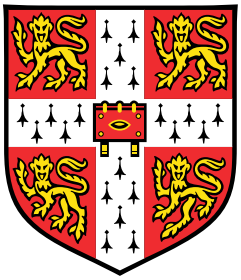
Simulation-based Design with Polynomial Ridge Approximations



Chun Yui Wong

Department of Engineering
University of Cambridge

This thesis is submitted for the degree of
Doctor of Philosophy

Jesus College

November 2021

Declaration

I declare that this thesis is the result of my own work, except where it is specified in the text and Acknowledgements. Except where specific reference is made to work of others, this thesis contains original content, and has not been submitted in whole or in part for consideration for any other degrees or qualifications in this or any other universities. It does not exceed the word and figure limits stipulated by the Degree Committee.

Chun Yui Wong
November 2021

Abstract

Simulation-based Design with Polynomial Ridge Approximations

Chun Yui Wong

Computer simulations continue to have an increasing importance in engineering design. Despite advances in computing power, simulation models that capture complex physical phenomena at high fidelities remain intractable for design tasks requiring repeated evaluations of quantities of interest. In these cases, engineers can benefit from developing design insights—a deep understanding of the underlying design space through visualisation and isolating physically significant input parameters.

In this thesis, we explore the use of orthogonal polynomial ridge approximations to construct surrogate models. These models enable rapid evaluations and facilitate a deeper, physically-intuitive understanding of the quantities of interest. For modelling smoothly varying functions, the use of orthogonal polynomials has seen considerable research that establishes its efficacy for surrogate modelling, especially in applications such as uncertainty quantification. Meanwhile, ridge approximations provide low-dimensional representations of functions via techniques from model-based dimension reduction. By expressing high-dimensional functions via a much smaller parameter space, ridge approximations enable the visualisation of functional behaviour and can massively speed up computations.

The interplay between these two classes of methods is explored through novel algorithms presented in this thesis. First, the application of polynomial ridge approximations to the sensitivity analysis of parameterised models is studied. It is shown that polynomial ridge approximations are competitive against other sparse approximation methods for evaluating moment-based sensitivity indices. Moreover, a polynomial-based set of indices—extremum Sobol' indices—is proposed for *extremum sensitivity analysis*, which is aimed at revealing input parameters responsible for driving the output near extrema. These indices are compared against those based on skewness decomposition, revealing qualitative similarities.

Following this, new methods to form ridge approximations for multi-objective (vector-valued) functions are proposed and studied. The focus of this study is on how ridge approximations of individual objectives can be combined while accounting for relations between the multiple outputs under different situations. First, assuming the underlying objectives represent a scalar field (such as a fluid flow field or finite element mesh), physical properties of the underlying field can facilitate ridge approximations of the field itself and associated quantities of interest. In this work, heuristics grouped under *embedded ridge approximations* are proposed, which allow emulators of scalar fields to be constructed with efficiency in both computational load and storage space. Using embedded ridge approximations, the storage size of the pressure field around an airfoil can be reduced by over 70% with a small mean squared error of 0.002 times the output variance.

Second, techniques for keeping multiple objectives constant through invariant subspaces are proposed. While the finding of invariant subspaces has been established for single objectives as a consequence of existing ridge approximation methods, the extension of this to multiple objectives is relatively new. Depending on the characteristics of the objectives, two different methods—the intersection approach and the vector-valued approach—are discussed. Both of these methods are then applied as part of the computational backbone for *blade envelopes*, which is a novel computational framework for designing performance-based tolerance bounds of bladed components in turbomachinery. During manufacturing and in-service use, blades inevitably suffer from geometric deviations away from design intent. Blade envelopes provide a guideline for judging whether geometric variations are likely to lead to performance degradation without explicitly solving for the resultant flow field. They are created by quantifying the statistical distribution of permissible geometries first through sampling parametric multi-objective invariant subspaces, which are then lifted into the parameter-free geometric space via the Mahalanobis distance. The resultant framework is able to accommodate multiple scalar- and vector-valued objectives, and assist the robust design of blades through enabling inverse design and visualisation.

Author contributions

The contents of this thesis are largely based on research published in the following papers:

- Wong, C. Y., Seshadri, P., Parks, G. T., Extremum sensitivity analysis with polynomial Monte Carlo filtering. *Reliability Engineering and Systems Safety* 212 (2021) 107609.

- Wong, C. Y., Seshadri, P., Parks, G. T., Girolami, M., Embedded Ridge Approximations. *Computer Methods in Applied Mechanics and Engineering* 372 (2020) 113383.

- Wong, C. Y., Seshadri, P., Scillitoe, A., Duncan, A., Ubald, B. N., Parks, G. T., Blade Envelopes. *Journal of Turbomachinery*, 144 (6) 061006 and 061007.

During the PhD, the author has also contributed to work leading to the following papers:

- Scillitoe, A., Seshadri, P., Wong, C. Y., and Duncan, A. (2021). Polynomial ridge flowfield estimation. *Physics of Fluids*, 33 (12) 127110.

- Wong, C. Y., Seshadri, P., Parks, G. T., Automatic Borescope Damage Assessments for Gas Turbine Blades via Deep Learning. In *AIAA Scitech Forum 2021*.

Acknowledgements

I would like to thank my mentors, colleagues, friends and family for their support during these years, without which this PhD would not have been possible, especially during the tough times of the pandemic. My gratitude goes to Dr Geoffrey Thomas Parks for being my supervisor at Cambridge, overseeing the journey, providing meticulous feedback for my work and lending an ear to me whenever I needed. Many thanks must also go to Dr Pranay Seshadri at Imperial College London for giving me close guidance and many fruitful research opportunities throughout my studies. Whether it was in person, on the phone or messages on Slack, he always provided me with enlightening advice whenever I was stuck on my research. It was my pleasure and honour to be able to work with Pranay. I also thank my advisor, Dr Jerome Jarrett, for giving me encouragement and feedback on my work. Thanks also go to my sponsors: the Lloyd's Register Foundation, the Alan Turing Institute, Jesus College, Cambridge and Cambridge Trust. In addition, I thank Rolls-Royce plc for sponsoring my computing equipment.

I am also grateful to be able to collaborate with folks at the Data-Centric Engineering Aeronautics group—Dr Ashley Scillitoe and Dr Bryn Noel Ubald—over many different projects, providing me with invaluable help over the technical aspects of CFD, cluster computing and 3-D rendering. Before the pandemic, I was able to enjoy working alongside many colleagues at the Engineering Design Centre: James Caleb Gross, Henry Shaowu Yuchi and Dr Sławomir Konrad Tadeja. I offer my gratitude and best wishes to them for the many conversations beside the espresso machine and over a Golden House meal, as well as their suggestions on my work.

My stay at Cambridge has been fulfilling thanks to the friends I have made over these years, starting from my undergraduate years. Specifically, I thank all my friends from the Cambridge University Chinese Orchestra Society (CUCOS) for the rehearsals, concerts, jamming and social events, and Cambridge University Anime and Manga Society (CUAMS) for the show screenings, trips to conventions, online chats and "unofficial meetings", and for all other unforgettable memories shared among us. Last but not least, I would like to thank my parents and grandparents for their unconditional support for all my endeavours.

Table of contents

List of figures

List of tables

Nomenclature

Roman Symbols

$\boldsymbol{A}$	Polynomial design matrix
$\boldsymbol{C}$	Gradient covariance matrix
$\mathbf{c}$	Polynomial expansion coefficients
C_d	Coefficient of drag
C_l	Coefficient of lift
C_p	Coefficient of pressure
$\mathcal{D}$	Input domain
d	Input dimensionality
$\mathbb{E}$	Expectation
f	Function
f_m	Mass flow function
$\mathbb{G}$	Grassmann manifold
$\boldsymbol{G}$	Gram matrix
g	Approximation function
$\boldsymbol{H}$	Vector gradient covariance matrix
$\boldsymbol{I}$	Identity matrix
$\boldsymbol{J}$	Jacobian matrix

M	Number of sample points
N	Output dimensionality
n	Number of reduced dimensions
$\mathbb{P}$	Space of polynomials
$\boldsymbol{P}$	Preconditioning squared weights matrix
P	Basis cardinality
$\boldsymbol{Q}$	Orthogonal matrix
$\mathbb{R}$	Set of real numbers
$\boldsymbol{R}$	Upper triangular matrix in QR/Weight matrix
$\boldsymbol{S}$	Tolerance covariance
s	Spatial coordinate
t	Skewness
$\mathbf{u}$	Active coordinates
$\boldsymbol{V}$	Invariant subspace matrix
v	Polynomial basis functions
$\boldsymbol{W}$	Dimension-reducing subspace matrix
$\mathbf{x}$	Input parameters
Y_p	Loss coefficient
$\mathbf{z}$	Inactive coordinates

Greek Symbols

ε	Random noise/error
λ	Quadrature points
$\boldsymbol{\Lambda}$	Eigenvalue matrix
μ	Mean

ω	Input distribution/weight function
σ	(Partial) variance
θ	(Quadrature) weights
ζ	Mahalanobis distance

Other Symbols

colspan	Column span
cond	Condition number
dist	Subspace distance
$\langle \cdot \rangle$	Inner product
L_2	Hilbert space
∇	Gradient
$\|\cdot\|_v$	The v-norm
nz	Non-zero indices
$\|$	Such that
$\subseteq$	Subset of

Acronyms / Abbreviations

BPDN	Basis pursuit denoising
CFD	Computational fluid dynamics
DoE	Design of experiments
FFD	Free-form deformation
LS	Least squares
MAVE	Minimum average variance estimation
MC	Monte Carlo
MCF	Monte Carlo filtering

MSE	Mean squared error
PCA	Principal components analysis
PCE	Polynomial chaos expansion
PDE	Partial differential equation
PDF	Probability density function
QoI	Quantity of interest
SA	Sensitivity analysis
SDR	Sufficient dimension reduction
VP	Variable projection

Chapter 1

Introduction

Computational approaches have seen a rapid rise in popularity within engineering in recent years. The use of computer-based simulation codes, such as computational structural mechanics and fluid dynamics, has played a pivotal role in emulating a wide range of physical phenomena where experimentation may be too costly or dangerous. The design of computational codes for predictive modelling and the quantification of associated uncertainties have therefore seen a great amount of research effort.

High performance computing resources have been made easier to access with recent advances in cloud computing technologies, driving down the financial cost of running large-scale simulations. However, the simulation of complicated physical systems remains challenging despite the ever growing abundance of computational resources. For example, high-fidelity computational fluid dynamics (CFD) simulations based on methods such as large eddy simulations (LES) and direct numerical simulations (DNS) may still be prohibitively expensive even with the introduction of exascale computing in the coming decade [132]. The deployment of simulation models can also be quite energy-intensive; in 2020, it was estimated that the energy usage of data-centre accounts for 1–2% of the global electricity demand, totalling up to several hundred terawatt-hours [109].

To deploy simulation models in engineering design while maintaining a reasonable computational budget, engineers often turn towards *surrogate models*. These models approximate the behaviour of complex simulations, and can be evaluated with a fraction of the original computational cost. Gains in computational speed are crucial for scenarios where models need to be deployed in real-time, such as process control. When the model needs to be evaluated many times, such as in optimisation studies, surrogate models are invaluable to drive down the computational effort and turn an intractable problem into a tractable, approximate one (see e.g. [66]).

The utility of surrogate modelling goes beyond speeding up computations. In forming approximations via curve fitting, noise arising from physical processes can be smoothed out. In some cases, approximate gradients can be furnished through the surrogate model, which is helpful in optimisation. Multi-fidelity approaches also rely on surrogate models to study the relation between simulation codes at different levels of accuracy and cost, integrating results obtained with multiple models [120].

Nevertheless, the design and deployment of surrogate models come with additional challenges. Although deep learning models constitute the state-of-the-art in pattern recognition, their performance often needs to be supported by a large amount of data. In computational engineering applications, the assumption of *big data* may not apply, owing to the expense and risk of real-world data collection. In performance-critical engineering applications, one also requires trustworthy guarantees on the reliability of the results obtained from a computational model. Uncertainty quantification (knowing how much one can trust the results of a model) and interpretability (understanding the effects of model parameters) are thus of the utmost importance.

In this thesis, we explore the use of specialised surrogate models that not only expedite computational studies, but—more importantly—reveal *design insights* pertinent to the physical systems at hand. Design insights offer a deep understanding of the underlying processes dictating the behaviour of design quantities of interest (QoIs), leading to reliable and interpretable design choices [40]. For instance, one can develop design insights through

- *Visualising* the behaviour of the QoIs in the design space, and
- Analysing the *sensitivities* of the QoIs to different design parameters.

The research presented in this thesis is thus focused on seeking design insights via two main mathematical tools, namely *orthogonal polynomials* and *ridge approximations*. The use of orthogonal polynomials has been the subject of considerable research efforts which apply them to approximate a wide range of physical phenomena [126, 146, 161]. On the other hand, dimension reduction reformulates a parametric model with a smaller set of inputs to a ridge approximation, enabling visualisation of functional behaviours as well as significant savings in computational effort.

Approximations that combine these two computational paradigms are called *polynomial ridge approximations*, a simple example of which is illustrated in Figure 1.1. From a high-dimensional space (here illustrated as a 2-D square on the left), one identifies projection directions that capture most of the functional variation using training data. These directions need not align with the cardinal axes of the original inputs. Here, one

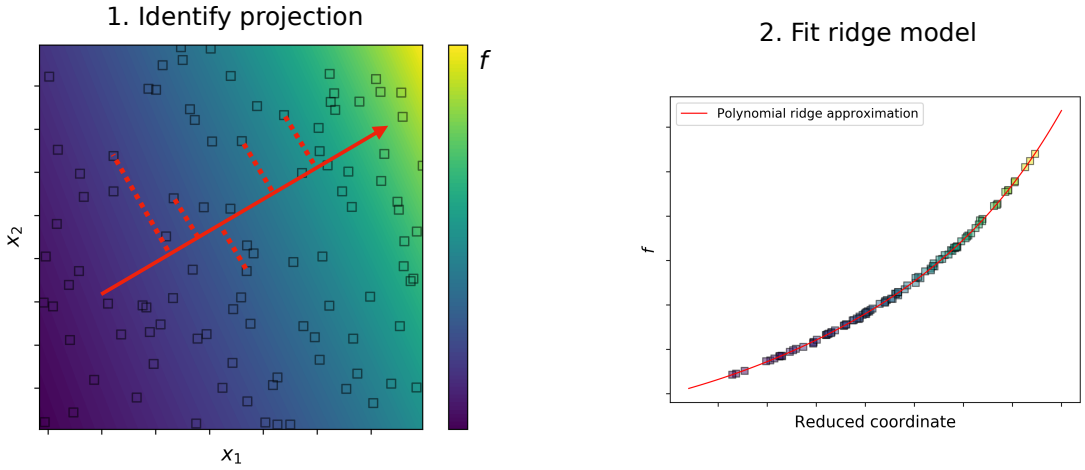


Figure 1.1 An example of a polynomial ridge approximation, which is an approximation model fitted in a low-dimensional projected space. After identifying a projection direction (shown as the red arrow on the left figure), data points are projected onto that direction, and a polynomial in the projected space can be fitted (shown on the right).

direction, indicated by the red arrow, can be identified by inspection to contain the majority of the functional variation; indeed, the function is approximately constant when going at a direction perpendicular to the red arrow. Projecting the data to this direction orthogonally yields the right plot, which is an instance of a *sufficient summary plot*. A polynomial can then be fitted in this projected space, shown by the red curve on the right plot. As will be explained in Chapter 2, the process of fitting this polynomial is greatly facilitated by the lowered dimensionality of the projected space when compared with the original input space.

In this thesis, polynomial ridge approximations form the foundations of algorithms that can be applied to form end-to-end processes, providing intuitive and quantitative understanding of the design space as well as an efficient and automatic way to make design decisions.

1.1 Research contributions

The main contributions of this thesis consist of novel algorithms for computational engineering based on polynomial ridge approximations. They revolve around the construction of surrogate models that not only brings efficiency in computation, but also confer understanding of the design parameter space and facilitate the development of design insights. Topics addressed in the subsequent chapters are briefly summarised below.

1.1.1 Sensitivity analysis with polynomial ridges

Sensitivity analysis involves understanding how input parameters influence QoIs. By isolating parameters that most strongly affect the output, one can reduce the complexity of a computational model by removing parameters that do not cause large output variations. The reduced set of parameters can also inform engineers of the dominant physical phenomena that are critical to design. However, for models with a large number of parameters, the calculation of sensitivities can be intractable due to the need to evaluate high-dimensional integrals [144].

It has been shown [145] that orthogonal polynomial models can be used to emulate a model's outputs through its input parameters. Using polynomials, sensitivity indices can be calculated efficiently from the expansion coefficients. The main challenge of this approach is the potentially high cost of constructing the polynomial that accurately describes the QoI. In this part of the thesis, ridge approximations are used to discover and exploit low-dimensional structures present in the model. This enables efficient polynomial approximations and calculations of sensitivity indices.

1.1.2 Extremum Sensitivity analysis

Engineering systems are often operated in regions where the output is at an extremum. Therefore, it is beneficial to know which input parameters contribute the most to driving the output near extrema. Methods under *extremum sensitivity analysis* are proposed by adapting methods of sensitivity analysis to input regions that yield extreme outputs. A new set of indices called *extremum Sobol' indices* is introduced to compute these sensitivities efficiently using polynomial ridge functions. They are compared with skewness-based sensitivity indices and the connections between the two sets of indices are explored.

1.1.3 Multi-objective ridge approximations

A great number of design decisions in engineering are made through trade-offs between multiple objectives. In this thesis, we study approaches within *multi-objective ridge approximations*, which are able approximate and control the variation of more than one function of parameters. These approaches fall into two paradigms, listed below:

- **Multi-objective invariance**: Given multiple outputs dependent on a common set of input parameters, how can one discover subspaces within the input domain where outputs are kept approximately constant? Depending on the nature of the set of output objectives, two approaches, namely the *intersection approach* and *vector-valued dimension reduction*, are described. While the former offers stricter control over a small set of outputs, the latter can be used for more flexible but approximate control over a larger number of objectives.
- **Embedded ridge approximations**: Parameterised scalar fields are used frequently in the modelling of continuous physical systems, such as a fluid flow field or an elastic structure, under the influence of certain inputs. Discretising the field spatially, one can treat the field as a set of output objectives dependent on a set of parameters. We call ridge approximations formed on field values defined on the computational mesh *embedded ridge approximations*. Novel algorithms that formulate and leverage embedded ridge approximations are proposed, which show that under smoothness assumptions of the underlying scalar field, surrogate models of the field and related QoIs can be formed efficiently.

1.1.4 Blade envelopes

Manufacturing processes and in-service degradations can cause bladed components in jet engines to deviate from the design intent geometrically. Depending on the location, size and shape of these deviations, the repercussions on aerodynamic performance can vary greatly. When presented with the scan of a deformed geometry, how can an engineer determine whether the blade maintains performance standards? *Blade envelopes* quantify the demarcation between geometries which can be used and those which need to be scrapped by defining a data-driven statistical distribution in the space of deformed geometries. Going beyond pointwise tolerance bounds, blade envelopes capture the geometric characteristics of acceptable geometries through a statistical covariance. They are supported by computational strategies for multi-

objective invariance to rigorously identify subspaces containing geometries that are invariant in more than one QoI. As a result, they furnish useful tools for the robust design of turbomachinery blades, such as a scrap-or-use confidence score for scanned geometries, and key manufacturing modes that indicate locations on a blade that one should pay attention to during a scan. The result is an end-to-end system that can be used by both manufacturing and design engineers to understand and draft tolerance guidelines.

1.2 Outline of thesis

To preface the development of the aforementioned topics, the thesis starts with an introduction of the mathematical foundations of orthogonal polynomials and ridge approximations in Chapter 2. Following this, sensitivity analysis using orthogonal polynomials is introduced in Chapter 3, leading to the topics of using polynomial ridges for sensitivity analysis and extremum sensitivity analysis.

Chapter 4 starts with explaining the theory and algorithms of multi-objective ridge approximations, followed by introducing embedded ridge approximations and demonstrating them to approximate a flow field around an airfoil. In Chapter 5, the statistical methodology behind blade envelopes is explained. The framework is tested on a test case involving deformations of the von Karman Institute LS89 turbine profile.

Finally, Chapter 6 summarises the developments made within this work and recommends directions for further research. Figure 1.2 shows the connections between the different topics covered in this thesis.

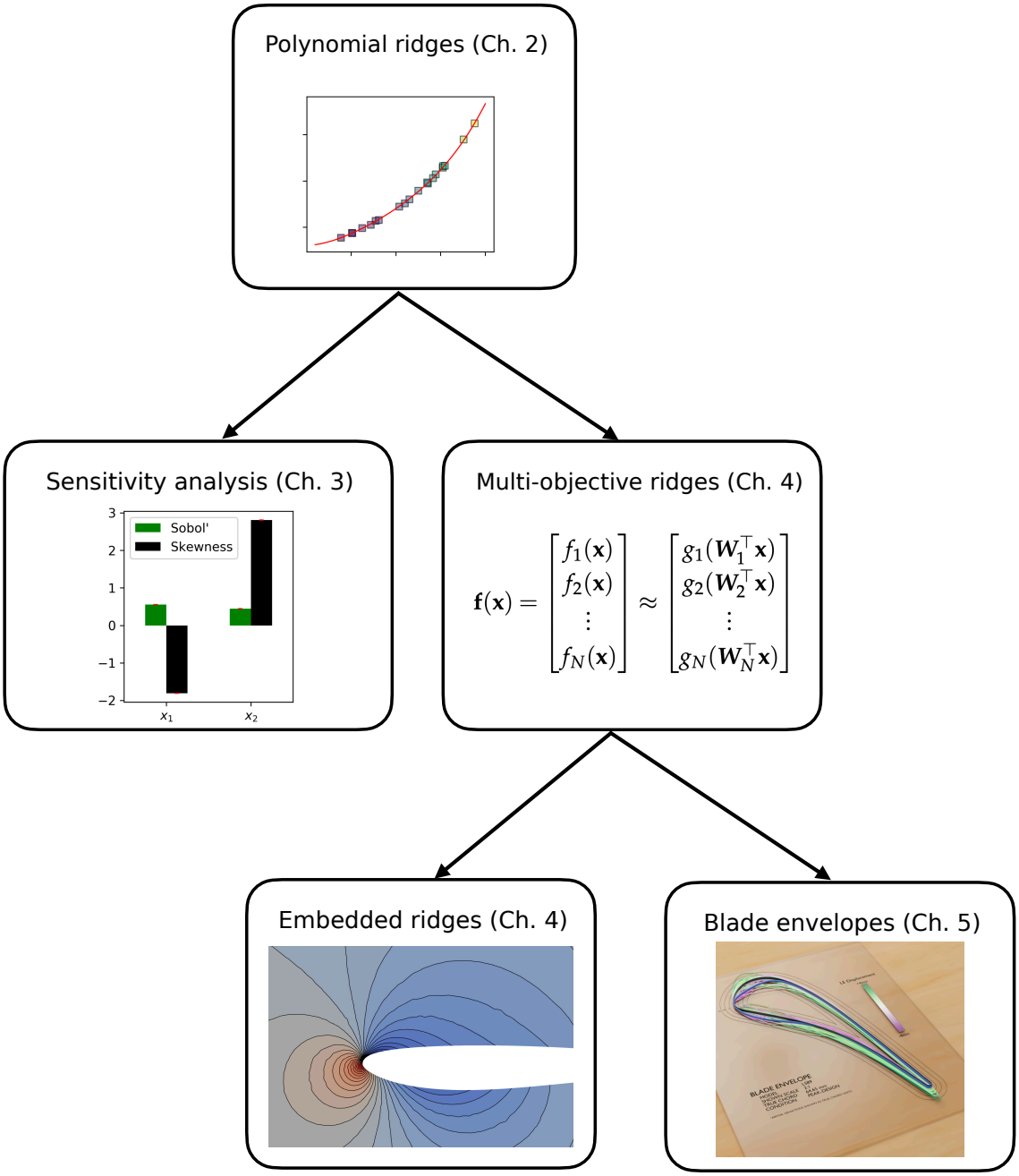


Figure 1.2 Roadmap of topics covered in this thesis.

