u} + \boldsymbol{B}\mathbf{v})\,(\boldsymbol{A}\mathbf{u} + \boldsymbol{B}\mathbf{v})^\top \, \mathrm{d}\pi_{\mathbf{v}|\mathbf{u}}(\mathbf{v}), \tag{A.61}$$

where $\pi_{\mathbf{v}|\mathbf{u}}$ is the conditional measure induced by (A.58) and $\pi_{\mathbf{x}}$ over the set $\mathcal{X}(\mathbf{u})$ from (A.24).

Expanding and simplifying (A.61),

$$\begin{aligned}
\int \mathbf{x}\mathbf{x}^\top \,\mathrm{d}\pi_{\mathbf{x}|\mathbf{u}}(\mathbf{x}) &= \int (\boldsymbol{A}\mathbf{u}+\boldsymbol{B}\mathbf{v})\,(\boldsymbol{A}\mathbf{u}+\boldsymbol{B}\mathbf{v})^\top \,\mathrm{d}\pi_{\mathbf{v}|\mathbf{u}}(\mathbf{v}) \\
&= \int \left(\boldsymbol{A}\mathbf{u}\mathbf{u}^\top\boldsymbol{A}^\top + \boldsymbol{A}\mathbf{u}\mathbf{v}^\top\boldsymbol{B}^\top + \boldsymbol{B}\mathbf{v}\mathbf{u}^\top\boldsymbol{A}^\top + \boldsymbol{B}\mathbf{v}\mathbf{v}^\top\boldsymbol{B}^\top\right) \mathrm{d}\pi_{\mathbf{v}|\mathbf{u}}(\mathbf{v}) \\
&= \boldsymbol{A}\mathbf{u}\mathbf{u}^\top\boldsymbol{A}^\top \left(\int \mathrm{d}\pi_{\mathbf{v}|\mathbf{u}}(\mathbf{v})\right) + \boldsymbol{A}\mathbf{u}\left(\int \mathbf{v}^\top \,\mathrm{d}\pi_{\mathbf{v}|\mathbf{u}}(\mathbf{v})\right)\boldsymbol{B}^\top \ldots \\
&\quad + \boldsymbol{B}\left(\int \mathbf{v}\,\mathrm{d}\pi_{\mathbf{v}|\mathbf{u}}(\mathbf{v})\right)\mathbf{u}^\top\boldsymbol{A}^\top + \boldsymbol{B}\left(\int \mathbf{v}\mathbf{v}^\top \,\mathrm{d}\pi_{\mathbf{v}|\mathbf{u}}(\mathbf{v})\right)\boldsymbol{B}^\top \\
&= \boldsymbol{A}\mathbf{u}\mathbf{u}^\top\boldsymbol{A}^\top + \boldsymbol{B}\boldsymbol{B}^\top.
\end{aligned} \tag{A.62}$$

Plugging this result into (A.60),

$$\begin{aligned}
\int \mathbf{x}\mathbf{x}^\top \,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x}) &= \int \left[\boldsymbol{A}\mathbf{u}\mathbf{u}^\top\boldsymbol{A}^\top + \boldsymbol{B}\boldsymbol{B}^\top\right] \mathrm{d}\pi_{\mathbf{u}|y}(\mathbf{u}) \\
&= \boldsymbol{A}\left(\int \mathbf{u}\mathbf{u}^\top \,\mathrm{d}\pi_{\mathbf{u}|y}(\mathbf{u})\right)\boldsymbol{A}^\top + \boldsymbol{B}\boldsymbol{B}^\top.
\end{aligned} \tag{A.63}$$