Chapter 2

Mathematical preliminaries

In this chapter, we review ideas and algorithms within the areas of polynomial approximations and model-based dimension reduction, capturing recent developments in addition to classical underpinnings. These methods serve as the foundation which applications in subsequent chapters are built upon.

2.1 Polynomial Approximations

A ubiquitous task in simulation-based engineering is the construction of approximations for the complex model at hand. An effective approximation model serves as a surrogate which facilitates uncertainty quantification, optimisation, reliability analysis and more. In this section, we describe the framework of polynomial approximations, largely following the notation from [70].

Before proceeding with the exposition, it is worthwhile to motivate the use of polynomials. As an example, take the task of evaluating the output mean of a model under input uncertainty. Figure 2.1 studies the average error in estimating the means of univariate functions whose inputs are characterised by uniform uncertainties, comparing two different approaches:

- A Monte Carlo approach, where the functions are evaluated on a randomly sampled set of points, and the mean is estimated by the ensemble mean, and
- First approximating the functions with an orthogonal polynomial series evaluated on Gauss quadrature points, then inferring the statistics of the output from the expansion coefficients.

The results clearly show that using quadrature with a polynomial approximation yields much more rapid convergence in accuracy than Monte Carlo as the number of

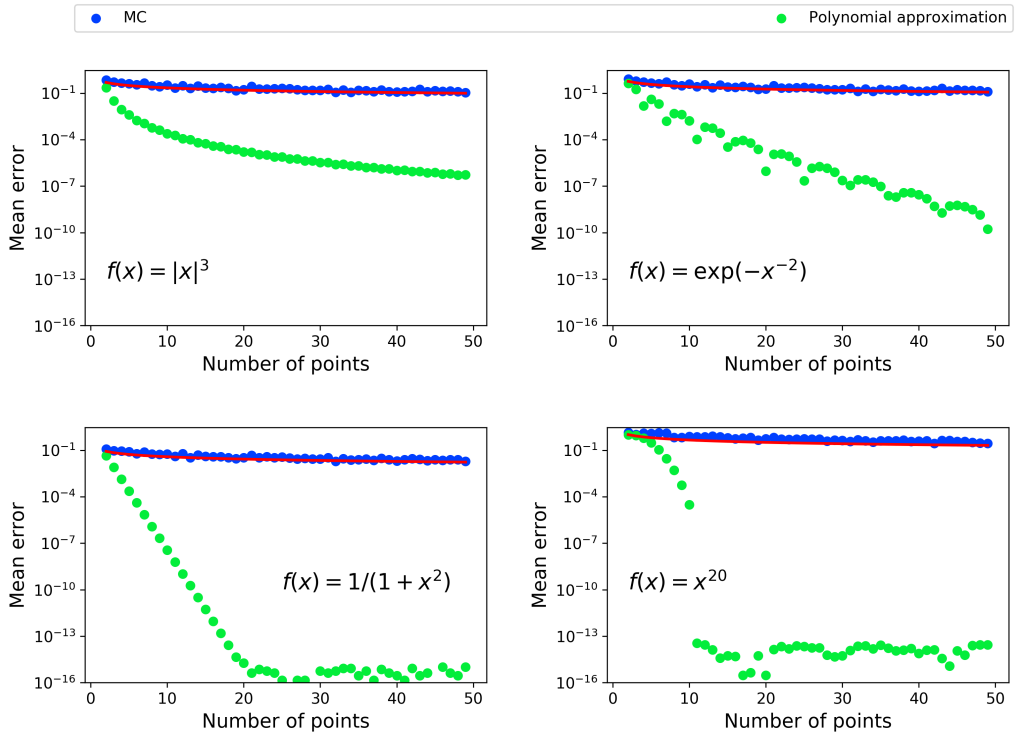


Figure 2.1 Comparing the numerical estimation accuracy of output mean of four exemplar 1-D functions using Monte Carlo (MC) and a polynomial approximation (Legendre polynomial chaos expansion with Gauss quadrature, see Section 2.1.1 and Section 2.1.2.1 for definitions of these terms). The red curve shows a $\sim 1/\sqrt{M}$ trend where M is the number of points used. The maximum order of polynomials used is equal to $M - 1$.

points increases. The latter obeys an inverse square root rate of convergence, while the former can often achieve exponential rates [27]. Other advantages of using a polynomial model include readily interpretable coefficients, e.g. in the context of sensitivity analysis, as we elaborate in Chapter 3.

We begin by representing a parameterised computational model as a function $f : \mathcal{D} \to \mathbb{R}$ where $\mathcal{D} \subseteq \mathbb{R}^d$ is the domain of the *d input parameters* $\mathbf{x} = [x_1, x_2, ..., x_d]$. Assume that this function resides in the Hilbert space[1] of ω-weighted square-integrable functions

$$L_2(\omega) = \left\{ f : \mathcal{D} \to \mathbb{R} \mid \int_{\mathcal{D}} f(\mathbf{x})^2 \, \omega(\mathbf{x}) \, d\mathbf{x} < \infty \right\}, \tag{2.1}$$

[1]Definitions of underlined terms in this thesis can be found in Appendix A.

with the weight function $\omega(\mathbf{x})$ satisfying

$$\int_{\mathcal{D}} \omega(\mathbf{x})\, d\mathbf{x} = 1, \quad \omega(\mathbf{x}) > 0 \text{ for all } \mathbf{x} \in \mathcal{D}. \tag{2.2}$$

Note that analogous formulations for discrete weight functions can be defined (for example, see Section 2.2 of [160]), but we assume the use of continuous weight functions in this thesis. The function f itself does not have to be continuous, though additional assumptions may be placed on f in subsequent chapters. For $u, v \in L_2(\omega)$, the inner product and norm of the Hilbert space can be defined as

$$\langle u, v \rangle = \int_{\mathcal{D}} u(\mathbf{x})\, v(\mathbf{x})\, \omega(\mathbf{x})\, d\mathbf{x}, \qquad \|u\|^2_{L_2(\omega)} = \langle u, u \rangle. \tag{2.3}$$

Any function f in the Hilbert space $L_2(\omega)$ admits an infinite series expansion of basis polynomials $\{v_i(\mathbf{x})\}$

$$f(\mathbf{x}) = \sum_{i=1}^{\infty} c_i v_i(\mathbf{x}) \tag{2.4}$$

satisfying the orthonormal relations

$$\int_{\mathcal{D}} v_k(\mathbf{x})\, v_l(\mathbf{x})\, \omega(\mathbf{x})\, d\mathbf{x} = \begin{cases} 1 & \text{if } k = l, \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

Using orthonormality, the coefficients c_i are given by

$$c_i = \langle f, v_i \rangle = \int_D f(\mathbf{x})\, v_i(\mathbf{x})\, \omega(\mathbf{x})\, d\mathbf{x}, \tag{2.6}$$

which can be interpreted as the projection of f onto the direction of v_i. The goal of polynomial approximation is to find a polynomial function with $P < \infty$ terms that approximates the function $f(\mathbf{x})$ well. Consider the truncation of the infinite series (2.4),

$$f_P(\mathbf{x}) = \sum_{i=1}^{P} c_i v_i(\mathbf{x}). \tag{2.7}$$

With c_i defined as in (2.6), it can be shown that f_P is the function in the subspace $V \subset L_2(\omega)$

$$V = \text{span}(v_1, v_2, ..., v_P) \tag{2.8}$$

which minimises the $L_2(\omega)$ error [70]

$$f_P = \underset{g \in V}{\operatorname{argmin}} \|f - g\|_{L_2(\omega)}. \tag{2.9}$$

The resultant $L_2(\omega)$ error is the *truncation error* of the approximation when the infinite series is truncated to P terms. To find the approximation in practice, the following must be answered:

- What is the choice for the weight function $\omega(\mathbf{x})$ and basis functions $v_i(\mathbf{x})$?
- How do we decide on the value of P and which P basis functions to use?
- How can we solve for the coefficients c_i accurately while minimising the number of function evaluations?

In subsequent sections, we address these questions by reviewing existing approaches from literature.

2.1.1 Choice of the weight function and basis

The weight function $\omega(\mathbf{x})$ takes on a natural interpretation in the context of uncertainty quantification. Here, $\mathbf{x}$ is treated as a random variable defined on the input probability space $(\mathcal{D}, \mathcal{F}, \mathbb{P})$ with $\mathcal{F}$ being the Borel σ-algebra on $\mathcal{D}$, and $\mathbb{P}$ being the input probability measure admitting a probability density function $\omega(\mathbf{x})$. In this setting, (2.4) is also known as a generalised *polynomial chaos expansion* (PCE) of the random variable $f(\mathbf{x})$. For this application, $\omega(\mathbf{x})$ is also known as the *input probability density function*. Outside of uncertainty quantification, weight functions can encode relative importance of different parts of the input domain in approximation, especially in the case of infinite domains. Since the choice of $\omega(\mathbf{x})$ dictates the basis polynomials used through the constraints in (2.5), certain choices of weight functions can also lead to special basis functions which have desirable algorithmic qualities, such as the Chebyshev polynomials corresponding to the Chebyshev distribution $\omega(x) = 2/(\pi\sqrt{1-x^2})$ (see [151] and the discussion on least squares strategies in Section 2.1.2).

So, how can one construct the basis polynomials $v_i(\mathbf{x})$ given $\omega(\mathbf{x})$? We start with the case of $d = 1$, i.e. basis polynomials and weight function defined over a univariate parameter x. It is well-known (see [55]) that all univariate orthogonal polynomials obey a three-term recurrence relation

$$\beta_{i+1} v_{i+1}(x) = (x - \alpha_i) v_i(x) - \beta_i v_{i-1}(x) \tag{2.10}$$

for $i = 0, 1, 2, ...$ with a set of recurrence coefficients (α_i, β_i) fully determined by $\omega(x)$. Example univariate polynomial bases and their corresponding input distributions obeying (2.5) include

- the Legendre polynomials

$$v_i(x) = \frac{\sqrt{2i+1}}{2^i i!} \frac{d^i}{dx^i}(x^2 - 1)^i, \tag{2.11}$$

  which are orthogonal to the uniform distribution on $[-1, 1]$,

$$\int_{-1}^{1} v_k(x)\, v_l(x)\, \frac{1}{2}\, dx = \begin{cases} 1 & \text{if } k = l, \\ 0 & \text{otherwise.} \end{cases} \tag{2.12}$$

- the Hermite polynomials

$$v_i(x) = \frac{(-1)^i}{\sqrt{i!}} \mathrm{e}^{\frac{x^2}{2}} \frac{d^i}{dx^i} \mathrm{e}^{\frac{-x^2}{2}}, \tag{2.13}$$

  which are orthogonal to the normal distribution,

$$\int_{-\infty}^{\infty} v_k(x)\, v_l(x)\, \frac{1}{\sqrt{2}} \mathrm{e}^{\frac{-x^2}{2}}\, dx = \begin{cases} 1 & \text{if } k = l, \\ 0 & \text{otherwise.} \end{cases} \tag{2.14}$$

- the Laguerre polynomials

$$v_i(x) = \frac{\mathrm{e}^x}{i!} \frac{d^i}{dx^i}(\mathrm{e}^{-x} x^i) \tag{2.15}$$

  which are orthogonal to the exponential distribution,

$$\int_{0}^{\infty} v_k(x)\, v_l(x)\, \mathrm{e}^{-x}\, dx = \begin{cases} 1 & \text{if } k = l, \\ 0 & \text{otherwise.} \end{cases} \tag{2.16}$$

The recurrence coefficients can be found via algorithms such as the Stieltjes procedure and modified Chebyshev algorithm [55]. Using (2.10), orthogonal polynomials at any degree can then be evaluated.

Now consider the case of $d > 1$. If the input parameters are independent, implying that the input domain is a (possibly non-compact) hypercube that can be decomposed

as the Cartesian product of 1-D (possibly unbounded or semi-bounded) intervals

$$\mathcal{D} = \mathcal{D}_1 \times \mathcal{D}_2 \times \cdots \times \mathcal{D}_d, \tag{2.17}$$

one can factorise the probability density $\omega(\mathbf{x})$ into marginal distributions for each parameter

$$\omega(\mathbf{x}) = \prod_{i=1}^{d} \omega_i(x_i). \tag{2.18}$$

In this case, orthogonal polynomials can be formed from the product of univariate polynomials

$$v_k(\mathbf{x}) = \prod_{i=1}^{d} v_{k_i}^{(i)}(x_i) \tag{2.19}$$

where k_i is the degree of v_k in the i-th direction. The univariate orthogonal polynomials obey the same orthonormal constraints described in (2.5) with respect to $\omega_i(x_i)$ for $i = 1, ..., d$. Thus, each polynomial index k corresponds to a *multi-index* $(k_1, k_2, ..., k_d)$ defined via (2.19) denoting the degree of the polynomial in each dimension. The combined set of multi-indices corresponding to each $k = 1, ..., P$ is called an *index set*, which indicates which polynomial basis functions are included in the approximation. An *isotropic* index set is one where the maximal degrees of polynomials used in each dimension are equal. Common isotropic index sets include

- the tensor grid index set: $\{(k_1, k_2, ..., k_d) \mid k_i \leq p \text{ for all } i = 1, ..., d\}$,
- the total order index set: $\{(k_1, k_2, ..., k_d) \mid \sum_i k_i \leq p\}$,
- the hyperbolic index set: $\{(k_1, k_2, ..., k_d) \mid \left(\sum_i k_i^q\right)^{1/q} \leq p\}$ for $0 < q < 1$,
- the Euclidean degree index set: $\{(k_1, k_2, ..., k_d) \mid \sqrt{\sum_i k_i^2} \leq p\}$.

Figure 2.2 illustrates the elements of the above index sets for $d = 2$ and $p = 10$, showing their relative sizes. A smaller index set implies a smaller set of coefficients to estimate, implying a smaller cost of approximation. However, a larger index set is able to capture higher-order interaction terms which can reduce the approximation error, but has more coefficients that need to be estimated. The choice of index set is thus application dependent. There are some works that argue for the choice of certain index sets (e.g. see [152] for the motivation behind the Euclidean degree index set). In addition, there are applications (e.g. formulation of sparse grid index sets, see [13, 34] and the end of Section 2.1.2.1) which require the use of anisotropic index sets, but we

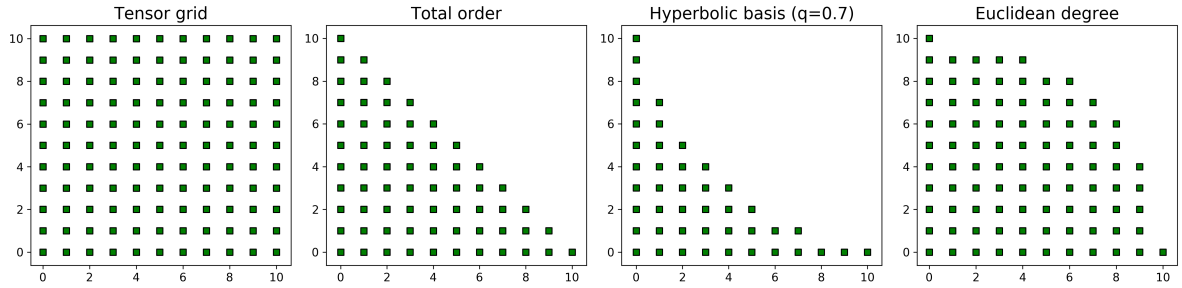


Figure 2.2 Elements of four example index sets with $d = 2$ and $p = 10$.

leave a detailed discussion outside the scope of this thesis since the ideas presented here are independent of the basis chosen.

When the assumption of independence between input variables cannot be made (i.e. the input distribution cannot be factorised as in Equation (2.18)), an orthogonal polynomial basis can still be constructed. In the context of this work, input correlations occur when constructing polynomials over non-rectangular domains. Starting from the basis functions corresponding to the distribution ignoring correlations, the Gram-Schmidt process can be used to generate the orthogonal basis while factoring in correlations. More details on this are given in Chapter 3. Note that through generalising polynomial approximations to multiple outputs, potential gains in computational efficiency can be obtained by exploiting the correlations between different output components. This will be explored in Chapter 4.

2.1.2 Solving for the coefficients

Once the polynomial basis is fixed, the approximation task can be reduced to solving for the polynomial coefficients c_i. In practice, this must be done with a finite number of function evaluations at certain designated points in the input domain. We group these into a set of M *input-output pairs*

$$\mathcal{S} = \left\{ \left(\mathbf{x}^{(1)}, f(\mathbf{x}^{(1)})\right), \left(\mathbf{x}^{(2)}, f(\mathbf{x}^{(2)})\right), ..., \left(\mathbf{x}^{(M)}, f(\mathbf{x}^{(M)})\right) \right\}, \tag{2.20}$$

where each output point $f(\mathbf{x}^{(i)})$ is a scalar. Since the polynomial expansion (2.7) can be expressed as the sum of polynomial basis functions multiplied by unknown coefficients, the coefficients can be found by solving a linear system

$$\sqrt{\boldsymbol{P}}\boldsymbol{A}\mathbf{c} = \sqrt{\boldsymbol{P}}\mathbf{f} \tag{2.21}$$

where

- the weight matrix $\boldsymbol{P} \in \mathbb{R}^{M\times M}$ is a diagonal matrix with positive entries, which can be used to improve the stability of the problem as will be explained in this section;
- the polynomial design matrix $\boldsymbol{A} \in \mathbb{R}^{M\times P}$ contains evaluations of the polynomial basis terms;

$$\boldsymbol{A} = \begin{bmatrix} v_1(\mathbf{x}^{(1)}) & v_2(\mathbf{x}^{(1)}) & \dots & v_P(\mathbf{x}^{(1)}) \\ v_1(\mathbf{x}^{(2)}) & v_2(\mathbf{x}^{(2)}) & \dots & v_P(\mathbf{x}^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ v_1(\mathbf{x}^{(M)}) & v_2(\mathbf{x}^{(M)}) & \dots & v_P(\mathbf{x}^{(M)}) \end{bmatrix}. \tag{2.22}$$

- the coefficients vector $\mathbf{c} \in \mathbb{R}^P$ contains the polynomial coefficients to be found;
- the function evaluations vector $\mathbf{f} \in \mathbb{R}^M$ contains the outputs from the input-output pairs.

In the following, three different coefficient solution strategies are described as variations on this theme.

2.1.2.1 Pseudospectral approximation

Pseudospectral approximation approaches are based on numerical evaluation of the integrals in (2.6) via a quadrature rule,

$$\begin{aligned} c_k &= \int_D f(\mathbf{x})\, v_i(\mathbf{x})\, \omega(\mathbf{x})\, d\mathbf{x} \\ &\approx c_{k,PSC} = \sum_{j=1}^{Q} f(\lambda^{(j)})\, v_k(\lambda^{(j)})\, \theta^{(j)} \end{aligned} \tag{2.23}$$

where $(\theta^{(j)}, \lambda^{(j)})$ are the Q quadrature weights and points that are derived from the quadrature rule used. A correspondence can be established with the linear system formulation (2.21) by substituting the evaluation points $\mathbf{x}^{(j)} = \lambda^{(j)}$ and the weights $\boldsymbol{P} = \mathrm{diag}(\theta^{(j)})$. Define the *exact set* of the quadrature rule as the set of functions that can be integrated exactly by the rule. Assuming that the exact set includes all pairwise products of the polynomial basis functions, one can assert that

$$(\sqrt{\boldsymbol{P}}\boldsymbol{A})^\top \sqrt{\boldsymbol{P}}\boldsymbol{A} = \boldsymbol{I} \tag{2.24}$$

because the entries can be expressed as $L_2(\omega)$ inner products of basis polynomials via a quadrature formula in the form of (2.23). The discrete least squares solution[2] to the linear system (2.21) is thus

$$\mathbf{c}_{PSC} = \left((\sqrt{\boldsymbol{P}}\boldsymbol{A})^\top \sqrt{\boldsymbol{P}}\boldsymbol{A}\right)^{-1} (\sqrt{\boldsymbol{P}}\boldsymbol{A})^\top \sqrt{\boldsymbol{P}}\mathbf{f} = \boldsymbol{A}^\top \boldsymbol{P}\mathbf{f}, \tag{2.25}$$

which is equivalent to (2.23).

Common choices of quadrature rules in 1-D come from the Gauss quadrature rules and their variations, such as the Gauss-Radau and Gauss-Lobatto rules. For semi-bounded integration domains ($[a, \infty)$ or $(-\infty, a]$ where $-\infty < a < \infty$), Gauss-Radau quadrature includes the finite endpoint of the interval; for bounded domains, Gauss-Lobatto quadrature includes both endpoints (see [55] and references therein for further details). Gauss quadrature points and weights can be obtained using an eigendecomposition of a tri-diagonal matrix with entries composed of recurrence coefficients of the orthogonal polynomial system [61]. Gauss quadrature rules are optimal in the degree of exactness (the highest degree of polynomials that can be integrated exactly) obtained for a fixed number of points—integrals of polynomial functions up to degree $2q - 1$ can be evaluated exactly with q points. Higher degrees of exactness can be achieved with more points via the Gauss-Kronrod and Gauss-Patterson rules. Gauss-Turán quadrature extends this with gradient observations. Beyond Gauss quadrature, the Clenshaw-Curtis rule is another popular quadrature rule. Though it achieves a lower degree of exactness than Gauss rules for a given number of points theoretically, in practice it has only a small difference in accuracy when compared to Gauss quadrature [150]. The advantage of the Clenshaw-Curtis rule is the efficiency in computing its weights via the fast Fourier transform without the need for an eigendecomposition.

The degree of exactness of orthogonal polynomials is closely tied to the approximation accuracy of pseudospectral methods. Conrad and Marzouk [25] term the error committed by numerical quadrature on the product of polynomial basis functions the *aliasing error*. Recall that f can be expressed as an infinite Fourier series from (2.4) and

[2] When $Q = P$, this is equivalent to the unique interpolatory solution.

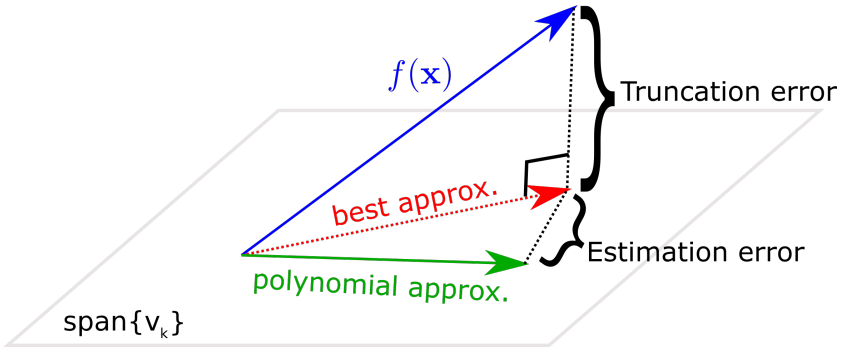


Figure 2.3 Errors incurred by practical polynomial approximation methods.

substitute that into $f(\lambda^{(j)})$ in (2.23) to yield

$$\begin{aligned} c_{k,PSC} &= \sum_{j=1}^{Q} \left(\sum_{i=1}^{\infty} c_i v_i(\lambda^{(j)}) \right) v_k(\lambda^{(j)})\, \theta^{(j)} \\ &= \sum_{i=1}^{P} c_i \underbrace{\left(\sum_{j=1}^{Q} v_i(\lambda^{(j)})\, v_k(\lambda^{(j)})\, \theta^{(j)} \right)}_{\text{internal}} + \sum_{l=P+1}^{\infty} c_i \underbrace{\left(\sum_{j=1}^{Q} v_l(\lambda^{(j)})\, v_k(\lambda^{(j)})\, \theta^{(j)} \right)}_{\text{external}}. \end{aligned} \tag{2.26}$$

The first term on the second line involves quadrature approximations of the inner products containing polynomials within the basis, while the second term involves polynomials outside of the basis. With a sufficient number of quadrature points, the bracketed expression in the first term can be driven to zero for $i \neq k$ and one otherwise, depending on the quadrature degree of exactness. When this is not true, the first term is no longer exactly equal to c_k and the difference is referred to as an *internal aliasing error*.

The second term involves polynomials of arbitrarily high degree, implying that it cannot be driven to zero unless $P \to \infty$. This is called an *external aliasing error*, and causes the pseudospectral approximation to have an additional source of deviation from the true function f on top of the truncation error. Summarising this briefly (also see Figure 2.3), the approximation error can be split into

$$\text{Approximation error} = \text{Coefficient estimation error} + \text{Truncation error}, \tag{2.27}$$

where the coefficient estimation error refers to the aliasing errors in the context of pseudospectral methods.

As an example, consider a univariate Legendre polynomial basis of maximum degree 5. Using six Gauss-Legendre quadrature points and weights, the degree of exactness is 11, which is sufficient to integrate all inner products of polynomials in the basis exactly. Thus, no internal aliasing error is committed. However, when integrating polynomials of higher degree, external aliasing errors can exist. Figure 2.4 shows a heatmap plot of the Gram matrix

$$\boldsymbol{G} = (\sqrt{\boldsymbol{P}}\boldsymbol{A})^\top (\sqrt{\boldsymbol{P}}\boldsymbol{A}) \tag{2.28}$$

where $\boldsymbol{A}$ contains evaluations of Legendre polynomials up to degree 7, two higher than the original basis, at quadrature points. Note that the (a, b) entry of $\boldsymbol{G}$ is given by

$$G_{ab} = \sum_{j=1}^{6} v_a(\lambda^{(j)})\, v_b(\lambda^{(j)})\, \theta^{(j)}, \tag{2.29}$$

which is the quadrature approximation of the inner product between the polynomials v_a and v_b by Gauss quadrature. The red line indicates all polynomials that can be integrated exactly. Indeed, within the top-left of the red line, all off-diagonal entries are exactly zero (up to machine precision), and all diagonal entries are unity. Because of the degree of exactness of Gauss quadrature, the internal aliasing error is zero. Beyond the red line, this is no longer true because the combined degree of the integrand is greater than that obtained by calculating the degree of exactness. This results in external aliasing errors, which manifest as deviations from the identity in the Gram matrix, highlighted by cyan boxes in Figure 2.4.

When the input density is separable into a product of marginals, multivariate quadrature rules for tensor grid polynomial bases can be constructed from the tensorisation of 1-D quadrature rules, each dependent on the marginal density and the maximum degree of polynomials in that dimension. That is, the set of quadrature points is the Cartesian product of 1-D quadrature points, and the set of corresponding weights the product of 1-D quadrature weights in each dimension. This immediately brings one major limitation to light: the exponential scaling of the number of points required as the dimension increases. Specifically, $(p+1)^d$ evaluations are needed where p is the highest polynomial degree, which is too expensive even for modest d on the order of 10. The development of efficient numerical integration methods, from sparse grids [13, 34] to specialised quadrature rules for custom grids [81], allows the construction of quadrature rules on smaller index sets whose sizes scale sub-exponentially with input dimensions. However, another limitation remains: the

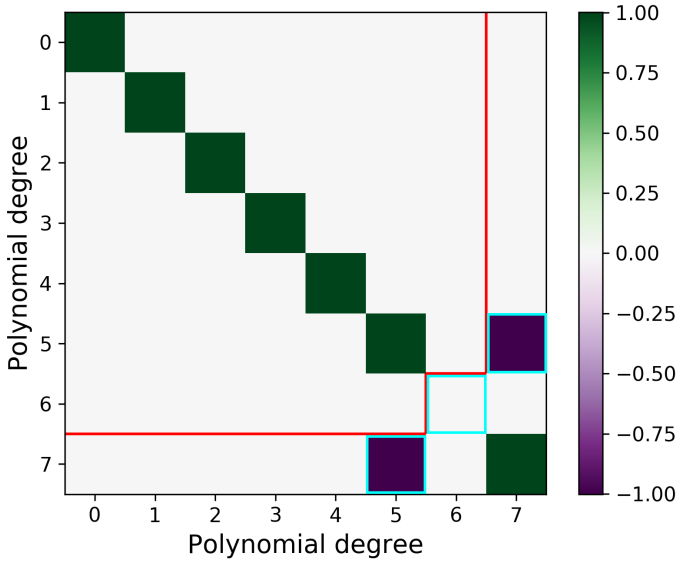


Figure 2.4 Gram matrix plot of a Legendre basis of maximum degree 7, using 6 Gauss-Legendre points and weights. The region top-left of the red line contains entries that can be exactly integrated by the quadrature rule. Cyan boxes show deviations from the identity matrix, indicating external aliasing errors.

requirement for the points at which the model is evaluated to be at specific quadrature points. This poses a challenge for problems where control over the data gathering process is difficult, such as the analysis of systems through sensor data collected from routine operation.

2.1.2.2 Least squares approximation

In lieu of numerical integration rules, a separate class of methods leverages alternative sampling strategies for constructing the input-output dataset. While sparse grids can be used to abate the exponential scaling of the number of sample points with input dimension, they are limited to specific index sets dictated by their growth rates [134]. Least squares approximations offer a larger degree of flexibility with respect to the choice of polynomial basis and sample points. Upon fixing the input points $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(M)}$ as sample points from a certain distribution or otherwise, the least squares coefficients are given by

$$\mathbf{c}_{LS} = \underset{\mathbf{c}\in\mathbb{R}^P}{\operatorname{argmin}} \left\| \sqrt{\boldsymbol{P}}\boldsymbol{A}\mathbf{c} - \sqrt{\boldsymbol{P}}\mathbf{f} \right\|_2 . \tag{2.30}$$

Supposing that $M \geq P$ and $\sqrt{\boldsymbol{P}}\boldsymbol{A}$ is full rank up to machine precision—i.e. the first P singular values of $\sqrt{\boldsymbol{P}}\boldsymbol{A}$ must be significantly larger than machine precision—the least squares solution can be given analytically by the normal equations,

$$\mathbf{c}_{LS} = \left((\sqrt{\boldsymbol{P}}\boldsymbol{A})^\top \sqrt{\boldsymbol{P}}\boldsymbol{A}\right)^{-1} (\sqrt{\boldsymbol{P}}\boldsymbol{A})^\top \sqrt{\boldsymbol{P}}\mathbf{f}, \tag{2.31}$$

and can be found by taking the QR factorisation of $\sqrt{\boldsymbol{P}}\boldsymbol{A}$.

Asymptotically, it can be shown that sampling according to the input distribution and setting $\boldsymbol{P} = \frac{1}{M}\boldsymbol{I}$, the least squares coefficients converge to the projection coefficients in (2.6),

$$\lim_{M\to\infty} c_{k,LS} = c_k, \tag{2.32}$$

for $k = 1, ..., P$ [70]. However, in practice we are interested in the accuracy of the least squares approximation in the pre-asymptotic regime; how can accurate least squares approximations be furnished with a limited computational budget? Certain choices of sample points may lead to instability as the polynomial degree increases. For example, approximations on equally spaced points (Newton-Cotes points) leads to the Runge phenomenon, creating large oscillations in the polynomial approximant. In [23, 3] and other works, authors draw attention to the proximity of the Gram matrix $\boldsymbol{G}$ (see (2.28)) to the identity $\boldsymbol{I}$ in the operator norm

$$\|\boldsymbol{G} - \boldsymbol{I}\|_2 \tag{2.33}$$

as an objective to minimise. One motivation behind this is its connection with the condition number of $\boldsymbol{G}$ [24],

$$\|\boldsymbol{G} - \boldsymbol{I}\|_2 \leq \delta \Rightarrow \text{cond}(\boldsymbol{G}) \leq \frac{1+\delta}{1-\delta}, \tag{2.34}$$

for $0 < \delta < 1$. Instead of sampling with respect to the input distribution, one can sample with respect to another distribution $\rho(\mathbf{x})$, where

$$\rho(\mathbf{x}) = q(\mathbf{x})^2 \omega(\mathbf{x}), \qquad \int_{\mathcal{D}} q(\mathbf{x})^2\, \omega(\mathbf{x})\, d\mathbf{x} = 1. \tag{2.35}$$

To retrieve the asymptotic convergence of the least squares solution, the weights are set as $\boldsymbol{P}_{mm} = \frac{1}{Mq(\mathbf{x}^{(m)})^2}$. In [23], it is shown that if the number of samples M satisfies

$$\frac{M}{P \log M} \geq C(1+r) \sup_{\mathbf{x} \in \mathcal{D}} \sum_{i=1}^{N} \left(\frac{v_i(\mathbf{x})}{q(\mathbf{x})} \right)^2, \tag{2.36}$$

where $C = 2/\log(27/8\mathrm{e})$ where $\mathrm{e} \approx 2.718$ is Euler's number for a specific $r > 0$, then

$$\Pr(\|\boldsymbol{G} - \boldsymbol{I}\|_2 \leq 0.5) \geq 1 - 2M^{-r}. \tag{2.37}$$

Thus, for a given budget M the goal is to minimise the quantity

$$\sup_{\mathbf{x} \in \mathcal{D}} \sum_{i=1}^{N} \left(\frac{v_i(\mathbf{x})}{q(\mathbf{x})} \right)^2 \tag{2.38}$$

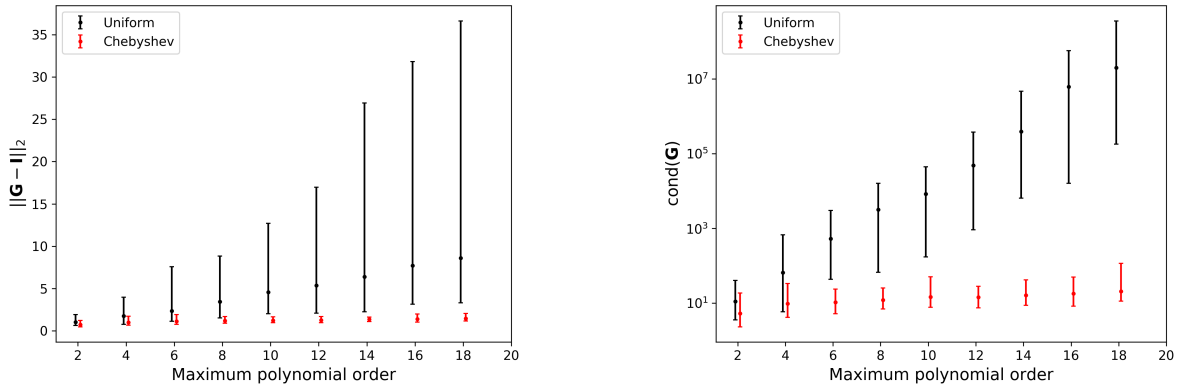
to yield a high probability of $\|\boldsymbol{G} - \boldsymbol{I}\|_2$ being bounded by 0.5, implying stability in the least squares solution. Selecting

$$q(\mathbf{x})^2 = \frac{1}{P} \sum_{i=1}^{P} v_i(\mathbf{x})^2 \tag{2.39}$$

thus minimises the quantity in (2.38) (also see [73] where a similar formulation is known as the *mutual coherence*). The resultant $\rho(\mathbf{x})$ is known as the *induced distribution*. Note that this density is no longer a product density; algorithms such as sequential conditional sampling [24] and mixture sampling [24, 1] have been proposed to sample from this density. Alternatively, the authors of [113] propose the method of *Christoffel least squares*, where samples are drawn from the limit of the induced distribution as the maximal polynomial degree tends to infinity. This is known as the *asymptotic distribution*, or the *weighted pluripotential equilibrium density*. In 1-D, if $\omega(x)$ is uniform on $[-1, 1]$, the asymptotic density is the Chebyshev density

$$\rho_\infty(x) = \frac{1}{\pi\sqrt{1-x^2}}. \tag{2.40}$$

In multiple dimensions, much less is known theoretically, with the asymptotic density depending on the shape of the index set (total order, tensor grid etc.). For total order index sets, if $\omega(\mathbf{x})$ is uniform on the hypercube $[-1, 1]^d$, the asymptotic density is the



(a) Norm deviation.

(b) Condition number.

Figure 2.5 Norm deviation from the identity and condition number of the Gram matrix for various maximal polynomial degrees. Dots indicate the mean over 50 trials, and error bars indicate the minimum-maximum interval over the same trials.

tensorised Chebyshev density

$$\rho_\infty(\mathbf{x}) = \frac{1}{\pi^d \prod_{i=1}^d \sqrt{1 - x_i^2}}. \tag{2.41}$$

If $\omega(\mathbf{x})$ is Gaussian, the density is conjectured to be uniform on a d-dimensional ball. Thus, in cases where the asymptotic density is known, it may be easier to sample from the asymptotic distribution instead of the induced distribution.

As a brief example, consider a bivariate Legendre polynomial basis, formed from the tensor product of two univariate Legendre polynomial bases of maximum degree p. Here, the input points are randomly sampled by the tensorisation of two univariate distributions over $[-1, 1]$, the uniform distribution (input distribution) and the Chebyshev distribution (asymptotic sampling). Figure 2.5 shows the resultant deviation from norm ($\|\boldsymbol{G} - \boldsymbol{I}\|_2$) and the condition number of $\boldsymbol{G}$ for various maximum polynomial degrees p, with the number of sample points fixed at $M = 5P$ where $P = (p+1)^2$ is the basis cardinality. While the condition number of the Gram matrix increases rapidly with a large probability for uniform sample points when the degree increases, the increase for Chebyshev sample points is much milder. At high degrees, least squares approximations by sampling the input distribution are infeasible. However, note that this effect is only pronounced when the polynomial degree is high. To fit low degree polynomials in multiple dimensions, sampling from the input distribution with a sizeable oversampling ratio often suffices.

In conjunction with designing distributions for random sampling, methods to determine effective points via subsampling have formed an active area of research. Starting from K points from a tensor grid basis, or a large pool of randomly sampled points, these approaches use various heuristics to select a subset of points with size $M < K$ where M is closer to P. The main challenge of these methods is to overcome the combinatorial complexity of selecting points that yield desirable approximation properties. Points that maximise the determinant of the (square) polynomial design matrix are known as Fekete points, and Guo et al. [68] showed that greedy optimisation approaches can find the global optimum under mild assumptions. Thus, approximate Fekete points can be found via QR column pivoting [68, 134], LU row pivoting (Leja sequences) [10] and singular value decompositions [134]. The review by Seshadri et al. [133] compares these approaches and finds that Gauss-Legendre points can be approximately recovered using subsampling routines on tensor grid bases. Other optimisation-based methods such as convex relaxation of the sensor selection problem [86] are also mentioned. The sensor selection problem is similar to the D-optimal experiment design problem, which can also be tackled via convex relaxation. Other criteria for optimal experiment design have also been studied in the context of least squares polynomial approximations (see [72] for a review).

2.1.2.3 Compressed sensing

Recall the assumption that the weighted polynomial design matrix $\sqrt{\boldsymbol{P}}\boldsymbol{A}$ needs to be full rank for a least squares approximation. This implies a lower limit on the number of model observations needed. Driven by the need to drive down the computational cost incurred from evaluating expensive models, *compressed sensing* has seen a rise in popularity as a method for estimating polynomial coefficients. Originating from the signal processing community [43, 18], compressed sensing encapsulates a class of methods that estimate unknown parameters of a model under an underdetermined condition, where the amount of information is in some sense smaller than required for estimating all model parameters. In signal processing, this implies that the sampling rate of a signal to be reconstructed is smaller than the Nyquist rate—twice the highest frequency of the signal. In polynomial approximations, this refers to the situation where $M < P$. That is, the linear system (2.21) is underdetermined and compatible with an infinite number of solutions $\mathbf{c}$ which reside in a subspace of $\mathbb{R}^P$ in the absence of noise. In compressed sensing, an s-sparse solution—one that satisfies $\|\mathbf{c}\|_0 \leq s$

where $\|\mathbf{c}\|_0$ counts the number of non-zero entries in $\mathbf{c}$—is sought by solving

$$\begin{aligned} &\underset{\mathbf{c}\in\mathbb{R}^P}{\text{minimise}} && \|\mathbf{c}\|_0 \\ &\text{subject to} && \sqrt{\boldsymbol{P}}\boldsymbol{A}\mathbf{c} = \sqrt{\boldsymbol{P}}\mathbf{f}. \end{aligned} \tag{2.42}$$

In [114], this is shown to be an NP-hard problem—the cost of solving this problem grows exponentially with the size of the polynomial basis, posing a challenge in implementation. However, provided that the matrix $\sqrt{\boldsymbol{P}}\boldsymbol{A}$ satisfies the Restricted Isometry Property (RIP), i.e. each submatrix formed from columns of $\sqrt{\boldsymbol{P}}\boldsymbol{A}$ has singular values bounded around unity [16], one can recover the solution to (2.42) by *pursuit algorithms*, which include orthogonal matching pursuit (OMP) [153] and basis pursuit denoising (BPDN) [19, 45]. Another characterisation of the ability of pursuit algorithms to approximate the compressed sensing solution uses the *mutual coherence*, the largest normalised inner product between the columns of $\sqrt{\boldsymbol{P}}\boldsymbol{A}$ [44]. In Chapter 3 and Chapter 5, an implementation of BPDN is used to form sparse polynomial approximations. BPDN solves a convex relaxation of the compressed sensing problem,

$$\begin{aligned} &\underset{\mathbf{c}\in\mathbb{R}^P}{\text{minimise}} && \|\mathbf{c}\|_1 \\ &\text{subject to} && \left\| \sqrt{\boldsymbol{P}}\boldsymbol{A}\mathbf{c} - \sqrt{\boldsymbol{P}}\mathbf{f} \right\|_2 < \varepsilon, \end{aligned} \tag{2.43}$$

where ε is a small positive constant that accommodates errors including the truncation error of the a finite polynomial expansion, and measurement uncertainties when $\mathbf{f}$ contains noisy measurements of the true underlying function. This quantity is not known *a priori*, but can be empirically determined by cross validation [121].

As with the case of least squares approximations, the search for optimal sampling methods to achieve the best economy from a limited number of model observations is an active area of research. Similar techniques for least squares approximations are also applicable here because of the close relation between the mutual coherence of polynomial design matrices and the weighted Christoffel function defined for least squares (2.38). Hampton and Doostan [74] analyse the use of asymptotic sampling and an algorithm based on Markov chain Monte Carlo to minimise the coherence metric. Jakeman et al. [82] describe the *Christoffel sparse approximation* method, using the same asymptotic distributions as in the Christoffel least squares method. Moreover, Tang and Iaccarino [147] investigate subsampling from tensor grid Gauss-Legendre

quadrature nodes. Gradients of the quantity of interest are also useful in forming sparse polynomial approximations [122, 69].

The success of compressed sensing relies fundamentally on the assumption that the underlying model can be well-approximated by a polynomial with sparse coefficients. The assumption of sparsity is also required for pursuit algorithms to be accurate substitutions of the compressed sensing problem (2.42). In the context of signal processing, sparse representations have been found for a wide range of signals of practical interest such as medical images and audio signals, underpinning standard compression algorithms. In the context of polynomial approximations applied to physical systems, the justification for this assumption is less clear. Most arguments are based on the decay of polynomial coefficients as the polynomial order increases, or the fact that extraneous physical factors tend to be included in a computational analysis. However, due to the much larger scope of applications, it is difficult to make general statements about the sparsity of the polynomial coefficients, and the applicability of compressed sensing varies on a case-by-case basis.

2.2 Model-based dimension reduction

For problems with a large set of input parameters, the size of the index set poses a challenge for calculating coefficients of polynomial approximations. Strategies such as compressed sensing can help ameliorate the computational cost by feature selection in the polynomial coefficients space; in conjunction, techniques for selecting sample points reduce the computational load by ensuring stability in the pre-asymptotic regime. Alternatively, one can simplify the model by reformulating the inputs with a smaller set of parameters by introducing ideas from *model-based dimension reduction*. For a QoI f, the aim is to find functions g and h such that

$$f(\mathbf{x}) \approx g(\mathbf{u}) \text{ with } \mathbf{u} = h(\mathbf{x}), \tag{2.44}$$

where $h : \mathbb{R}^d \to \mathbb{R}^n$ is the *projection function* and $g : \mathbb{R}^n \to \mathbb{R}$, with $n < d$. Embedding the input $\mathbf{x}$ into a lower-dimensional submanifold can make an intractable approximation problem in high-dimensional space feasible. Although the main rationale for model-based dimension reduction is to reduce computational costs for high-dimensional modelling tasks, the benefits that dimension reduction brings for parametric studies do not end here. In the rest of this thesis, we explore how dimension reduction aids the process of computational design by facilitating visualisation of the

behaviour of QoIs in a low-dimensional space, and helps partition the input space to discover regions where output quantities are constrained, leading to performance-based tolerance criteria.