For $\mathbf{w} \in \mathrm{null}(\boldsymbol{A}^\top)$,

$$\boldsymbol{\Sigma}(y)\mathbf{w} \;=\; \left(\int \mathbf{x}\mathbf{x}^\top \,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x})\right)\mathbf{w} - \boldsymbol{\mu}(y)\boldsymbol{\mu}(y)^\top\mathbf{w} \;=\; \left(\int \mathbf{x}\mathbf{x}^\top \,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x})\right)\mathbf{w}, \tag{A.64}$$

since $\boldsymbol{\mu}(y)^\top\mathbf{w} = 0$ as shown in (A.27). Therefore,

$$\begin{aligned}
\boldsymbol{\Sigma}(y)\mathbf{w} &= \left(\int \mathbf{x}\mathbf{x}^\top \,\mathrm{d}\pi_{\mathbf{x}|y}(\mathbf{x})\right)\mathbf{w} \\
&= \left(\boldsymbol{A}\left(\int \mathbf{u}\mathbf{u}^\top \,\mathrm{d}\pi_{\mathbf{u}|y}(\mathbf{u})\right)\boldsymbol{A}^\top + \boldsymbol{B}\boldsymbol{B}^\top\right)\mathbf{w} \\
&= \boldsymbol{A}\left(\int \mathbf{u}\mathbf{u}^\top \,\mathrm{d}\pi_{\mathbf{u}|y}(\mathbf{u})\right)\left(\boldsymbol{A}^\top\mathbf{w}\right) + \boldsymbol{B}\boldsymbol{B}^\top\mathbf{w} \\
&= \mathbf{0} + \mathbf{w} \\
&= \mathbf{w}.
\end{aligned} \tag{A.65}$$

By (3.41), $\mathbf{w}^\top\boldsymbol{C}_{\mathrm{AVE}}\mathbf{w} = 0$, which implies that $\mathbf{w} \in \mathrm{null}(\boldsymbol{C}_{\mathrm{AVE}})$. Therefore, $\mathrm{null}(\boldsymbol{A}^\top) \subseteq \mathrm{null}(\boldsymbol{C}_{\mathrm{AVE}})$, which implies,

$$\mathrm{colspan}(\boldsymbol{C}_{\mathrm{AVE}}) \;\subseteq\; \mathrm{colspan}(\boldsymbol{A}) \;=\; \mathcal{S}_{f,\pi_{\mathbf{x}}} \tag{A.66}$$

as required. □

A.6 Proof of Theorem 7

Theorem 7. *Assume that Algorithm 2 has been applied to the data set* $\{[\mathbf{x}_i^\top, y_i]\}$*, with* $i = 1, \dots, N$*, where the* $\mathbf{x}_i$ *are drawn independently according to* $\pi_{\mathbf{x}}$ *and* $y_i = f(\mathbf{x}_i)$ *are point evaluations of* f*. Then, for* $k = 1, \dots, m$*,*

$$\mathbb{E}\left[\left(\lambda_k(\boldsymbol{C}_{\text{SAVE}}) - \lambda_k(\hat{\boldsymbol{C}}_{\text{SAVE}})\right)^2\right] \;=\; \mathcal{O}(N_{r_{\min}}^{-1}) \tag{3.50}$$

where $\lambda_k(\cdot)$ *denotes the kth eigenvalue of the given matrix.*

Proof. Algorithm 2 computes a sample approximation of (3.18),

$$\hat{\boldsymbol{C}}_{\text{SAVE}} \;=\; \sum_{r=1}^{R} \hat{\omega}(r)\,\left(\boldsymbol{I} - \hat{\boldsymbol{\Sigma}}_h(r)\right)^2, \tag{A.67}$$

where

$$\hat{\omega}(r) \;=\; \frac{N_r}{N} \qquad \text{and} \qquad \hat{\boldsymbol{\Sigma}}_h(r) \;=\; \frac{1}{N_r - 1}\sum_{i\in\mathcal{I}_r} \left(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_h(r)\right)\left(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_h(r)\right)^\top \tag{A.68}$$

where $\mathcal{I}_r$ is the set of indices such that $y_i \in J_r$, N_r is the cardinality of $\mathcal{I}_r$, and $\hat{\boldsymbol{\mu}}_h(r)$ is the sample estimate of the average from (3.12). Rewrite (A.68) as

$$\hat{\omega}(r) \;=\; \frac{1}{N}\sum_{i=1}^{N} \chi\left(y_i \in J_r\right) \quad \text{and} \quad \hat{\boldsymbol{\Sigma}}_h(r) \;=\; \frac{1}{N_r - 1}\sum_{i\in\mathcal{I}_r} \mathbf{x}_i\mathbf{x}_i^\top - \frac{N_r}{N_r - 1}\hat{\boldsymbol{\mu}}_h(r)\hat{\boldsymbol{\mu}}_h(r)^\top \tag{A.69}$$