The focus of model-based dimension reduction algorithms is to find low-dimensional structures within the computational model f. This is in contrast with unsupervised dimension reduction algorithms such as principal components analysis (PCA) [85] and its extensions [130, 17], which aim to find similar structures within the input distribution. Our focus will be on linear projection functions,

$$h(\mathbf{x}) = \boldsymbol{W}^\top \mathbf{x}, \tag{2.45}$$

where $\boldsymbol{W} \in \mathbb{R}^{d \times n}$ projects $\mathbf{x}$ onto a linear subspace of the original input domain, which is called the *dimension-reducing subspace*. Since the dimension-reducing subspace can be completely specified by the column span of $\boldsymbol{W}$, it can be assumed that $\boldsymbol{W}$ contains orthogonal columns with unit Euclidean norm without loss of generality.

As an illustrative example (adapted from [138]), consider the function $f(\mathbf{x}) = \log(x_1 + x_2 + x_3)$. This function takes in three parameters, but in fact only varies in one direction $\boldsymbol{W} = [1/\sqrt{3}, 1/\sqrt{3}, 1/\sqrt{3}]^\top$. It is effectively a univariate function of $\boldsymbol{W}^\top \mathbf{x}$, once the inputs are reparameterised accordingly. Graphically, one can illustrate this observation with *sufficient summary plots*, also known as *shadow plots*, such as in Figure 2.6. From the figure, it can be seen that choosing the correct $\boldsymbol{W}$ allows the data to collapse to a curve in one variable. Thus, the pertinent problem of dimension reduction is to identify an optimal projection matrix $\boldsymbol{W}$ that summarises the function of interest, i.e. the correlation structure between the input and the output. Model-based dimension reduction has been studied by communities across various disciplines, which place different assumptions and definitions on the nature of the task. The focus of the present work is on the application of model-based dimension reduction as component steps within composite algorithms. However, to understand the properties of these components, we review algorithms developed for model-based dimension reduction by broadly dividing them into two classes: regression-based and approximation-based.

2.2.1 Model-based dimension reduction as Regression

In regression, the goal is to characterise the relation between the *dependent variable* y and a set of *independent variables* $\mathbf{x}$, both as random variables. The joint distribution $\mathrm{p}(y, \mathbf{x})$ is unknown *a priori*. Generally, the conditional distribution $\mathrm{p}(y|\mathbf{x})$ is estimated through samples from the joint distribution. Here, dimension reduction is used to

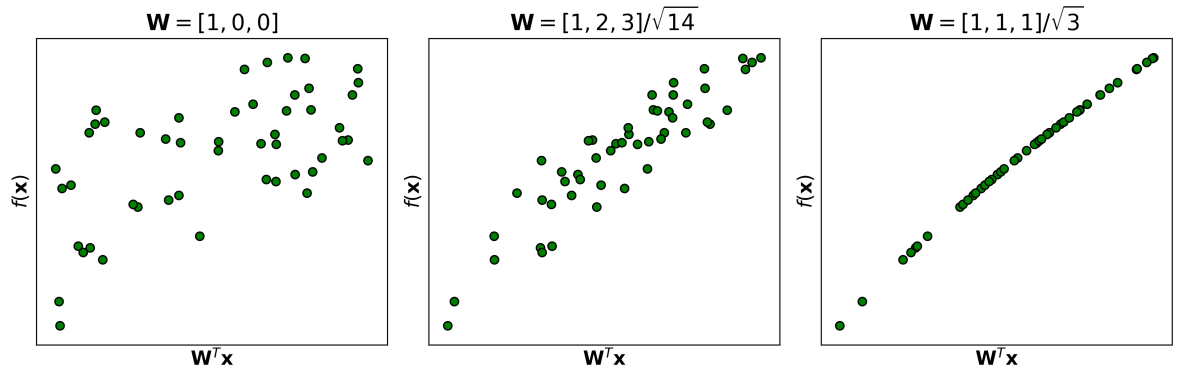


Figure 2.6 Shadow plots of different projections of the function $f(\mathbf{x}) = \log(x_1 + x_2 + x_3)$ with randomly sampled $\mathbf{x}$.

identify the *structural dimension* of the problem; namely, the smallest number of distinct linear combinations of $\mathbf{x}$ required to characterise the distribution of $y|\mathbf{x}$ [38]. Cook [36] formalised the notion of model-based dimension reduction by the concept of *sufficient dimension reduction* (SDR). SDR can be described as finding a subspace spanned by $\boldsymbol{W}$ such that the output is conditionally independent of the original inputs given the projected inputs, i.e.

$$y \perp\!\!\!\perp \mathbf{x} \mid \boldsymbol{W}^\top \mathbf{x}. \tag{2.46}$$

Ma and Zhu [108] classify SDR methods into ones based on inverse regression and ones based on non-parametric or semi-parametric forward regression. The former category contains methods which depend on the moments of the quantity $\mathbf{x}|y$. Since y is scalar, the problem no longer suffers from issues associated with high dimensionality. Important works in this category include sliced inverse regression [99], sliced average variance estimation [37], parametric inverse regression [14], contour regression [97] and directional regression [96]. One limitation common to the listed methods is the need to compute inverse moments via output slicing or kernel regression, introducing hyperparameters that need to be tuned. In addition, some assumptions must be placed on the input moments (see [108, sec. 2.1] for further details).

A relatively new branch of SDR formulates the dimension reduction problem based on non-parametric forward regression. The minimum average variance estimation (MAVE) method [159] solves for a dimension-reducing subspace by local linear kernel smoothing. Assuming a model of the form

$$y = g(\boldsymbol{W}^\top \mathbf{x}) + \varepsilon \tag{2.47}$$

where ε is a random variable uncorrelated with $\mathbf{x}$, a Taylor expansion around observations is used to estimate $\boldsymbol{W}$. More details on the MAVE algorithm are given in Section 2.2.4. Other SDR methods include the non-parametric kernel dimension reduction [53] and direction estimation [162] methods, and the semi-parametric approach by Ma and Zhu [107].

Beyond SDR, the problem of model-based dimension reduction can be approached by assuming special forms of the regression function. One notable example is *projection pursuit regression* [52] where non-parametric regression is used to fit a model of the form

$$g(\mathbf{x}) = \sum_{i=1}^{n} g_i(\mathbf{w}_i^\top \mathbf{x}). \tag{2.48}$$

A similar goal to SDR is achieved when the number of projected variables $n < d$. Indeed, this model is a special case of the more general one defined in [99] for sliced inverse regression.

The presence of gradient information can be used to aid the process of SDR. Samarov [129] finds that the rank of the gradient covariance matrix of the regression function $f(\mathbf{x}) = \mathbb{E}[y|\mathbf{x}]$, namely

$$\boldsymbol{C} = \mathbb{E}\left[\nabla f(\mathbf{x}) \nabla f(\mathbf{x})^\top\right], \tag{2.49}$$

indicates the dimensionality of a space that almost certainly[3] contains the variation of $f(\mathbf{x})$. That is, $f(\mathbf{x})$ almost certainly does not vary on the null space of $\boldsymbol{C}$. In this setting, if $\boldsymbol{W}$ spans the orthogonal complement of the null space of $\boldsymbol{C}$, then (2.46) is satisfied. This implies that $\boldsymbol{C}$ is a good indicator for tuning the number of projected directions in a dimension reduction model.

2.2.2 Model-based dimension reduction as Approximation

When working with data derived from deterministic models, one may challenge the validity of regression-based methods. As explained previously, these methods assume a regression model of the form $g(\boldsymbol{W}^\top \mathbf{x}) + \varepsilon$ with ε being a random variable uncorrelated with $\mathbf{x}$. As is the case for computer simulations with CFD and finite elements in the absence of convergence issues, deterministic models yield the same output $f(\mathbf{x})$ for a fixed $\mathbf{x}$, so the assumption of regression models may not apply. However, errors can arise from approximation owing to the function lying outside of the space of functions that the approximant belongs to, and from the approximation

[3] with probability 1

methods used. For polynomial approximations, sources of error include the truncation error and aliasing errors.

Also known as the study of *computer experiments* [89, 93], methods more closely related to the area of function approximation can be more appropriate. To approximate a model $f(\mathbf{x})$, these methods assume a *known* input distribution $\omega(\mathbf{x})$ and the goal is to find a function within a Hilbert space to minimise the $L_2(\omega)$ approximation error, as explained in the beginning of Section 2.1.

Logan and Shepp [104] first studied functions of the form $g(\mathbf{w}^\top \mathbf{x})$ for $\mathbf{w} \in \mathbb{R}^d$ in the context of plane waves in computed tomography, coining the term *ridge functions*. Pinkus [123] defined *generalised ridge functions* as functions of the form $g(\boldsymbol{W}^\top \mathbf{x})$, where $\boldsymbol{W} \in \mathbb{R}^{d\times n}, n < d$ are called the *ridge directions* and the function $g : \mathbb{R}^n \to \mathbb{R}$ the *ridge profile*. In this thesis, functions of this form are referred to as ridge functions for brevity. The use of ridge functions for approximation is called *ridge approximation*.

In [32], Constantine et al. proposed an alternating optimisation procedure for ridge approximation based on *active subspaces*. First studied in [127] and [31], active subspaces are based on an eigendecomposition of the gradient covariance matrix

$$\boldsymbol{C} = \mathbb{E}\left[\nabla f(\mathbf{x}) \nabla f(\mathbf{x})^\top\right] = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top. \tag{2.50}$$

The matrices $\boldsymbol{Q}$ and $\boldsymbol{\Lambda}$ are partitioned

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{W} & \boldsymbol{V} \end{bmatrix}, \quad \boldsymbol{\Lambda} = \begin{bmatrix} \boldsymbol{\Lambda}_1 & \\ & \boldsymbol{\Lambda}_2 \end{bmatrix}, \tag{2.51}$$

where $\boldsymbol{W}$ contains the n eigenvectors of $\boldsymbol{C}$ that correspond to the largest n eigenvalues lying on the diagonal of $\boldsymbol{\Lambda}_1$. The column span of $\boldsymbol{W}$ is called the active subspace, a subspace within which most of the function's activity is present. Its orthogonal complement, the column span of $\boldsymbol{V}$, is called the *inactive subspace*, where it is expected that the function varies little. The choice of $\boldsymbol{W}$ here as the ridge directions allows the expected $L_2(\omega)$ error of the optimal ridge approximation $\mathbb{E}[f(\mathbf{x})|\boldsymbol{W}^\top \mathbf{x}]$[4] to be bounded,

$$\left\| f(\mathbf{x}) - \mathbb{E}[f(\mathbf{x})|\boldsymbol{W}^\top \mathbf{x}] \right\|_{L_2(\omega)} \leq \mathcal{C}(\omega) \mathrm{Trace}(\boldsymbol{\Lambda}_2)^{1/2}, \tag{2.52}$$

where $\mathcal{C}(\omega)$ is the Poincaré constant dependent only on $\omega(\mathbf{x})$. Note that formulations beyond the gradient covariance matrix can also be used, such as that proposed by

[4] Theorem 8.3 of [123] shows that this is the optimal ridge approximation for a fixed $\boldsymbol{W}$ in terms of the $L_2(\omega)$ error.

[94]. The idea of active subspaces has seen a broad range of applications for design space exploration, in areas including aerofoil shape [106, 41, 65], compressor blades [135, 137], hypersonic scramjet [26] and battery design [30, 35]. It is also studied as a tool for automated discovery of governing physical laws through dimensional analysis [40, 84].

A limitation of active subspaces is the need for gradient evaluations of the function. In the absence of adjoints, such as when dealing with legacy codes or experimental data, gradient evaluations may not be readily available. In certain situations, gradients can be approximated by finite differences [50, 33] or a low-order global surrogate model [135], but this does not always yield sufficient accuracy. This motivates gradient-free approaches for ridge approximation. Examples include polynomial variable projection [77] and Gaussian ridge functions [138]. The former method specifies the ridge profile as a polynomial defined over the dimension-reducing subspace, and analytically evaluates its gradient with respect to both the polynomial coefficients and the ridge subspace to minimise the training mean squared error (MSE) with a Gauss-Newton method. More details are given in Section 2.2.4.

2.2.3 Discussion: Regression vs. Approximation

From the discussion in this section, it is apparent that methods in the category of function approximation are more appropriate for the context of this thesis. Although crucial differences exist in the setting of regression-based and approximation-based algorithms, the outcomes of both classes of methods share some similarities. In [138] and [137], the authors find that SDR methods perform worse than approximation-based methods on CFD simulation data of compressor fan blades. In [6], projection pursuit regression is applied to data from computer experiments, concluding that it is less accurate than Kriging. Glaws et al. [58] draw theoretical and practical links between SDR and ridge *recovery*, where the assumption that the QoI is exactly a ridge function is made. In [57], the authors demonstrate that the moment matrices underpinning inverse regression methods can be computed in models from computer experiments, but do not show whether application of inverse regression is suitable for ridge approximation.

To the best of the author's knowledge, whether regression methods can be applied directly to ridge approximation of deterministic models remains an open question. Nonetheless, the work presented in the remainder of this thesis serves to provide empirical evidence that some degree of success can be derived from applying a re-

gression method (MAVE) to ridge approximation. With this in mind, we leave further theoretical consideration of this matter outside the scope of this thesis.

2.2.4 Details on dimension reduction algorithms in this thesis

To conclude the review of dimension reduction methods, we provide an overview of the implementations of exemplar dimension reduction methods that are used in numerical examples. For more details on the algorithms and theory, the reader is referred to the works cited.

2.2.4.1 Linear model

The coefficients of a global linear model of a QoI give a rough estimation of the leading dimension-reducing subspace direction. If the QoI varies largely linearly in the input domain, and it is known that the dimension-reducing subspace is one-dimensional, this heuristic will find the required subspace at a low cost. This method is described with Algorithm 1.3 in [28], which we reproduce below. We also note the relationship of this heuristic with the ordinary least squares (OLS) method in SDR [100].

1. Draw input-output pairs $\{\mathbf{x}^{(m)}, f(\mathbf{x}^{(m)})\}_{m=1}^{M}$ where $M = \alpha d$ for an oversampling ratio $\alpha > 1$, where d is the dimension of the input space.

2. Solve the least squares problem

$$\underset{c,\mathbf{w}}{\text{minimise}} \quad \|\boldsymbol{X}\mathbf{w} + c - \mathbf{f}\|_2 , \tag{2.53}$$

   where the m-th row of $\boldsymbol{X}$ is $\mathbf{x}^{(m)}$ and $f_m = f(\mathbf{x}^{(m)})$.

3. Take the leading ridge direction to be $\mathbf{w}/\|\mathbf{w}\|_2$.

2.2.4.2 Active subspaces

Assuming access to gradient evaluations, the active subspace is formed by computing the gradient covariance matrix (2.50) and partitioning its eigendecomposition (2.51) such that the eigenvalue gap between the last eigenvalue of $\boldsymbol{\Lambda}_1$ and the first eigenvalue of $\boldsymbol{\Lambda}_2$ is large. In practice, the gradient covariance matrix is approximated with a finite sample estimate. Constantine [28, Algorithm 3.1] provides a recipe as follows.

1. Draw input points randomly and evaluate the output gradients at these points to form the set $\{(\mathbf{x}^{(m)}, \nabla f(\mathbf{x}^{(m)}))\}_{m=1}^{M}$.

2. Approximate the gradient covariance matrix as

$$\boldsymbol{C} \approx \widehat{\boldsymbol{C}} = \frac{1}{M} \sum_{m=1}^{M} \nabla f(\mathbf{x}^{(m)}) \nabla f(\mathbf{x}^{(m)})^{\top}. \tag{2.54}$$

3. Compute the eigendecomposition

$$\widehat{\boldsymbol{C}} = \widehat{\boldsymbol{Q}} \widehat{\boldsymbol{\Lambda}} \widehat{\boldsymbol{Q}}^{\top}, \tag{2.55}$$

and partition $\widehat{\boldsymbol{Q}}, \widehat{\boldsymbol{\Lambda}}$ according to (2.51) to obtain the active and inactive subspaces.

2.2.4.3 Polynomial variable projection

In [77], the authors formulate the ridge approximation problem from a data-driven perspective. Given a set of input-output pairs $\{\mathbf{x}^{(m)}, f(\mathbf{x}^{(m)})\}_{m=1}^{M}$, the aim is to find the ridge profile g and dimension-reducing subspace $\boldsymbol{W}$ such that the MSE is minimised,

$$\begin{aligned} \underset{g, \boldsymbol{W}}{\text{minimise}} \quad & \sum_{m=1}^{M} \left[f(\mathbf{x}^{(m)}) - g(\boldsymbol{W}^{\top} \mathbf{x}^{(m)}) \right]^2 \\ \text{subject to} \quad & g \in \mathbb{P}^p(\mathbb{R}^n) \\ & \operatorname{colspan}(\boldsymbol{W}) \in \mathbb{G}(n, \mathbb{R}^d), \end{aligned} \tag{2.56}$$

where $\mathbb{P}^p(\mathbb{R}^n)$ is the set of all n-dimensional polynomials of total degree up to p and $\mathbb{G}(n, \mathbb{R}^d)$ is the Grassmann manifold—i.e. the set of all n-dimensional subspaces in $\mathbb{R}^d$. This is a non-linear least squares problem, since the argument within the square loss is non-linear with respect to $\boldsymbol{W}$.

The authors propose the use of *variable projection* [59] to minimise this cost function, which is an alternating procedure. First, fix $\boldsymbol{W}$, which makes this problem a linear least squares problem with respect to the coefficients of g. This is solved via the Moore-Penrose pseudoinverse. Then, the residual orthogonal to the range of the Vandermonde matrix formed from g is minimised with respect to $\boldsymbol{W}$ via the Gauss-Newton method on the Grassmann manifold [49]. An algorithm of this method is provided in [77].

2.2.4.4 Minimum average variance estimation

The MAVE method for sufficient dimension reduction was proposed by Xia et al. [159]. Given a sample $\{\mathbf{x}^{(m)}, y^{(m)}\}_{m=1}^{M}$ from the predictor and response $(\mathbf{x}, y)$, it aims to fit a

regression model of the form $g(\boldsymbol{W}^\top \mathbf{x})$ by considering

$$\begin{aligned} \underset{\boldsymbol{W}}{\text{minimise}} \quad & \mathbb{E}\left[\left(y - \mathbb{E}[y|\boldsymbol{W}^\top \mathbf{x}]\right)^2\right] \\ \text{subject to} \quad & \boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I}. \end{aligned} \tag{2.57}$$

The authors express the quantity $\mathbb{E}[y^{(m)}|\boldsymbol{W}^\top \mathbf{x}^{(m)}]$ using a local linear expansion about a given point $\mathbf{x}^{(0)}$

$$\mathbb{E}[y^{(m)}|\boldsymbol{W}^\top \mathbf{x}^{(m)}] \approx a_0 + \mathbf{b}_0^\top \boldsymbol{W}^\top (\mathbf{x}^{(m)} - \mathbf{x}^{(0)}). \tag{2.58}$$

Using this form, they solve the approximation to (2.57) as an alternating least squares problem

$$\begin{aligned} \underset{a_j, \mathbf{b}_j, \boldsymbol{W}}{\text{minimise}} \quad & \sum_{i=1}^{M} \sum_{j=1}^{M} \left[y^{(j)} - a_j - \mathbf{b}_j^\top \boldsymbol{W}^\top (\mathbf{x}^{(i)} - \mathbf{x}^{(j)})\right]^2 \omega_{ij} \\ \text{subject to} \quad & \boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I}, \end{aligned} \tag{2.59}$$

where the weights ω_{ij} are determined through a normalised kernel function K_h,

$$\omega_{ij} = \frac{K_h(\boldsymbol{W}^\top (\mathbf{x}^{(i)} - \mathbf{x}^{(j)}))}{\sum_{k=1}^{M} K_h(\boldsymbol{W}^\top (\mathbf{x}^{(i)} - \mathbf{x}^{(j)}))}. \tag{2.60}$$

The procedure iterates by

- fixing $a_j, \mathbf{b}_j$ and optimising with respect to $\boldsymbol{W}$,
- fixing $\boldsymbol{W}$ and optimising with respect to $a_j, \mathbf{b}_j$,

The minimiser for both steps can be expressed analytically, with details in, e.g., [95].

Chapter 3

Sensitivity analysis via polynomial ridges

In uncertainty quantification, sensitivity analysis (SA) is a task that aims to understand the relative importance of inputs on the output QoIs. Methods within this area can be broadly divided into *local* and *global* SA. The former is concerned with sensitivities at specific points within the input domain (e.g. partial derivatives with respect to the input parameters), whereas the latter aims to calculate sensitivity values based on the average effect of parameters on the whole domain. In this work, we first review global methods of SA, and proceed to propose a novel set of indices that lies in between the two paradigms. These indices, called *extremum Sobol' indices*, are specialised to characterise sensitivities near localised *regions* where the output reaches extremity. Throughout this chapter, we work within the remit of PCE-based methods, and describe, algorithmically and empirically, how ridge approximations expedite the process of sensitivity analysis.

3.1 Variance-based global sensitivity analysis

In global sensitivity analysis, the goal is to characterise the importance of individual or groups of input parameters throughout the input domain to an output QoI, when the input varies according to a known input distribution. A commonly used set of indices for measuring this importance is based on the conditional variance, namely the well-known *Sobol' indices* [143, 144]. They are related to the total indices [78] and the Fourier amplitude sensitivity test (FAST) indices [39]. However, these are not the only metrics available: over the years, other metrics for global SA have been proposed, including

those based on conditional skewness [156] and kurtosis [42]. Beyond this, Hooker [79] and Liu and Owen [102] introduce the notion of superset importance, which is the sum of all Sobol' indices based on a superset of an input set. Techniques based on averaging partial derivative evaluations of a function have also been extensively studied [15, 111], and the relationship between these *elementary effects* and total Sobol' indices have been investigated [91].

Here, we focus on the computation of the variance-based Sobol' indices. Given a subset of input parameters $\mathbf{x}_S$ indexed by $S \subset \{1, 2, ..., d\}$, the corresponding Sobol' index is [143]

$$\sigma_S = \frac{\mathrm{Var}[\mathbb{E}[f(\mathbf{x}) \mid \mathbf{x}_S]]}{\sigma}, \tag{3.1}$$

where σ is the output variance. When d is large and $f(\mathbf{x})$ expensive to evaluate, a Monte Carlo calculation of the integrals involved is intractable. In the seminal work by Sudret [145], it is shown that the coefficients of a PCE approximation of the QoI can be used to straightforwardly estimate the Sobol' indices. The paper starts with the Sobol' decomposition [144] (also known as the functional ANOVA decomposition) of $f(\mathbf{x})$

$$f(\mathbf{x}) = f_0 + \sum_{i=1}^{d} f_i(x_i) + \sum_{i<j} f_{ij}(x_i, x_j) + \cdots + f_{12\ldots d}(x_1, x_2, ..., x_d), \tag{3.2}$$

with $f_0 = \mathbb{E}[f]$ and the subfunction f_S defined recursively as

$$f_S(\mathbf{x}_S) = \mathbb{E}[f(\mathbf{x}) \mid \mathbf{x}_S] - \sum_{T \subset S} f_T(\mathbf{x}_T). \tag{3.3}$$

The Sobol' indices are then defined as the partial variances of the corresponding subfunction

$$\sigma_S = \frac{\mathrm{Var}[f_S(\mathbf{x}_S)]}{\mathrm{Var}[f(\mathbf{x})]}, \tag{3.4}$$

or, equivalently, in the form (3.1). By comparing the Sobol' decomposition with the PCE series, it can be seen that each subfunction $f_S(\mathbf{x}_S)$ can be expressed as the sum of the terms whose polynomial basis functions involve only the parameters $\mathbf{x}_S$. The orthonormality of the polynomial basis thus implies that the integrals involved in (3.4) can be evaluated simply by summing up the corresponding coefficients squared. Namely,

$$\sigma_S = \frac{\sum_{\mathrm{nz}(k)=S} c_k^2}{\sum_k c_k^2}, \tag{3.5}$$

where the function $\mathrm{nz}(\cdot)$ returns the positions of the non-zero indices in a multi-index. For instance,

$$\mathrm{nz}((0,1,2,0,4)) = \{2,3,5\}. \tag{3.6}$$

A measure of the importance of an input parameter is the total Sobol' index [144], which is the sum of all Sobol' indices that contain the parameter concerned. This index factors in all orders of interaction within the function that involve the parameter concerned. For instance, for parameter x_i, the corresponding total Sobol' index is

$$\widehat{\sigma}_i = \sum_{i \in S} \sigma_S. \tag{3.7}$$

As discussed in Chapter 2, solving for the coefficients of a PCE approximation can be intractable in high dimensions. So, can one exploit low-dimensional structures to compute sensitivity indices? The idea of calculating global sensitivity metrics via ridge approximations has been studied previously by Constantine and Diaz [29]. Using ideas from active subspaces, the authors define *activity scores* that are based on the eigendecomposition of the gradient covariance matrix (2.50). They draw comparisons between activity scores and total Sobol' indices (see Theorem 4.1 in [29]). While their work defines a new type of sensitivity index tailored to ridge approximations, the present work builds upon the idea of ridge approximations with polynomials to develop a method for calculating moment-based sensitivity indices, such as the Sobol' indices.

Consider the polynomial ridge approximation of a scalar function

$$f(\mathbf{x}) \approx g(\boldsymbol{W}^\top \mathbf{x}) \Rightarrow \mathbf{f} \approx \mathbf{g} = \boldsymbol{A}_n \mathbf{c}_n, \tag{3.8}$$

where $\boldsymbol{A}_n \in \mathbb{R}^{M \times q}$ is the polynomial design matrix for a polynomial basis defined on the dimension-reducing subspace that is spanned by columns of $\boldsymbol{W}$. That is,

$$\boldsymbol{A}_n = \begin{bmatrix} v_1(\boldsymbol{W}^\top \mathbf{x}^{(1)}) & v_2(\boldsymbol{W}^\top \mathbf{x}^{(1)}) & \dots & v_q(\boldsymbol{W}^\top \mathbf{x}^{(1)}) \\ v_1(\boldsymbol{W}^\top \mathbf{x}^{(2)}) & v_2(\boldsymbol{W}^\top \mathbf{x}^{(2)}) & \dots & v_q(\boldsymbol{W}^\top \mathbf{x}^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ v_1(\boldsymbol{W}^\top \mathbf{x}^{(M)}) & v_2(\boldsymbol{W}^\top \mathbf{x}^{(M)}) & \dots & v_q(\boldsymbol{W}^\top \mathbf{x}^{(M)}) \end{bmatrix}. \tag{3.9}$$

The corresponding coefficients of the polynomial ridge profile g are contained in $\mathbf{c}_n$. The advantage of using a polynomial defined over a subspace is the greatly reduced number of coefficients to calculate. Once a suitable ridge subspace has been

found, the low-dimensional space allows construction of polynomials with simple heuristics, such as least squares with uniform Monte Carlo sampling over the projected domain. Note that since moments and sensitivities are not quantified over the projected space, the basis and sampling strategy does *not* need to match the shape and induced measure over the projected space, which is in general a non-rectangular domain with a dependent distribution.

Calculating the sensitivity indices from the polynomial ridge coefficients then amounts to a *rearrangement of coefficients*. Noting that the polynomial ridge function can be seen as a polynomial of $\mathbf{x}$ instead of $\boldsymbol{W}^\top \mathbf{x}$, one can solve a linear system for the full space coefficients $\mathbf{c}$

$$\boldsymbol{A}\mathbf{c} = \boldsymbol{A}_n \mathbf{c}_n, \tag{3.10}$$

where $\boldsymbol{A} \in \mathbb{R}^{M \times P}$ is the polynomial design matrix in the full space. Disregarding the effect of numerical round-off errors and issues with ill-conditioning, $\mathbf{c}$ can be recovered without loss of accuracy given M non-degenerate points where $M \geq P$. This step incurs no computational costs related to additional function evaluations. In the case that the input dimension is so large that P is prohibitively large for solving this linear system, subset selection schemes for truncating the full space polynomial basis, such as that proposed by Blatman and Sudret [9], can be used when determining the full space coefficients. This reduces the number of coefficients to be determined and prunes the basis of unnecessary terms, but the rearrangement step is no longer strictly lossless.

The accuracy of calculating Sobol' indices via polynomial ridge approximations relies on the quality of the estimated ridge subspace. In turn, this relies on how amenable the quantity of interest is to a polynomial ridge approximation, i.e. whether a clear low-dimensional structure exists. To be more precise, a proposition accounting for an error bound in calculated sensitivities given a perturbation in the function in the $L_2(\omega)$ norm is established below.

Proposition 3.1.1. *Suppose that g and $\widetilde{g}$ are two orthonormal polynomial chaos expansions defined on $\mathbb{R}^d$ with input measure ω, satisfying $\|g - \widetilde{g}\|_{L_2(\omega)} < \varepsilon$. Let σ_S and $\widetilde{\sigma}_S$ be the Sobol' indices for a subset $S \subset \{1, 2, ..., d\}$ corresponding to g and $\widetilde{g}$ respectively. The following holds*

$$|\sigma_S - \widetilde{\sigma}_S| < C(S)\varepsilon^2 \tag{3.11}$$

where $C(S) \sim \mathcal{O}(1)$ is a constant depending on S.

Proof. Let $\mathbf{c}$ and $\widetilde{\mathbf{c}}$ be the coefficients of the polynomials g and $\widetilde{g}$ respectively over the orthonormal basis $\{v_i(\mathbf{x})\}$. Using the orthonormality of the polynomial basis,

$$\begin{aligned} \|g - \widetilde{g}\|_{L_2,\omega}^2 &= \int_{\mathcal{D}} \left(\sum_i (c_i - \widetilde{c}_i)^2 v_i(\mathbf{x}) \right)^2 \omega(\mathbf{x})\, d\mathbf{x} \\ &= \sum_i (c_i - \widetilde{c}_i)^2 \int_{\mathcal{D}} v_i(\mathbf{x})^2\, \omega(\mathbf{x})\, d\mathbf{x} \\ &= \|\mathbf{c} - \widetilde{\mathbf{c}}\|_2^2 < \varepsilon^2. \end{aligned} \tag{3.12}$$

(Parseval's identity). The corresponding error in Sobol' indices is thus

$$\begin{aligned} |\sigma_S - \widetilde{\sigma}_S| &= \left| \frac{a}{b} - \frac{\widetilde{a}}{\widetilde{b}} \right| \\ &= \frac{|a\widetilde{b} - \widetilde{a}b|}{b\widetilde{b}} \\ &= \frac{|a\widetilde{b} - ab + ab - \widetilde{a}b|}{b\widetilde{b}} \\ &= \frac{|a(\widetilde{b} - b) + b(a - \widetilde{a})|}{b\widetilde{b}} \\ &< \underbrace{\frac{|a| + |b|}{b\widetilde{b}}}_{C(S)} \varepsilon^2 \end{aligned} \tag{3.13}$$

where $a, \widetilde{a}, b, \widetilde{b}$ denote $\sum_{i \in S} c_i^2, \sum_{i \in S} \widetilde{c}_i^2, \sum_i c_i^2, \sum_i \widetilde{c}_i^2$ respectively

□

Proposition 3.1.1 establishes corresponding stability results for the following two situations:

- Approximating Sobol' indices for a QoI which is an exact polynomial ridge function $f(\mathbf{x}) = g(\boldsymbol{W}^\top \mathbf{x})$, but with a perturbation in the calculated subspace $g(\widetilde{\boldsymbol{W}}^\top \mathbf{x})$. In Theorem 4.2.2 in Chapter 4, it will be shown that a small perturbation in the ridge subspace translates to a corresponding $L_2(\omega)$ error, from which Proposition 3.1.1 can directly follow.
- Approximating Sobol' indices for a QoI which is not exactly a polynomial ridge function, $f(\mathbf{x}) = g(\boldsymbol{W}^\top \mathbf{x}) + e(\mathbf{x})$, with $g(\boldsymbol{W}^\top \mathbf{x})$. Provided with the mild assumption that $f \in L_2(\omega)$ (see (2.1)), it can be shown that the truncation error $e(\mathbf{x})$ can be expressed as an infinite polynomial chaos expansion of polynomial basis

functions not in the basis of $g(\boldsymbol{W}^\top \mathbf{x})$, usually corresponding to higher-order terms. Hence, an additional term that is a fraction of $\|e(\mathbf{x})\|^2_{L_2(\omega)}$ is added to the error in Sobol' indices. For smooth functions well-approximated by polynomials, the magnitude decay of coefficients as the polynomial order increases implies that this additional term is small.

3.1.1 Numerical example

In the following, polynomial ridge approximation is used to estimate the Sobol' indices of an analytical function and the results are compared against other known heuristics. The function of interest $f : [-1, 1]^6 \to \mathbb{R}$ is defined as

$$f(\mathbf{x}) = f_1(\boldsymbol{W}^\top \mathbf{x}) + s f_2(\boldsymbol{V}^\top \mathbf{x}), \tag{3.14}$$

with

$$\begin{aligned} f_1(\mathbf{u}) &= u_1^2 - 0.1u_2^2 - 2u_1^2u_2^2 + 2u_1u_2, && \mathbf{u} \in \mathbb{R}^2, \\ f_2(\mathbf{z}) &= z_1^3 - z_2 + 2z_3z_4^2, && \mathbf{z} \in \mathbb{R}^4, \end{aligned} \tag{3.15}$$

and the orthogonal columns of $(\boldsymbol{W}, \boldsymbol{V})$ span, respectively, the column and left null space of

$$\begin{bmatrix} 2 & 3 & 1 & -4 & 1 & 0.1 \\ -3 & 1 & -2 & 1 & -1 & -0.3 \end{bmatrix}^\top. \tag{3.16}$$

Thus, $f(\mathbf{x})$ consists of a function $f_1(\boldsymbol{W}^\top \mathbf{x})$ varying within the column space of $\boldsymbol{W}$ (a ridge function) plus a multiple of another function $f_2(\boldsymbol{V}^\top \mathbf{x})$ varying entirely outside of the column space of $\boldsymbol{W}$. Therefore, s can be considered a parameter that modulates the deviation of $f(\mathbf{x})$ from an exact ridge function within the column space of $\boldsymbol{W}$, i.e. it allows f to vary slightly outside of the plane spanned by the columns of $\boldsymbol{W}$. We evaluate the sensitivity indices from noisy samples $\{\mathbf{x}^{(i)}, f(\mathbf{x}^{(i)}) + \varepsilon^{(i)}\}_{i=1}^M$, where the sample inputs $\mathbf{x}^{(i)}$ are drawn from a uniform distribution over the input domain, $\omega(\mathbf{x}) = 1/2^6$, and $\varepsilon^{(i)}$ represents additive Gaussian random noise with a standard deviation of 10^{-4}. Four methods are compared for this task:

- **Polynomial ridge approximation**: Using MAVE [159] (see Section 2.2.4), a dimension-reducing subspace for the function of interest is calculated, setting the reduced dimension $n = 2$. Then, a low-dimensional quadratic polynomial ridge is fitted over the reduced space. Since the dimension of the reduced space is

small, we can afford to use a tensor grid. A Legendre polynomial basis (see (2.11)) is used in the projected space; since we do not need to quantify moments, the choice of basis is not critical here. Note that in this example, the choice of $n = 2$ is prescribed according to the known function, but in practice, for an unknown model, methods such as cross validation (or visualisation with projection plots) may be used for determining the suitable dimensionality.

Then, a random sample consisting of P points in the full input space $[-1, 1]^6$ is selected, where P is the cardinality of the full space polynomial (formed using a Legendre polynomial basis on an isotropic total order grid with maximum degree 4). The matrices $\boldsymbol{A}$ and $\boldsymbol{A}_n$ are evaluated in (3.10) based on these points, and the coefficients for the quadratic polynomial ridge are substituted into $\mathbf{c}_n$. From this, the full space coefficients $\mathbf{c}$ are solved by inverting $\boldsymbol{A}$ (which is well-conditioned due to the choice of Legendre polynomials). The coefficients $\mathbf{c}$ can then be applied in (3.5) to calculate the Sobol' indices. The implementation of MAVE is adapted from the R code from Hang and Xia[1].

- **Quasi-Monte Carlo**: The integrals defining the Sobol' indices are computed via a Sobol' sequence, as specified in [144]. The open-source Python library `SALib` [75] is adapted for this method.

- **Sparse regression with compressed sensing**: A full-space Legendre polynomial is fitted on an isotropic total order basis of maximum degree 4 using compressed sensing with the BPDN method, as in (2.43). Input points are sampled according to the uniform distribution on $[-1, 1]^6$. Then, the resultant coefficients are used in (3.5) to calculate the Sobol' indices.

- **Least squares regression on full coefficient space**: The same as the above, except that the full-space polynomial is fitted using least squares.

Note that the least squares approach is restricted by a lower limit on the number of function evaluations required (equal to the basis cardinality, $P = 210$), as explained in Section 2.1.2. In addition, compressed sensing is only applicable when the number of function evaluations is under the aforementioned limit. The two largest Sobol' indices σ_{13} and σ_{24} are computed using the methods above. To benchmark these results, these indices are also computed using integration via Gauss quadrature on a degree 4 tensor grid of Legendre polynomials with $(4 + 1)^6 = 15625$ points, the results of which are treated as the truth. The fact that $f(\mathbf{x})$ is a polynomial allows this integration to be

[1] https://cran.r-project.org/web/packages/MAVE/index.html

exact. Figure 3.1 shows the results of this comparison for $s = 0, 0.1, 0.25$ respectively over 30 trials. By considering multiple experiments, it was found that 30 trials give representative error bars.

From the results, it is clear that polynomial-based techniques offer a significant advantage over quasi-Monte Carlo integration. This shows that sensitivity analysis of a function that is smooth and well-approximated by a polynomial can be greatly facilitated by PCE. Additionally, the fact that this function is (approximately) a ridge function gives the ridge approximation approach an advantage over sparse methods such as compressed sensing. In general, ridge functions need not have sparse coefficients unless the ridge structure aligns well with the canonical directions in the input space. The cases of $s = 0.1$ and $s = 0.25$ add variation outside of the two-dimensional subspace spanned by the columns of $\boldsymbol{W}$, which reduces the accuracy of the ridge approximation. Nonetheless, the ridge approximation approach compares well against compressed sensing for modest values of s, with its performance starting to compare less favourably against compressed sensing when $s = 0.25$ for evaluating σ_{13}. However, if we are allowed to evaluate the function more than 210 times, regression on the full space using a total order grid is superior.

3.2 Extremum sensitivity analysis

Engineering systems are often designed to work at operating points where an output performance metric is near-optimal. In this section, we focus on the problem of extremum sensitivity analysis: which input variables, under what distributions, predominantly drive the output response near its highest/lowest levels? Extremum sensitivity analysis is useful for informing the construction of tailored surrogate models that aid optimisation, by isolating critical variables that most influence the output near extrema. Moreover, measuring uncertainties at output extrema helps engineers understand the occurrence of extreme events, and how to mitigate the associated risks. The goal of this section is to develop indices more localised than global sensitivity metrics (but not restricted to isolated points, e.g. derivatives), automatically adapted to regions of interest within the input domain corresponding to output extrema.

In the literature, papers directly addressing extremum sensitivity analysis are scarce. Recently, there has been work on higher-order indices utilising skewness and kurtosis, directed at the analysis of the tails of the output distribution. In Owen et al. [118], the authors mention that a positively skewed distribution will attain more extreme positive values than a symmetric distribution with the same variance; distributions

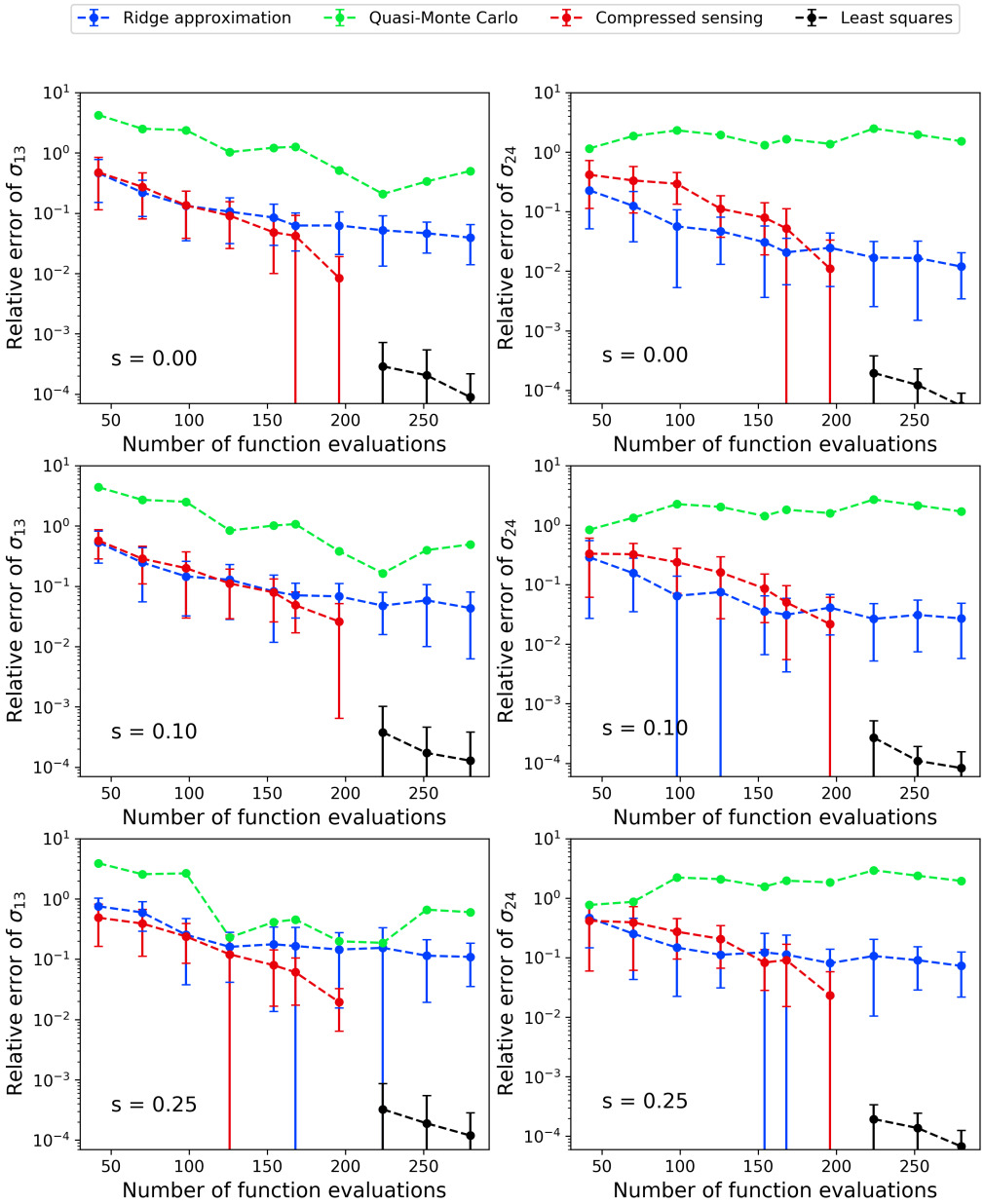


Figure 3.1 Accuracy of the second-order Sobol' indices σ_{13} (left) and σ_{24} (right) for the function $f(\mathbf{x})$ defined in (3.14), averaged over 30 trials. The parameter s is set to be 0 (top), 0.1 (middle) and 0.25 (bottom) respectively.

with a large kurtosis will also attain greater extremes on both sides. Geraci et al. [56], who introduce polynomial-chaos-based ideas for computing skewness- and kurtosis-based indices, find applications of these indices in more accurate approximation of the PDF of the output response, especially near output extrema. However, neither work directly addresses the problem of sensitivity analysis near output extrema. In this work, a novel type of sensitivity indices called *extremum Sobol' indices* is proposed. As the name suggests, the aim is to compute Sobol' indices of functions when the input distribution is restricted to regions in the domain leading to output extrema. This conditional distribution is modelled via a rejection sampling heuristic known as *Monte Carlo filtering* (MCF), which can be facilitated by polynomials and associated ridges. In this section, we first review the theory and computational details for skewness-based sensitivity indices, which sets the stage for introducing the MCF-based extremum Sobol' indices.