Fix r and let

$$\begin{gathered} \omega \;=\; \omega(r), \quad \hat{\omega} \;=\; \hat{\omega}(r), \quad \chi_i \;=\; \chi\left(y_i \in J_r\right), \\ \boldsymbol{\mu} \;=\; \boldsymbol{\mu}_h(r), \quad \hat{\boldsymbol{\mu}} \;=\; \hat{\boldsymbol{\mu}}_h(r), \quad \boldsymbol{\Sigma} \;=\; \boldsymbol{\Sigma}_h(r), \quad \text{and} \quad \hat{\boldsymbol{\Sigma}} \;=\; \hat{\boldsymbol{\Sigma}}_h(r). \end{gathered} \tag{A.70}$$

Assume without loss of generality that $\mathcal{I}_r = \{1, \dots, N_r\}$.

To compute the mean squared error, we focus on the computation of a single element in $\hat{\boldsymbol{C}}_{\text{SAVE}}$. To do this, we move the sample index to the superscript and let the subscript denote the vector or matrix element similar to proof of Theorem 4. Thus, for $1 \leq i, j \leq m$,

$$\begin{aligned}
\mathbb{E}\left[\hat{\omega}\left(\delta_{ij} - \hat{\Sigma}_{ij}\right)^2\right] &= \mathbb{E}\left[\left(\frac{1}{N}\sum_{k_1=1}^{N}\chi^{k_1}\right)\left(\delta_{ij} - \left(\frac{1}{N_r - 1}\sum_{k_2=1}^{N_r} x_i^{k_2} x_j^{k_2} \ \ldots\right.\right.\right. \\
&\qquad \left.\left.\left. - \left(\frac{1}{N_r}\sum_{k_3=1}^{N_r} x_i^{k_3}\right)\left(\frac{1}{N_r}\sum_{k_4=1}^{N_r} x_j^{k_4}\right)\right)\right)^2\right] \\
&= \frac{\delta_{ij}}{N}\sum_{k_1=1}^{N}\mathbb{E}\left[\chi^{k_1}\right] - \frac{2\delta_{ij}}{N(N_r-1)}\sum_{k_1=1}^{N}\sum_{k_2=1}^{N_r}\mathbb{E}\left[\chi^{k_1} x_i^{k_2} x_j^{k_2}\right]\ldots \\
&\quad + \frac{2\delta_{ij}}{NN_r^2}\sum_{k_1=1}^{N}\sum_{k_2,k_3=1}^{N_r}\mathbb{E}\left[\chi^{k_1} x_i^{k_2} x_j^{k_3}\right]\ldots \\
&\quad + \frac{1}{N(N_r-1)^2}\sum_{k_1=1}^{N}\sum_{k_2,k_3=1}^{N_r}\mathbb{E}\left[\chi^{k_1} x_i^{k_2} x_j^{k_2} x_i^{k_3} x_j^{k_3}\right]\ldots \\
&\quad - \frac{2}{NN_r^2(N_r-1)}\sum_{k_1=1}^{N}\sum_{k_2,k_3,k_4=1}^{N_r}\mathbb{E}\left[\chi^{k_1} x_i^{k_2} x_j^{k_2} x_i^{k_3} x_j^{k_4}\right]\ldots \\
&\quad + \frac{1}{NN_r^4}\sum_{k_1=1}^{N}\sum_{k_2,k_3,k_4,k_5=1}^{N_r}\mathbb{E}\left[\chi^{k_1} x_i^{k_2} x_j^{k_3} x_i^{k_4} x_j^{k_5}\right].
\end{aligned} \tag{A.71}$$