3.2.1 Skewness-based sensitivity indices

Assuming that the input domain is a unit hypercube $\mathcal{D} = [0,1]^d$ endowed with uniform marginal distributions, Owen et al. [118] define *higher-order Sobol' indices* by generalising the Sobol' identity, which is based on the Sobol' decomposition (3.2). This identity states that [144, Thm. 2]

$$\sigma\sigma_S = \left(\int_{\mathcal{D}} \int_{\mathcal{D}} f(\mathbf{x}) f(\mathbf{x}_S; \mathbf{x}_{-S}^{(a)}) \, d\mathbf{x} \, d\mathbf{x}^{(a)} - \mu^2 \right), \tag{3.17}$$

where the notation $\mathbf{x}_S; \mathbf{x}_{-S}^{(a)}$ denotes a point $\mathbf{y}$ in $\mathcal{D}$ where $y_j = x_j$ for $j \in S$ and $y_j = x_j^{(a)}$ for $j \notin S$, and μ denotes the mean of the output. The generalisation for calculating the (unnormalised) skewness index for a subset of parameters S reads

$$t'_S = \int_{\mathcal{D}} \int_{\mathcal{D}} \int_{\mathcal{D}} f(\mathbf{x}) f(\mathbf{x}_S; \mathbf{x}_{-S}^{(a)}) f(\mathbf{x}_S; \mathbf{x}_{-S}^{(b)}) \, d\mathbf{x} \, d\mathbf{x}^{(a)} \, d\mathbf{x}^{(b)} - \mu^3. \tag{3.18}$$

Also in [118], it is shown that

$$t'_S + \mu^3 = \mathbb{E}\left[\mathbb{E}[f(\mathbf{x}) \mid \mathbf{x}_S]^3 \right], \tag{3.19}$$

which parallels the definition of Sobol' indices where

$$\sigma\sigma_S + \mu^2 = \mathbb{E}\left[\mathbb{E}[f(\mathbf{x}) \mid \mathbf{x}_S]^2 \right]. \tag{3.20}$$

The authors in [118] remark that skewness indices may provide indication that certain variables are important for attaining extreme values of the output, with the *sign* of the skewness indices indicating whether the variables contribute significantly to output maximum (positive skewness index) or minimum (negative skewness index). However, they do not provide further theoretical justification for this claim.

The formulation described by Owen et al. lends itself straightforwardly to computation by Monte Carlo samples, but the high dimensionality of the integrals implies that many samples are required. In [56], skewness indices are defined by evaluating the third central moment of the Sobol' decomposition (3.2) and expanding the sum, resulting in indices that are different from those defined in [118]. The authors provide an approach for evaluating these indices via polynomial chaos, by expressing the indices in terms of polynomial coefficients. However, unlike the case for variance-based indices, the resultant indices cannot be expressed simply as a ratio of sum-squared coefficients. Here, the full result from [56] is cited for the skewness index of a set of variables indexed by S. Letting $\mathbf{p}, \mathbf{q}, \mathbf{r}$ be the corresponding multi-indices to the scalar indices p, q, r respectively, define the following sets

$$S^1 = \{p \,:\, \mathrm{nz}(\mathbf{p}) = S\},$$

$$S^2 = \{(p,q) \,:\, \mathrm{nz}(\mathbf{p}) \cup \mathrm{nz}(\mathbf{q}) = S\},$$

$$S^3 = \{(p,q,r) \,:\, \mathrm{nz}(\mathbf{p}) \cup \mathrm{nz}(\mathbf{q}) \cup \mathrm{nz}(\mathbf{r}) = S\}.$$

It can be shown that the skewness index for S is

$$t_S = \frac{1}{\sigma^{3/2}\gamma}\left(\sum_{p\in S^1} \alpha_{\mathbf{p}}^3 \mathbb{E}\left[v_{\mathbf{p}}^3\right] + 3\sum_{\substack{(p,q)\in S^2 \\ p\neq q}} \alpha_{\mathbf{p}}^2\alpha_{\mathbf{q}}\mathbb{E}\left[v_{\mathbf{p}}^2 v_{\mathbf{q}}\right] + 6\sum_{\substack{(p,q,r)\in S^3 \\ q>p}} \alpha_{\mathbf{p}}\alpha_{\mathbf{q}}\alpha_{\mathbf{r}}\mathbb{E}\left[v_{\mathbf{p}} v_{\mathbf{q}} v_{\mathbf{r}}\right]\right), \quad (3.21)$$

where γ is the total output skewness. The total skewness index for x_i is defined analogously to (3.7)

$$\hat{t}_i = \sum_{i\in S} t_S. \quad (3.22)$$

In the same work, the authors also establish the importance of these indices in modelling the tails of the output PDF, but make no explicit reference to sensitivity analysis near output extrema.

3.2.2 Extremum Sobol' indices

Although skewness indices pertain to the analysis of the tails of the output distribution, limitations exist with respect to the use of these indices for extremum sensitivity analysis. As a simple example, consider the following function

$$f(x_1, x_2) = x_1^2 + 100x_2, \qquad (x_1, x_2) \sim \mathcal{U}\left[-1, 1\right]^2, \tag{3.23}$$

where x_1 and x_2 are independently distributed. Referring to Figure 3.2, it can be seen that the original densities of x_1 and x_2 are symmetric about the mean, and thus have zero skewness. On the other hand, the output component x_1^2 is positively skewed, while $100x_2$ remain symmetric. It can be concluded that the skewness of $f(x_1, x_2)$ is entirely due to the effect of x_1^2. Thus, a skewness decomposition would show that the output skewness is entirely attributed to x_1. However, this does not correspond to the parameter that is the most important near the output maximum. Throughout the entire domain, x_2 has a dominant effect because of the large values of $100x_2$ compared to x_1^2. Thus, a skewness decomposition does not always indicate extremum sensitivities.

In this work, we propose a novel set of variance-based metrics that quantify the relative input sensitivities near regions in the domain yielding extreme output values, called extremum Sobol' indices. The variance decomposition of the function conditioned near its extrema is computed, producing indices that have a clear interpretation for extremum sensitivity analysis. To characterise the input distribution corresponding to output conditioning, a heuristic related to Monte Carlo filtering [128, p. 39] is used. In MCF, the input space is partitioned into two regions: one where the corresponding outputs fall in a range of interest B, and one where this does not happen, $\bar{B}$. The importance of a parameter x_i is determined by the difference between the distributions of x_i conditioned within B and its complement $\bar{B}$ respectively, where the distributions are inferred using Monte Carlo. Here, MCF is used to isolate regions containing points leading to output maximum and minimum, and the Sobol' indices of the quantity of interest are evaluated under the conditional distributions restricted to these regions. Sobol' indices resulting from points leading to output maxima are called *top Sobol' indices*, and those from points leading to output minima are called *bottom Sobol' indices*.

Computation of extremum Sobol' indices involves the following steps:

1. **Global polynomial approximation**: Fit a global polynomial approximation g to the model f by input-output pairs $\left\{\mathbf{x}^{(m)}, f(\mathbf{x}^{(m)})\right\}$ via sampling from $\mathcal{D}$ according to $\omega(\mathbf{x})$. Note that this global approximation can take the form of a ridge approximation.

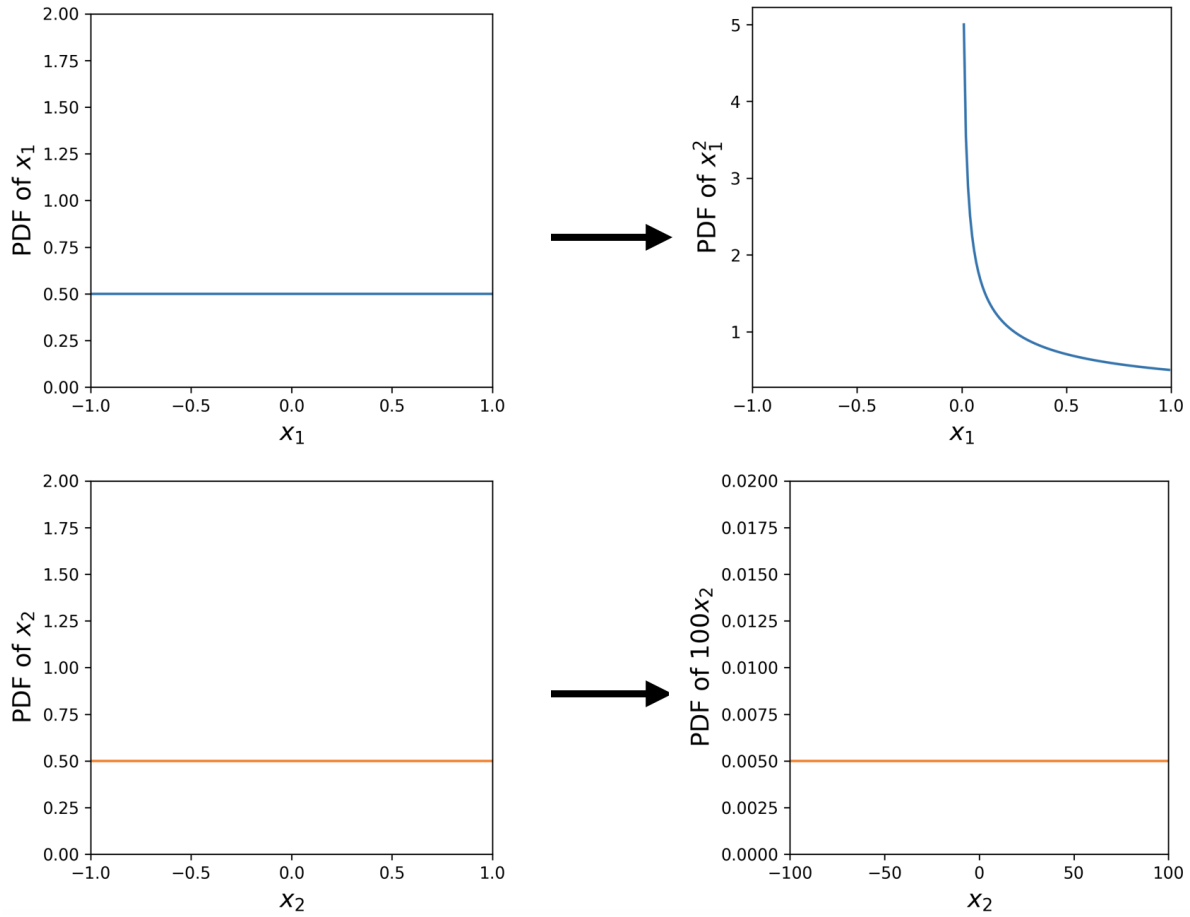


Figure 3.2 Probability densities of the random variables x_1 and x_2 and the densities of x_1^2 and $100x_2$. It can be seen that x_1^2 is positively skewed, but $100x_2$ is not skewed.

2. **Rejection sampling**: Generate another M' random input points in $\mathcal{D}$ according to $\omega(\mathbf{x})$. Estimate the maxima and minima by pushing these samples through the polynomial approximation. Isolate input-polynomial-output $\left\{\mathbf{x}^{(m)}, g(\mathbf{x}^{(m)})\right\}$ pairs that fall within the top 5% and bottom 5% of the maxima and minima respectively.

3. **Marginals and correlation determination**: Approximate the marginal distributions for each parameter for the two groups using input-output pairs. Fit a tailored bandwidth kernel density estimate to these marginals. Additionally, evaluate the empirical correlation matrix for these two groups.

4. **Extremum Sobol' indices**: Construct polynomial chaos approximations over the groups, using additional input-output pairs $\left\{\mathbf{x}^{(m)}, f(\mathbf{x}^{(m)})\right\}$ sampled from the correlated distribution over the respective groups. Then, use Algorithm 3.1 to compute the extremum Sobol' indices for each group.

For the final step, it is important to note that the extremum points resulting from MCF are in general distributed with a correlated measure, even if the original input measure is independent. In this case, the output variance cannot be decomposed in a similar manner to the Sobol' decomposition, and the indices computed from the definition (3.1) mixes the contribution of variables outside of the variables in S; the sum of all Sobol' indices thus does not result in unity. Therefore, extremum sensitivities are computed using the formulation proposed by Li et al. [98] which involves a covariance decomposition, upon which Navarro et al. [115] expanded with an algorithm based on polynomial chaos expansions. Algorithm 3.1 in Section 3.4.1 summarises the steps of this computation given extreme points isolated by MCF.

To calculate the extremum Sobol' indices using a polynomial chaos approximation, a polynomial basis orthogonal to the correlated measure determined in step 4 above is required. There are different methods to construct orthogonal polynomial bases in correlated spaces. For instance, one can map the dependent parameters to an independent space via the Nataf transform [101]. In Algorithm 3.1, the Gram-Schmidt method [158, 80] is used to generate evaluations on a polynomial basis orthogonal to the correlated measure. We start with the polynomial design matrix $\boldsymbol{A}$, with the M points of evaluation $\mathbf{x}^{(m)}$ sampled from the correlated measure. The basis polynomials here can be taken as ones orthogonal to the full (independent) input space measure $\omega(\mathbf{x})$. The sampling step can be performed via the Nataf transform. Then, the QR decomposition

$$\frac{1}{\sqrt{M}}\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R} \tag{3.24}$$

is calculated, where $\boldsymbol{Q}$ has orthogonal columns, and $\boldsymbol{R}$ is upper triangular. Because of the distribution of the sample points, this step is a Monte Carlo approximation to the orthogonalisation of the polynomial basis with respect to the inner product

$$\langle u_1, u_2 \rangle_{\omega_e} = \int u_1(\mathbf{x}) u_2(\mathbf{x})\, \omega_e(\mathbf{x})\, d\mathbf{x} \tag{3.25}$$

where ω_e is the correlated (extremum) measure and u_1 and u_2 are two polynomial basis functions. Thus, by post-multiplying a polynomial design matrix by the $\boldsymbol{R}$ matrix obtained here, one can obtain the polynomial design matrix corresponding to the same points on the basis orthogonal to ω_e. Polynomial coefficients can be found on this basis using least squares and Sobol' indices quantified via the coefficients. However, note that the Gram-Schmidt process produces basis polynomials that mix contributions from different parameters. Therefore, the Sobol' indices need to be calculated via Monte Carlo on the basis polynomials instead of being simply read out from (3.5). Note that for this algorithm, additional function evaluations are required only for finding the coefficients in the new polynomial basis; all Monte Carlo steps are performed on evaluations of the polynomial basis terms.

3.2.3 Numerical examples

In the following, we present three numerical examples where extremum sensitivities are evaluated using extremum Sobol' indices and skewness indices. Through empirical results, the parallels between the two indices in capturing important features near extreme output values are discussed.

3.2.3.1 Extremum sensitivity analysis of an analytical function

Consider the additive function $f : [0,1]^2 \to \mathbb{R}$ where

$$f(\mathbf{x}) = f_1(x_1) + f_2(x_2) = -x_1(x_1 - 2) + x_2^4, \qquad \mathbf{x} \sim \mathcal{U}[0,1]^2. \tag{3.26}$$

We calculate total sensitivity indices for this function via polynomial methods, which are exact in this case because the function is a polynomial[2]. Sobol' indices, skewness indices and extremum Sobol' indices are plotted for this function in Figure 3.3. In all bar chart figures in this section, the bar heights show the mean and error bars show

[2]However, Monte Carlo is still used to calculate the extremum Sobol' indices after polynomials are fitted near output extrema. Sufficient samples are used to ensure that the uncertainty is small.

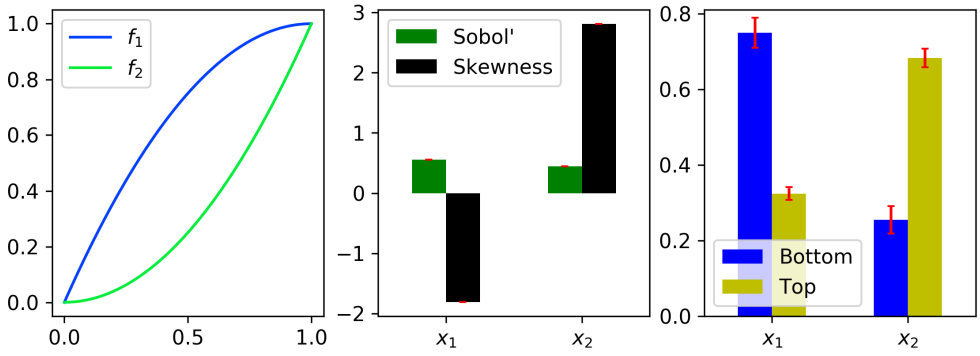


Figure 3.3 The plots for the two subfunctions f_1 and f_2 in (3.26) (left), and the associated total Sobol' and skewness (middle) and extremum Sobol' indices (right).

± 2 standard deviations over 30 trials[3]. It can be seen from the middle plot that both parameters have similar Sobol' indices, but there is a clear difference for the bottom and top Sobol' indices. For low output values, f_2 is nearly flat, implying that it is not influential to the function value; at high outputs, f_1 is relatively flat, implying that f_2 should be the more important variable. These pieces of information are revealed from the extremum Sobol' indices.

From the skewness indices, it can be seen that x_1 is responsible for skewing the output distribution to the right, and x_2 to the left. In this example, it can be observed that x_1, with a negative skewness index, is important near the output minimum, while x_2 is important near the output maximum, and has a positive skewness index. This agrees with the interpretation of skewness indices by Owen et al. [118] mentioned in Section 3.2.1, although it should be noted that their definition of skewness indices is different from the ones computed here. However, whether the sign of skewness indices is indicative of extremum sensitivities in general and its connection with extremum Sobol' indices remain open topics to be explored beyond the scope of this thesis.

3.2.3.2 Borehole function

In this case study, we apply methods of extremum sensitivity analysis on the borehole function, which models the steady-state water flow rate for a hypothetical borehole geometry [112]. The objective here is to identify which variable is responsible for the greatest variation over extreme values of the steady-state water flow. The model is specified by eight input parameters assumed to be independent. Their distributions

[3]Where positive quantities are plotted (e.g. Sobol' indices), the lower limit of the error bars is truncated at 0.

Table 3.1 Input parameters of the borehole function. Distribution parameters to a Gaussian distributed variable refer to the mean and standard deviation respectively; for a uniformly distributed variable, they refer to the lower and upper limits of the support.

Input	Distribution	Distribution parameters	Description
r_w	Gaussian	0.10, 0.0161812	Radius of borehole (m)
r	Uniform	100, 50000	Radius of influence (m)
T_u	Uniform	63070, 115600	Transmissivity of upper aquifer (m^2/yr)
H_u	Uniform	990, 1110	Potentiometric head of upper aquifer (m)
T_l	Uniform	63.1, 116	Transmissivity of lower aquifer (m^2/yr)
H_l	Uniform	700, 820	Potentiometric head of lower aquifer (m)
L	Uniform	1120, 1680	Length of borehole (m)
K_w	Uniform	1500, 15000	Hydraulic conductivity of borehole (m/yr)

are given in Table 3.1. The water flow rate depends on the parameters as

$$f(\mathbf{x}) = \frac{2\pi T_u (H_u - H_l)}{\ln\left(\frac{r}{r_w}\right)\left(1 + \frac{2LT_u}{\ln(r/r_w) r_w^2 K_w} + \frac{T_u}{T_l}\right)}. \tag{3.27}$$

Treating the water flow rate as the quantity of interest, we compute the total Sobol' and skewness indices with respect to the input variables. Three methods of computation are compared:

1. **(Poly)** Using an orthogonal polynomial least squares approximation on an isotropic basis with maximum total order of 3 (with a cardinality of 165), trained from 300 input-output pairs.

2. **(Ridge)** Using a polynomial ridge approximation of degree 3 in a 4-dimensional subspace. Here, the subspace dimension is found by trial-and-error, and the lowest value that results in acceptable performance is selected. In practice, methods such as cross validation (or visualisation with sufficient summary plots) can be used for determining the suitable dimensionality. That is, the dimension of the reduced space can be increased until polynomial fits on validation data are sufficiently accurate over the projected space.

   The coefficients of the polynomial in the projected subspace are subsequently converted for a polynomial on an isotropic basis with maximum total order of 3 by sampling in the input space and using (3.10). The subspace is found via polynomial variable projection using 150 input-output pairs. In this and the

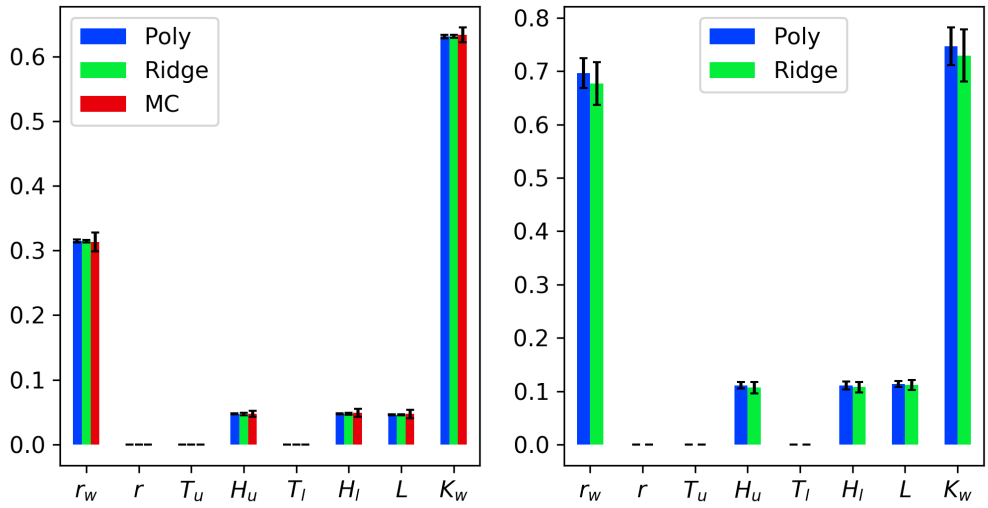


Figure 3.4 Total Sobol' (left) and skewness (right) indices for the borehole function (3.27).

next numerical examples, Legendre bases on tensor grids are used for fitting polynomials in projected spaces.

3. **(MC)** Using 10^5 model evaluations to calculate the indices via the Monte Carlo method by Sobol' [144, Thm. 2].

The training set $\{\mathbf{x}^{(m)}, f(\mathbf{x}^{(m)})\}$ is formed by sampling the inputs independently according to the corresponding marginal input distributions, and evaluating the corresponding output values. In Figure 3.4, results for the computations are shown. The third method is omitted for the skewness indices because no equivalent Monte Carlo formulation is proposed for skewness indices. The algorithm for computing skewness indices via polynomial chaos is explained in Section 3.4.2. From this figure, it can be seen that the Sobol' indices computed using a polynomial approximation match well with the Monte Carlo estimates, and using a ridge approximation in lieu of a full space polynomial reduces the amount of training data required while having minor effects on the accuracy of the computed indices. The choice of polynomial degrees for this and the next example is confirmed by perturbation studies.