Using (A.1) and (A.2)

$$\begin{aligned}
\mathbb{E}\left[\hat{\omega}\left(\delta_{ij} - \hat{\Sigma}_{ij}\right)^2\right] &= \frac{\delta_{ij}}{N}\sum_{k_1=1}^{N}\mathbb{E}\left[\chi^{k_1}\right] - \frac{2\delta_{ij}}{N(N_r-1)}\sum_{\mathbf{k}\in\mathcal{K}^{1,1}} B_{\mathbf{k}}^{1,1}\ldots \\
&\quad + \frac{2\delta_{ij}}{NN_r^2}\sum_{\mathbf{k}\in\mathcal{K}^{1,2}} B_{\mathbf{k}}^{1,2} + \frac{1}{N(N_r-1)^2}\sum_{\mathbf{k}\in\mathcal{K}^{1,2}} B_{\mathbf{k}}^{1,2}\ldots \\
&\quad - \frac{2}{NN_r^2(N_r-1)}\sum_{\mathbf{k}\in\mathcal{K}^{1,3}} B_{\mathbf{k}}^{1,3} + \frac{1}{NN_r^4}\sum_{\mathbf{k}\in\mathcal{K}^{1,4}} B_{\mathbf{k}}^{1,4}.
\end{aligned} \tag{A.72}$$

By Lemma 1,

$$\begin{aligned}
\mathbb{E}\left[\hat{\omega}\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2\right] &= \frac{\delta_{ij}}{N}\left[N\,\mathbb{E}\left[\chi\right]\right]-\frac{2\delta_{ij}}{N(N_r-1)}\left[NN_r\,\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_ix_j\right]+\mathcal{O}(N+N_r)\right]\dots\\
&+\frac{2\delta_{ij}}{NN_r^2}\left[NN_r^2\,\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_i\right]\mathbb{E}\left[x_j\right]+\mathcal{O}(NN_r+N_r^2)\right]\dots\\
&+\frac{1}{N(N_r-1)^2}\left[NN_r^2\,\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_ix_j\right]^2+\mathcal{O}(NN_r+N_r^2)\right]\dots\\
&-\frac{2}{NN_r^2(N_r-1)}\left[NN_r^3\,\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_ix_j\right]\mathbb{E}\left[x_i\right]\mathbb{E}\left[x_j\right]+\mathcal{O}(NN_r^2+N_r^3)\right]\dots\\
&+\frac{1}{NN_r^4}\left[NN_r^4\,\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_i\right]^2\mathbb{E}\left[x_j\right]^2+\mathcal{O}(NN_r^3+N_r^4)\right]\\
&= \delta_{ij}\mathbb{E}\left[\chi\right]-2\delta_{ij}\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_ix_j\right]+2\delta_{ij}\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_i\right]\mathbb{E}\left[x_j\right]\dots\\
&+\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_ix_j\right]^2-2\,\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_ix_j\right]\mathbb{E}\left[x_i\right]\mathbb{E}\left[x_j\right]\dots\\
&+\mathbb{E}\left[\chi\right]\mathbb{E}\left[x_i\right]^2\mathbb{E}\left[x_j\right]^2+\mathcal{O}(N_r^{-1})\\
&= \mathbb{E}\left[\chi\right]\left(\delta_{ij}-\left(\mathbb{E}\left[x_ix_j\right]-\mathbb{E}\left[x_i\right]\mathbb{E}\left[x_j\right]\right)\right)^2+\mathcal{O}(N_r^{-1})\\
&= \omega\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2+\mathcal{O}(N_r^{-1}).
\end{aligned} \tag{A.73}$$

Consider

$$\mathrm{Var}\left[\hat{\omega}\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2\right]=\mathbb{E}\left[\left(\hat{\omega}\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2\right)^2\right]-\mathbb{E}\left[\hat{\omega}\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2\right]^2. \tag{A.74}$$

To find $\mathbb{E}\left[\left(\hat{\omega}\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2\right)^2\right]$,

$$\begin{aligned}
\mathbb{E}\left[\left(\hat{\omega}\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2\right)^2\right] &= \mathbb{E}\left[\left(\frac{1}{N}\sum_{k_1=1}^{N}\chi^{k_1}\right)^2\left(\delta_{ij}-\left(\frac{1}{N_r-1}\sum_{k_2=1}^{N_r}x_i^{k_2}x_j^{k_2}\quad\dots\right.\right.\right.\\
&\left.\left.\left.-\left(\frac{1}{N_r}\sum_{k_3=1}^{N_r}x_i^{k_3}\right)\left(\frac{1}{N_r}\sum_{k_4=1}^{N_r}x_j^{k_4}\right)\right)\right)^4\right].
\end{aligned} \tag{A.75}$$

By expanding, we can rewrite (A.75) in a form which can be simplified using the tensor summation notation from (A.1) and (A.2). We then apply Lemma 1 to simplify these summations and obtain

$$\mathbb{E}\left[\left(\hat{\omega}\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2\right)^2\right] = \left(\omega\left(\delta_{ij}-\Sigma_{ij}\right)^2\right)^2+\mathcal{O}(N_r^{-1}). \tag{A.76}$$

Thus,

$$\begin{aligned}
\mathrm{Var}\left[\hat{\omega}\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2\right] &= \mathbb{E}\left[\left(\hat{\omega}\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2\right)^2\right] - \mathbb{E}\left[\hat{\omega}\left(\delta_{ij}-\hat{\Sigma}_{ij}\right)^2\right]^2 \\
&= \left(\left(\omega\left(\delta_{ij}-\Sigma_{ij}\right)^2\right)^2 + \mathcal{O}(N_r^{-1})\right) \\
&\qquad - \left(\omega\left(\delta_{ij}-\Sigma_{ij}\right)^2 + \mathcal{O}(N_r^{-1})\right)^2 \\
&= \mathcal{O}(N_r^{-1}).
\end{aligned} \tag{A.77}$$

Since (A.73) and (A.77) hold for each $r = 1, \ldots, R$,

$$\mathbb{E}\left[(\hat{C}_{\mathrm{SAVE}})_{ij}\right] = (C_{\mathrm{SAVE}})_{ij} + \mathcal{O}\left(N_{r_{\min}}^{-1}\right), \qquad \mathrm{Var}\left[(\hat{C}_{\mathrm{SAVE}})_{ij}\right] = \mathcal{O}\left(N_{r_{\min}}^{-1}\right) \tag{A.78}$$

where $N_{r_{\min}}$ from (3.35) denotes the minimum number of samples in any one slice. From (A.78), the mean squared error for a single element of the SAVE matrix is

$$\mathrm{MSE}\left[(\hat{C}_{\mathrm{SAVE}})_{ij}\right] = \mathcal{O}(N_{r_{\min}}^{-1}). \tag{A.79}$$

Similar to the proof of Theorem 4, we use this mean squared error to obtain

$$\mathbb{E}\left[||\boldsymbol{E}||_F^2\right] = \mathcal{O}(N_{r_{\min}}^{-1}) \tag{A.80}$$

where $\boldsymbol{E} = \boldsymbol{C}_{\mathrm{SAVE}} - \hat{\boldsymbol{C}}_{\mathrm{SAVE}}$. Combining this result with Corollary 8.1.6 in [64] yields the desired result,

$$\mathbb{E}\left[\left(\lambda_k(\boldsymbol{C}_{\mathrm{SAVE}}) - \lambda_k(\hat{\boldsymbol{C}}_{\mathrm{SAVE}})\right)^2\right] = \mathcal{O}\left(N_{r_{\min}}^{-1}\right). \tag{A.81}$$

□

A.7 Proof of Theorem 8

Theorem 8. *Assume the same conditions from Theorem 7. Then, for sufficiently large N,*

$$\mathrm{dist}\left(\mathrm{ran}(\boldsymbol{A}), \mathrm{ran}(\hat{\boldsymbol{A}})\right) = \frac{1}{\lambda_n(\boldsymbol{C}_{\mathrm{SAVE}}) - \lambda_{n+1}(\boldsymbol{C}_{\mathrm{SAVE}})}\, \mathcal{O}_p(N_{r_{\min}}^{-1/2}), \tag{3.51}$$

where $\mathcal{O}_p$ denotes convergence in probability.

Proof. In the proof of Theorem 7, we showed that

$$\mathbb{E}\left[||\boldsymbol{E}||_F^2\right] \;=\; \mathcal{O}(N_{r_{\min}}^{-1}) \tag{A.82}$$

where $\boldsymbol{E} = \boldsymbol{C}_{\text{SAVE}} - \hat{\boldsymbol{C}}_{\text{SAVE}}$. Given this result, the proof for Theorem 8 is identical to the proof for Theorem 5. $\square$