Next, we compute the extremum Sobol' indices using the procedure described in Section 3.2.2. The total top and bottom Sobol' indices are calculated using MCF to select 5% of the input points yielding maximum and minimum outputs respectively

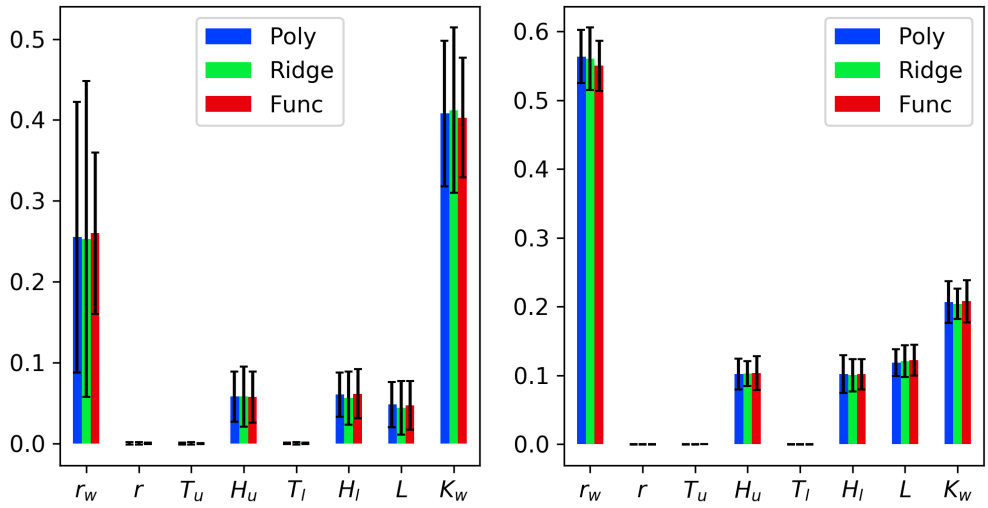


Figure 3.5 Bottom (left) and top (right) total Sobol' indices for the borehole function (3.27) using function evaluations (Func), a polynomial approximation (Poly) and a ridge approximation (Ridge).

out of 10^5 points. Three variants of the computation algorithm are tested, differing in the form of the model used to generate the pool of samples for MCF at step 2, namely

1. Using 10^5 function evaluations directly (instead of a polynomial approximation as specified in step 1);

2. Using a global polynomial model at total degree 4 (cardinality 495) fitted with 700 observations; and

3. Using a ridge polynomial at total degree 4 in a 4-dimensional subspace fitted with 400 observations.

Note that the polynomial degree is increased by one compared to the approximation models used to calculate global sensitivity indices previously in this section. This is to ensure better accuracy at the tails of the output. For all methods, an additional 1400 function evaluations are used to fit a degree 5 polynomial at the extrema.

The results over 30 trials are shown in Figure 3.5. It can be seen that near high outputs values, r_w has significantly increased importance compared to that predicted by full space Sobol' indices, along with somewhat increased importance for H_u, H_l and L. For low outputs, the importance ranking is similar to that evaluated from full space Sobol' indices. In both cases, it can be seen that no significant loss of accuracy

is incurred by using either polynomial or ridge approximations in place of function evaluations for MCF. In Figures 3.6 and 3.7, the marginal and pairwise distributions obtained from MCF are plotted, showing some correlation between r_w and K_w which is not present in the original full space input distribution. From the skewness indices, we see the increased importance of r_w, H_u, H_l and L with positive values. This accords with the interpretation of skewness indices mentioned in Section 3.2.3.1. The agreement is only qualitative however, with the relative ranking of r_w and K_w not in agreement.

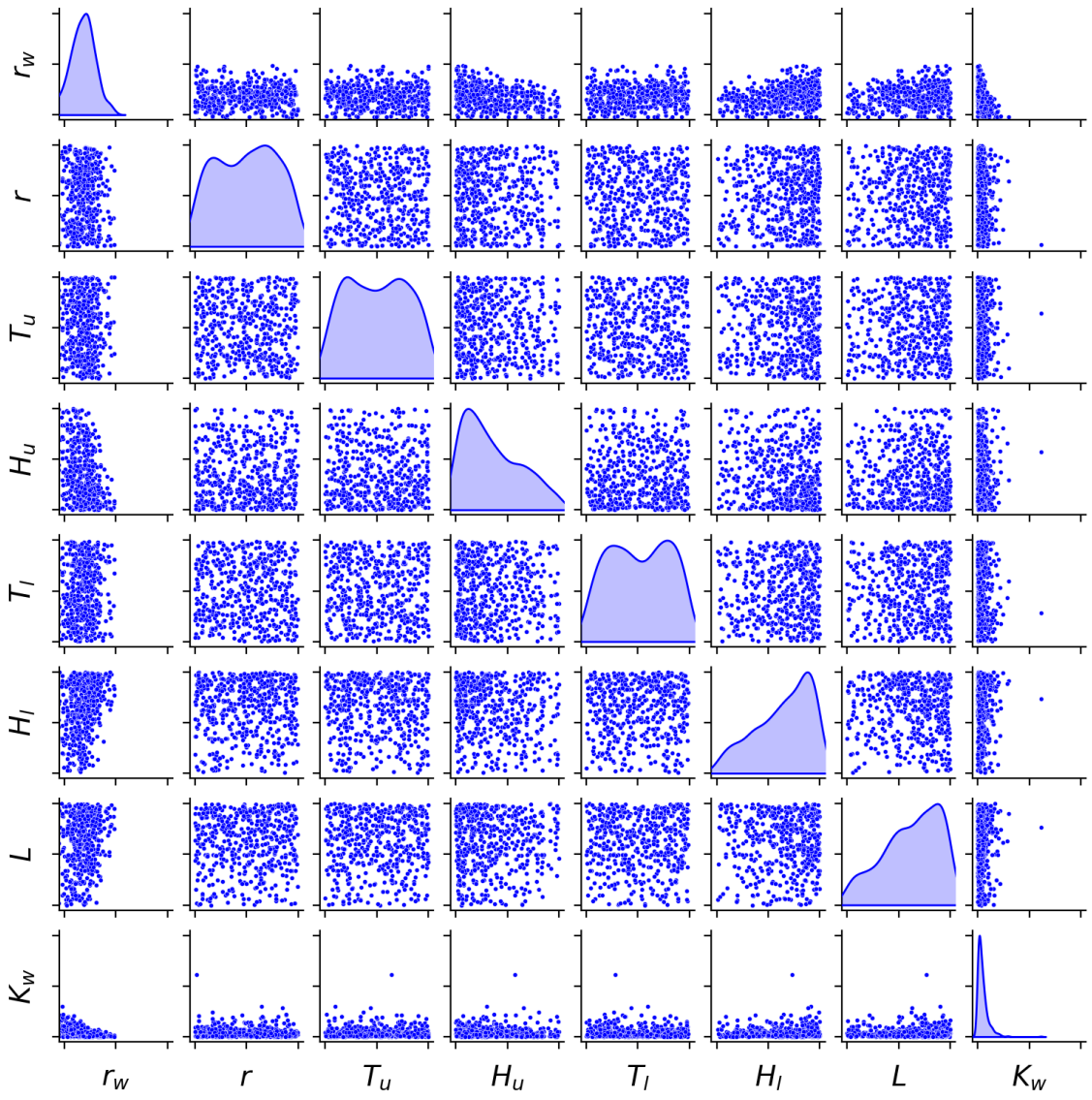


Figure 3.6 Pairwise scatter plot of isolated samples leading to low outputs for the borehole model (3.27).

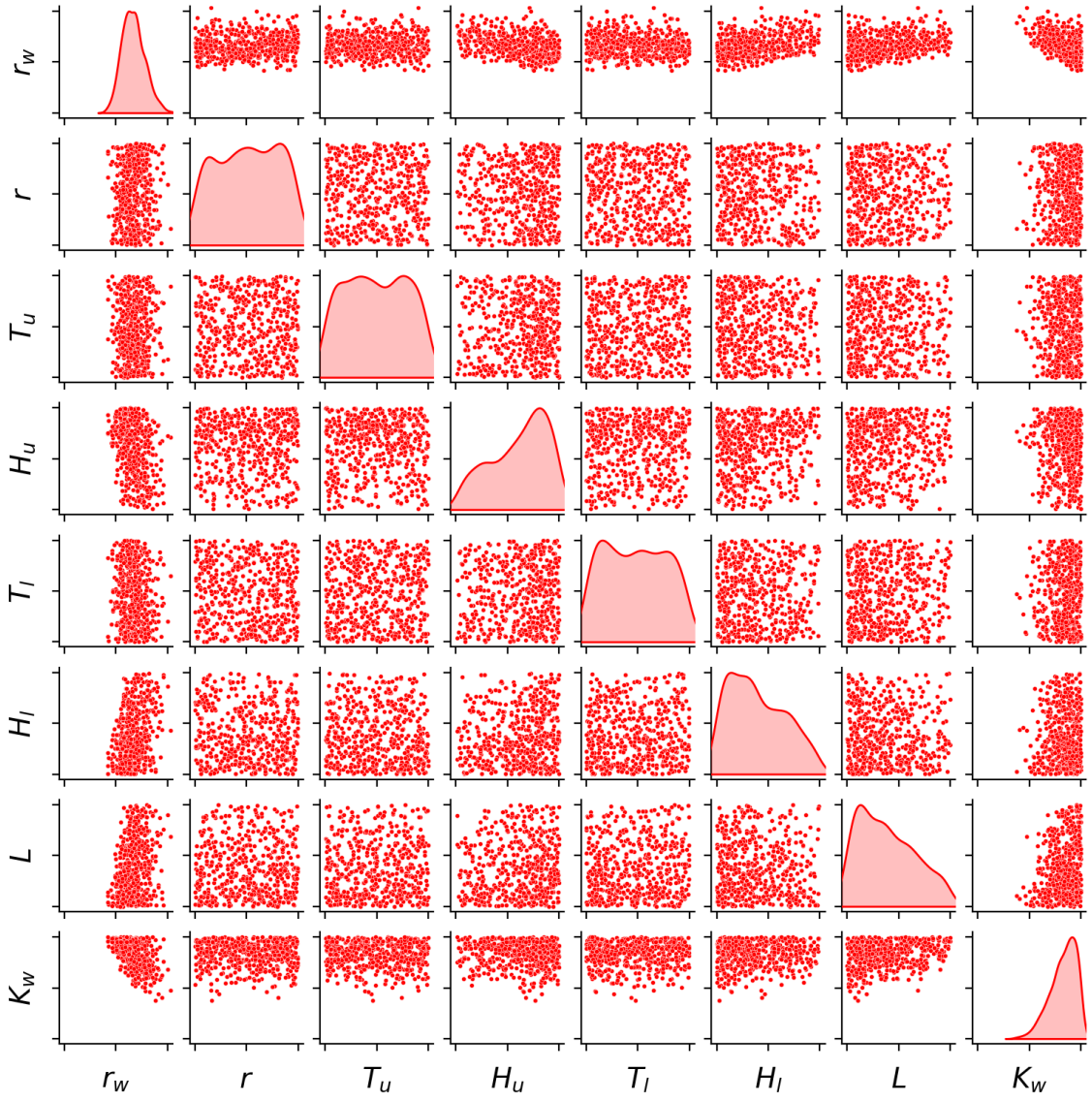


Figure 3.7 Pairwise scatter plot of isolated samples leading to high outputs for the borehole model (3.27).

Table 3.2 Input parameters of the piston model.

Input	Range	Description
M	[30, 60]	Piston mass (kg)
S	[0.005, 0.020]	Piston surface area (m^2)
V	[0.002, 0.010]	Initial gas volume (m^3)
k	[1000, 5000]	Spring coefficient (N/m)
P_0	[90000, 110000]	Atmospheric pressure (N/m^2)
T_a	[290, 296]	Ambient temperature (K)
T_0	[340, 360]	Filling gas temperature (K)

3.2.3.3 Piston cycle time function

Consider a model of a piston adapted from [88], which describes the cycle time of the piston as a function of seven parameters. These parameters are shown in Table 3.2. The inputs are independently and uniformly distributed over their support. As in the previous example, we are interested in obtaining the relative importance of each input variable near output extrema. The cycle time in seconds is

$$\tau(\mathbf{x}) = 2\pi \sqrt{\frac{M}{k + S^2 \frac{P_0 V T_a}{T_0 V^2}}}, \tag{3.28}$$

where

$$V = \frac{S}{2k}\left(\sqrt{A^2 + 4k\frac{P_0 V}{T_0}T_a} - A\right) \tag{3.29}$$

and

$$A = P_0 S + 19.62M - \frac{kV}{S}. \tag{3.30}$$

The total Sobol' and skewness indices are evaluated using the same three methods as in the previous example—Poly, Ridge and MC. For Poly, a degree 4 polynomial on a total order isotropic basis (cardinality 330) is fitted with 500 samples; for Ridge, a degree 4 polynomial over a 4-dimensional subspace is fitted with 300 samples and its coefficients. The results are shown in Figure 3.8, again showing that using ridge polynomial approximations incurs insignificant losses in accuracy.

The extremum Sobol' indices are evaluated using the same approaches as in the previous example. For Poly, we use a degree 5 polynomial (cardinality 792) trained from 1000 samples; for Ridge, a degree 5 polynomial over a 4-dimensional subspace trained with 700 samples is used. A degree 4 polynomial in the extrema is fitted with

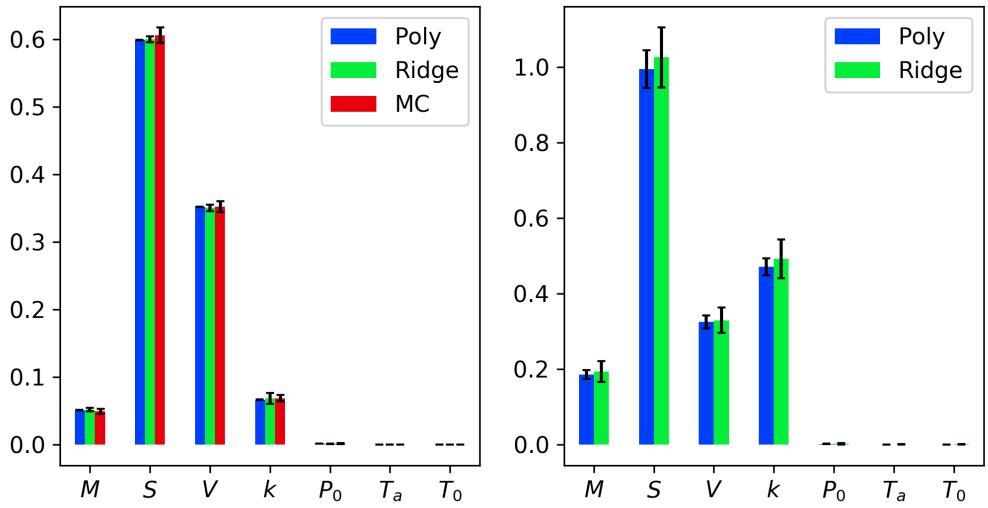


Figure 3.8 Total Sobol' (left) and skewness (right) indices for the piston function (3.28).

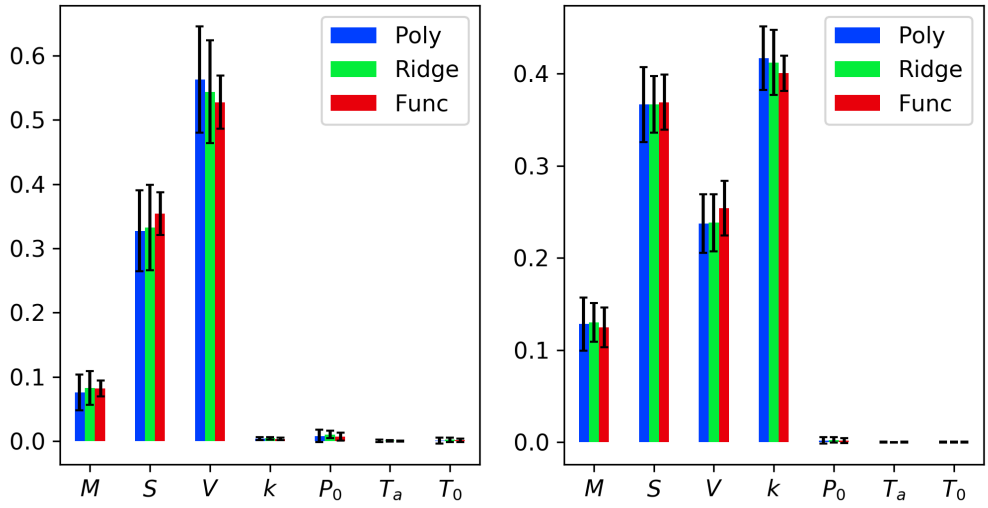


Figure 3.9 Bottom (left) and top (right) Sobol' indices for the piston function (3.28) using function evaluations (Func), a polynomial approximation (Poly) and a ridge approximation (Ridge).

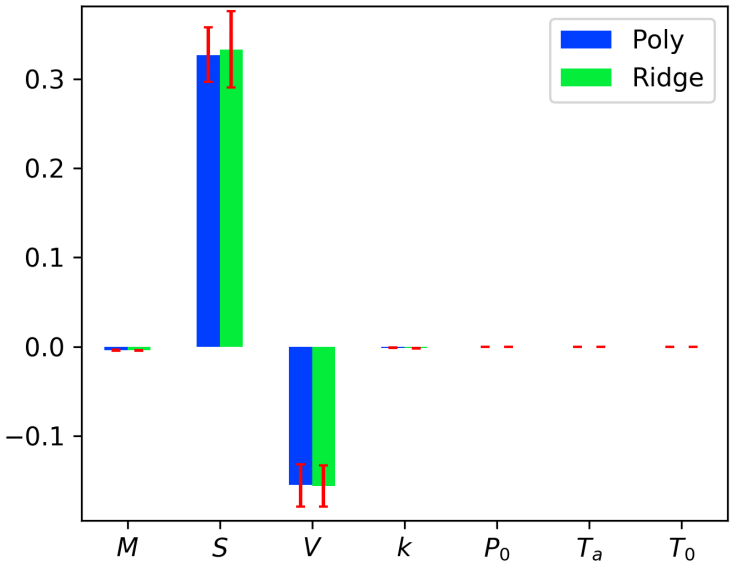


Figure 3.10 First-order skewness indices for the piston function (3.28).

an additional 500 training points for evaluating the indices. The results are shown in Figure 3.9. Compared with full space Sobol' indices, the bottom Sobol' indices show an increased importance of V in driving the output at low values. For high outputs, k becomes important. Examining the total skewness indices, an increased importance at maximum can be seen for k, which has a large positive total skewness index. The information for V is not found in the total skewness indices, but examining the *first-order* skewness indices (Figure 3.10), the importance of V at minimum is highlighted by the large negative value. From the top and bottom Sobol' indices, we find that this variable is important throughout the domain including the output extrema at both ends, but it is especially important at the minimum. With skewness indices, it is necessary to examine different orders to draw this conclusion. Figures 3.11 and 3.12 show the pairwise plots of the extrema PDFs, showing correlations between S, V and k near extrema, and justifying the use of polynomials taking this into account.

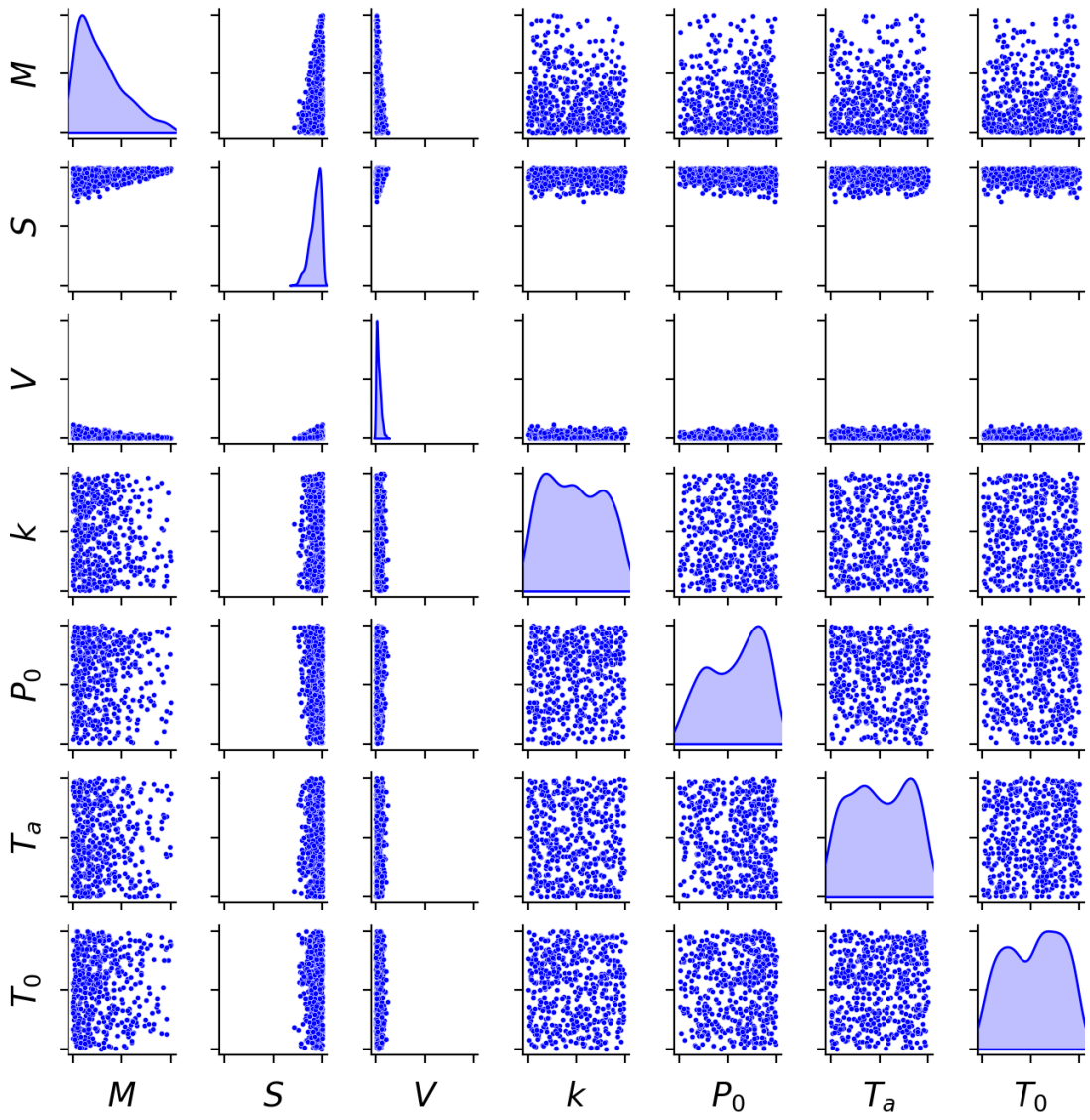


Figure 3.11 Pairwise scatter plot of isolated samples leading to low outputs for the piston model (3.28).

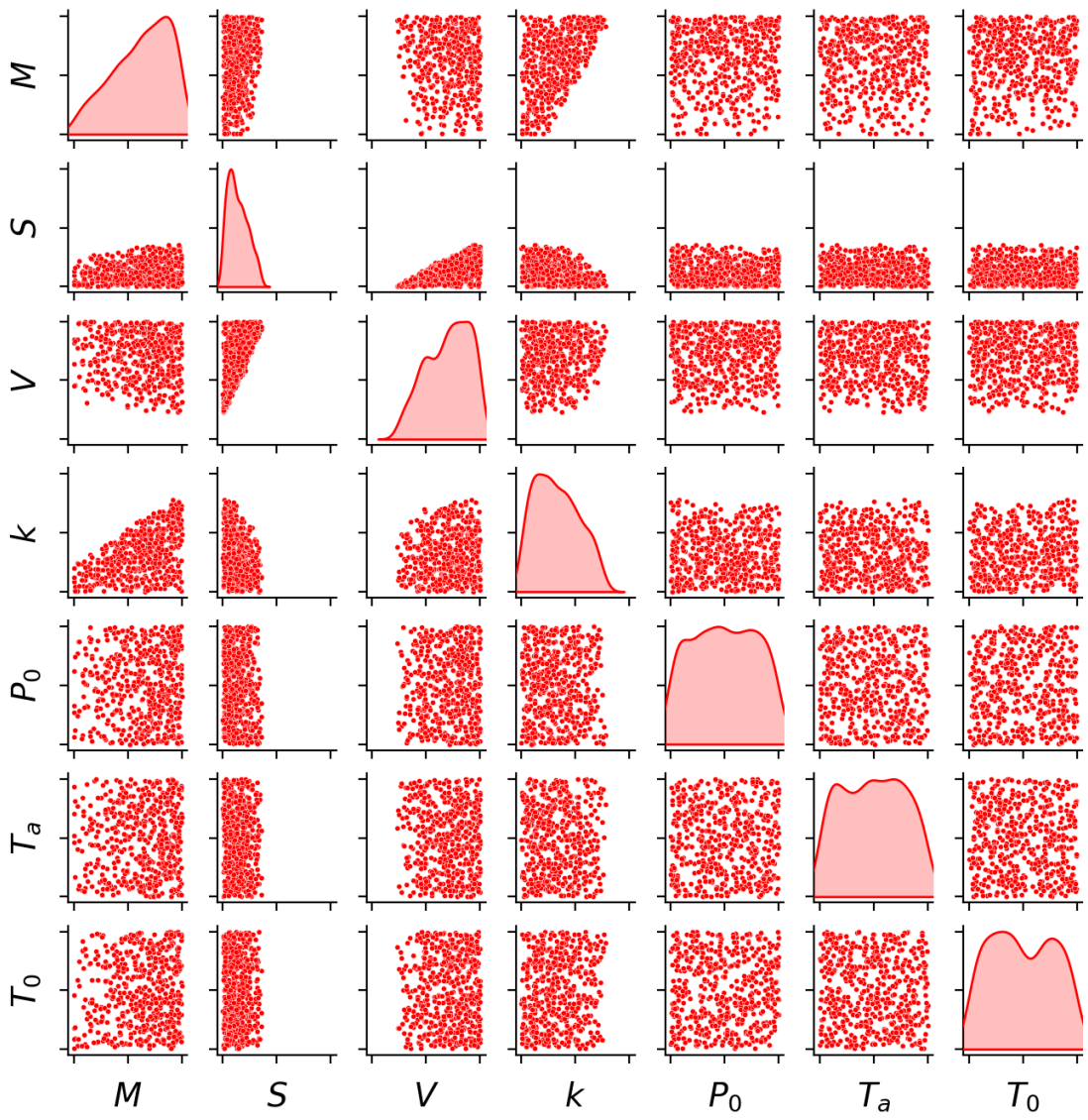


Figure 3.12 Pairwise scatter plot of isolated samples leading to high outputs for the piston model (3.28).

3.3 Conclusions for chapter

In this chapter, we focus on the application of polynomials and ridges to sensitivity analysis of models under input uncertainty. Through numerical examples, it is shown that polynomial ridge functions can be used to discover potential low-dimensional structures present in a model. In turn, sensitivity indices can be evaluated at a lower cost than a polynomial defined on the full input space if such structures exist. Moreover, extremum Sobol' indices are introduced and demonstrated for the task of sensitivity analysis near output extremes, noting empirical connections with skewness indices pertaining to the tails of the output moments. Polynomial approximations play a central role in the practical computation of these indices, and associated ridge approximations can facilitate the computations further.

3.4 Algorithm details

This section contains details for selected algorithms used in computing sensitivity indices.

3.4.1 Computing extremum Sobol' indices

The algorithm for computing extremum Sobol' indices after obtaining extremum samples from MCF is shown in Algorithm 3.1. Note that Sobol' indices beyond the first order need to be computed recursively by evaluating the associated quantities at lower orders (in line 15).

3.4.2 Computing skewness indices

As described by (3.21) in Section 3.2, the method for calculating skewness indices via polynomial expansion coefficients is more involved than calculating the Sobol' indices, because the integrals of third-order cross terms cannot be simplified easily with orthogonality. Hence, the expectation terms within each summand need to be evaluated using numerical quadrature. That is, one can approximate the expectation of a function $f(\mathbf{x})$ using Gauss quadrature

$$\mathbb{E}[f(\mathbf{x})] = \int_{\mathcal{D}} f(\mathbf{x})\ \omega(\mathbf{x})\ d\mathbf{x} \approx \sum_{q=1}^{M} f(\lambda^{(q)})\ \theta^{(q)}, \tag{3.35}$$

Algorithm 3.1 Computing extremum Sobol' indices

Input
    Extremum distribution $\omega_e(\mathbf{x})$, indices of the parameters of interest S.
Output
    Extremum Sobol' index σ_S^e.
Obtain M samples from $\omega_e(\mathbf{x})$ and form the polynomial design matrix

$$\boldsymbol{A}_u = \begin{bmatrix} v_1(\mathbf{x}^{(1)}) & v_2(\mathbf{x}^{(1)}) & \dots & v_P(\mathbf{x}^{(1)}) \\ v_1(\mathbf{x}^{(2)}) & v_2(\mathbf{x}^{(2)}) & \dots & v_P(\mathbf{x}^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ v_1(\mathbf{x}^{(M)}) & v_2(\mathbf{x}^{(M)}) & \dots & v_P(\mathbf{x}^{(M)}) \end{bmatrix}. \tag{3.31}$$

where (v_i) is an orthogonal polynomial basis with respect to the *product of the marginals* of $\omega_e(\mathbf{x})$.
Perform a QR decomposition of $(1/\sqrt{M})\boldsymbol{A}_u = \boldsymbol{Q}_u\boldsymbol{R}$, and form a new polynomial basis (u_i) where

$$u_i(\mathbf{x}) = \sum_{j=1}^{r} v_j(\mathbf{x})(\boldsymbol{R}^{-1})(j,i) \tag{3.32}$$

This basis is now orthogonal with respect to $\omega_e(\mathbf{x})$ [80].
Sample M' points and evaluate the function on these points ($\mathbf{f}_e$). Solve for the least squares coefficients

$$\underset{\mathbf{c}_e}{\text{minimise}} \quad \|\boldsymbol{A}_c\mathbf{c}_e - \mathbf{f}_e\|_2, \tag{3.33}$$

where $\boldsymbol{A}_c$ is the Vandermonde matrix based on (u_i).
Initialise empty matrix $\boldsymbol{B} \in \mathbb{R}^{M\times P}$.
for $i = 1, ..., P$ **do**
    Let $\mathbf{i}$ be the corresponding multi-index of (v_i).
    $\mathbf{i}_m = \mathbf{i}$ with its s-th element set to 0 for all $s \in S$.
    $\mathbf{i}_v =$ zero vector with its s-th element set to that of $\mathbf{i}$ for all $s \in S$.
    Set the i-th column of $\boldsymbol{B}$ as

$$\boldsymbol{B}(:,i) = \mu_{i_m}\boldsymbol{A}_u(:,i_v) \tag{3.34}$$

where μ_{i_m} is the mean of $\boldsymbol{A}_u(:,i_m)$.
end for
Calculate $\mathbf{m}_S = \boldsymbol{B}\boldsymbol{R}^{-1}\mathbf{c}_e - \sum_{T\subset S}\mathbf{m}_T$.
Calculate the empirical covariance $C_S = \mathrm{Cov}[\mathbf{m}_S, \boldsymbol{A}_c\mathbf{c}_e]$
$\sigma_S^e = C_S/\sigma$ where σ^e is the output variance according to ω_e, calculated by summing up the squares of the coefficients $\mathbf{c}_e$ without the first.

where $(\lambda^{(q)}, \theta^{(q)})_{q=1}^{M}$ are the quadrature points and weights. In Algorithm 3.2, the algorithm implemented for this work is described. In this algorithm, the *normalised multi-index* is defined corresponding to the set of variables indexed by S

$$\bar{\mathbf{j}} = (\bar{j_1}, ..., \bar{j_d}) \tag{3.36}$$

where $\bar{j_i} = 1$ if $i \in S$ and 0 otherwise, and $\odot$ denotes elementwise multiplication. In general, an $O(P^3)$ loop is required for calculating a skewness index, and since P can scale rapidly with the input dimension, this presents a significant computational burden. This algorithm incorporates several methods to reduce the computational time, namely:

- Using a selection function similar to that proposed in [56] to avoid computation of terms which we know are zero;
- Scanning the index set in an $O(P)$ loop before computation to filter out indices that involve more variables than specified. This cuts the cost of computation significantly for indices involving a small number of variables.
- Using a formulation amenable to matrix operations to avoid explicit loops.

Algorithm 3.2 Computing conditional skewness terms

Input
Polynomial basis and coefficients, quadrature points and weights $(\lambda^{(q)}, \theta^{(q)})_{q=1}^{M}$, set of variable indices S for desired skewness index.
Output
Skewness index t_S with corresponding normalised multi-index $\bar{\mathbf{j}}$.
Compute the weighted evaluation matrix $\boldsymbol{E}_w \in \mathbb{R}^{P\times M}$ such that

$$\boldsymbol{E}_w = \begin{bmatrix} c_1 v_1(\lambda^{(1)}) & c_1 v_1(\lambda^{(2)}) & \dots & c_1 v_1(\lambda^{(M)}) \\ c_2 v_2(\lambda^{(1)}) & c_2 v_2(\lambda^{(2)}) & \dots & c_2 v_2(\lambda^{(M)}) \\ \vdots & \vdots & \ddots & \vdots \\ c_P v_P(\lambda^{(1)}) & c_P v_P(\lambda^{(2)}) & \dots & c_P v_P(\lambda^{(M)}) \end{bmatrix}. \tag{3.37}$$

Sum every column to get the polynomial approximant at every point $g(\lambda_q)$:

$$g(\lambda^{(q)}) = \sum_{i=1}^{P} c_i v_i(\lambda^{(q)}) \tag{3.38}$$

Compute the global skewness $\gamma = \sum_{q=1}^{M} (g(\lambda^{(q)}))^3 \theta^{(q)}$ via a dot product.
Compute the global variance $\sigma = \sum_{i=2}^{P} c_i^2$.
Initialise set of valid indices $V = \varnothing$. ▷ Prune indices
for $a = 1, ..., P$ **do**
    Compute $ord = \sum_d \bar{a}_d$
    if $ord \leq \sum_d \bar{j}_d$ **then**
        $V \leftarrow V \cup \{a\}$.
    end if
end for
Set $r_1 = 0$. ▷ Calculate first sum term
for $a \in V$ **do**
    Compute $s_a = \sum_{q=1}^{M} (\mathbf{e}_a^3)_q \theta^{(q)}$ via a dot product, where $\mathbf{e}_a$ is the a-th column of $\boldsymbol{E}_w$.
    $r_1 \leftarrow r_1 + s_a/(\sigma^{3/2}\gamma)$.
end for
Set $r_2 = 0$. ▷ Calculate second sum term
for $a, b \in V$ **do**
    if ($a_d = 0$ and $b_d \neq 0$ for any d) or $a = b$ **then**
        Continue.
    end if
    Compute $s_{ab} = \sum_{q=1}^{M} (\mathbf{e}_a^2 \odot \mathbf{e}_b)_q \theta^{(q)}$ via a dot product.
    $r_2 \leftarrow r_2 + 3s_{ab}/(\sigma^{3/2}\gamma)$.
end for ▷ (Continued on the next page)

Set $r_3 = 0$. ▷ Calculate third sum term
for $a \in V, b \in \{a+1, ..., P\} \cap V, c = \{b+1, ..., P\} \cap V$ **do**
    if $\bar{a}_d + \bar{b}_d + \bar{c}_d = 1$ for any d **then**
        Continue.
    else if $\bar{a}_d + \bar{b}_d + \bar{c}_d = 2$ and $\bar{a}_d, \bar{b}_d, \bar{c}_d$ are pairwise distinct for any d **then**
        Continue.
    end if
    Compute $s_{abc} = \sum_{q=1}^{M} (\mathbf{e}_a \odot \mathbf{e}_b \odot \mathbf{e}_c)_q \theta^{(q)}$ via a dot product.
    $r_3 \leftarrow r_3 + 6 s_{abc} / (\sigma^{3/2} \gamma)$.
end for
Compute $t_S = r_1 + r_2 + r_3$.

Chapter 4

Multi-objective ridge approximations

This chapter presents the theory behind methods for dimension reduction of functions with multiple objectives, along with details of novel algorithms. Two example scenarios where one is interested in multi-objective functions are listed below.

- There is a set of multiple quantities of interest for a system we are interested in optimising or controlling simultaneously. Examples include the pressure ratio and stage efficiency of a compressor stage, and the stagnation pressure loss and the exit flow angle of a turbine row.
- One is interested in modelling a quantity defined over a finite spatial domain, such as the 2-D/3-D flow velocity field around an airfoil, or the temperature field of a fluid under the influence of heat sources. In this case, outputs are usually discretised and defined over a computational mesh.

Mathematically, multi-objective functions can be represented by a *vector-valued function* $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^N$ where each of the N components of the output corresponds to one quantity of interest, a scalar function of the inputs. The focus of this chapter is on the ridge approximation of these functions. At the basic level, this is the application of ridge approximation to each output, treating each component as a separate scalar objective,

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_N(\mathbf{x}) \end{bmatrix} \approx \begin{bmatrix} g_1(\boldsymbol{W}_1^\top \mathbf{x}) \\ g_2(\boldsymbol{W}_2^\top \mathbf{x}) \\ \vdots \\ g_N(\boldsymbol{W}_N^\top \mathbf{x}) \end{bmatrix}. \tag{4.1}$$

However, it can be challenging to store and use N ridge subspaces in surrogate models, especially if N is large. In fact, one is often interested in a single subspace that

encapsulates the behaviour of all components, and the presence of multiple subspaces poses a challenge in interpretation. In the following, we study techniques for multi-objective ridge approximations from two perspectives: invariance and emulation.

4.1 Multi-objective invariance

Finding regions within a parametric design space where quantities of interest remain within a small tolerance is useful for various engineering design tasks. We call these regions *invariant subspaces*. In this thesis, invariant subspaces are applied to design a procedure for performance-based manufacturing tolerancing, detailed in Chapter 5. However, one can envisage broader applications of this concept, such as algorithms for approximately constrained optimisation by allowing variation in the invariant subspaces of constrained quantities.

For a single scalar-valued objective, the invariant subspace can be straightforwardly determined through subspace-based dimension reduction, when it is interpreted as a partition of the design space into two mutually orthogonal subspaces. A vector of inputs $\mathbf{x}$ in the design space $\mathcal{D}$ can be written as

$$\begin{aligned} \mathbf{x} &= \boldsymbol{I}\mathbf{x} \\ &= \boldsymbol{Q}\boldsymbol{Q}^\top\mathbf{x} \\ &= [\boldsymbol{W}\ \boldsymbol{V}][\boldsymbol{W}\ \boldsymbol{V}]^\top\mathbf{x} \\ &= \boldsymbol{W}\boldsymbol{W}^\top\mathbf{x} + \boldsymbol{V}\boldsymbol{V}^\top\mathbf{x}, \end{aligned} \tag{4.2}$$

where $\boldsymbol{Q} \in \mathbb{R}^{d\times d}$ is an orthogonal matrix decomposed into $\boldsymbol{W} \in \mathbb{R}^{d\times n}$ and $\boldsymbol{V} \in \mathbb{R}^{d\times(d-n)}$. This expresses $\mathbf{x}$ in a new vector basis formed from the columns of $\boldsymbol{W}$ and $\boldsymbol{V}$, which is a rotation of the cardinal basis of $\mathbb{R}^d$. The goal is to find the pair $(\boldsymbol{W}, \boldsymbol{V})$ such that a QoI remains approximately constant when $\boldsymbol{W}^\top\mathbf{x}$ is fixed but $\boldsymbol{V}^\top\mathbf{x}$ is allowed to vary.

Several dimension reduction algorithms (such as active subspaces [32], polynomial variable projection [77] and Gaussian ridge functions [138]) seek subspace matrices $\boldsymbol{W}$ that minimise the expected $L_2(\omega)$ error

$$\left\| f(\mathbf{x}) - \mathbb{E}[f(\mathbf{x})|\boldsymbol{W}^\top\mathbf{x}] \right\|^2_{L_2(\omega)} = \int_{\mathcal{D}} \left(f(\mathbf{x}) - \mathbb{E}[f(\mathbf{x})|\boldsymbol{W}^\top\mathbf{x}] \right)^2 \omega(\mathbf{x})\, d\mathbf{x}, \tag{4.3}$$

or a discretised version of it. This can be interpreted as the mean squared error between the true function value $f(\mathbf{x})$ and the response surface within the dimension-reducing

subspace $\mathbb{E}[f(\mathbf{x})|\boldsymbol{W}^\top\mathbf{x}]$, i.e. the remaining output variation when the coordinates within the dimension-reducing subspace are fixed. Thus, when the columns of $\boldsymbol{W}$ span a dimension-reducing subspace, the quantity of interest can be expected to be approximately invariant when the coordinates $\boldsymbol{W}^\top\mathbf{x}$ are fixed, but with $\boldsymbol{V}^\top\mathbf{x}$ allowed to vary freely within the design space. The invariant subspace can then be found by taking the orthogonal complement of the column span of $\boldsymbol{W}$, e.g. through a full QR factorisation. We refer to the coordinates $\boldsymbol{W}^\top\mathbf{x}$ as the *active coordinates*, and $\boldsymbol{V}^\top\mathbf{x}$ as the *inactive coordinates*. Note that when the method of active subspaces is used, the invariant subspace is also known as the inactive subspace.

4.1.1 The intersection approach

Suppose now that we have multiple objectives $f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_N(\mathbf{x})$ to keep invariant near the values of $f_1(\bar{\mathbf{x}}), f_2(\bar{\mathbf{x}}), ..., f_N(\bar{\mathbf{x}})$ for some $\bar{\mathbf{x}} \in \mathcal{D}$, and the corresponding dimension-reducing subspaces/invariant subspaces are spanned by

$$(\boldsymbol{W}_1, \boldsymbol{V}_1), (\boldsymbol{W}_2, \boldsymbol{V}_2), ..., (\boldsymbol{W}_N, \boldsymbol{V}_N). \tag{4.4}$$

Consider an input vector $\mathbf{x}$ satisfying

$$\mathbf{x} = \bar{\mathbf{x}} + \delta, \qquad \delta \in \mathcal{V}_{int} := \bigcap_{i=1}^{N} \text{colspan}(\boldsymbol{V}_i), \tag{4.5}$$

i.e. the difference between $\mathbf{x}$ and $\bar{\mathbf{x}}$ lies in the *intersection* of the invariant subspaces $\mathcal{V}_{int}$. For $i = 1, 2, ..., N$,

$$\begin{aligned} \boldsymbol{W}_i^\top\mathbf{x} &= \boldsymbol{W}_i^\top\bar{\mathbf{x}} + \boldsymbol{W}_i^\top\delta \\ &= \boldsymbol{W}_i^\top\bar{\mathbf{x}} + \boldsymbol{W}_i^\top\boldsymbol{V}_i\boldsymbol{V}_i^\top\delta \\ &= \boldsymbol{W}_i^\top\bar{\mathbf{x}}, \end{aligned} \tag{4.6}$$

because $\boldsymbol{V}_i\boldsymbol{V}_i^\top\delta$ projects δ onto the column span of $\boldsymbol{V}_i$, but since $\delta \in \text{colspan}(\boldsymbol{V}_i)$ this does not change the vector. Moreover, $\boldsymbol{W}_i^\top\boldsymbol{V}_i = \mathbf{0}$ by orthogonality of the basis. Therefore, $\mathbf{x}$ is identical to $\bar{\mathbf{x}}$ when pre-multiplied by $\boldsymbol{W}_1^\top, \boldsymbol{W}_2^\top, ..., \boldsymbol{W}_N^\top$, and their active coordinates are identical with respect to all N objectives. This implies that the differences between $f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_N(\mathbf{x})$ and $f_1(\bar{\mathbf{x}}), f_2(\bar{\mathbf{x}}), ..., f_N(\bar{\mathbf{x}})$ are small.

To generate sample points satisfying the invariance criterion (4.5), a basis for the invariant subspace is required. For a single objective $i = 1$, this can be taken as the

columns of $\boldsymbol{V}$. For multiple objectives, a basis for the subspace $\mathcal{V}_{int}$ is needed. Note that this intersection is a subspace of $\mathbb{R}^d$, a fact explained in Section 6.4 of [60]. This implies that it is possible to find a basis matrix $\boldsymbol{V}_{int}$ with columns that span $\mathcal{V}_{int}$ when its dimension is larger than zero, i.e. contains more than the zero vector. The condition that

$$\dim(\text{colspan}(\boldsymbol{W}_1) \oplus \text{colspan}(\boldsymbol{W}_1) \oplus ... \oplus \text{colspan}(\boldsymbol{W}_N)) = r < d \tag{4.7}$$

ensures that the dimension of $\mathcal{V}_{int}$ is larger than 0, where $\oplus$ denotes the direct sum. To find $\boldsymbol{V}_{int}$, one can use the following recipe:

1. Form $\boldsymbol{W}_{\oplus} = [\boldsymbol{W}_1 \ \boldsymbol{W}_2 \ ... \ \boldsymbol{W}_N]$.
2. Decompose $\boldsymbol{W}_{\oplus} = \boldsymbol{QR}$ by QR-decomposition.
3. Set $\boldsymbol{V}_{int}$ to be the last $(d - r)$ columns of $\boldsymbol{Q}$.

Section 6.4 of [60] discusses algorithms based on singular value decompositions to find r. However, for a small number of independent objectives, r can be determined by inspection and is usually the sum of the number of active coordinates for all objectives.

4.1.2 Vector-valued invariance

A limitation of the intersection approach to finding invariant subspaces for multiple objectives is the assumption that (4.7) is satisfied. When the number of objectives N is comparable to or larger than the input dimension d, this condition is unlikely to be true. This can occur if the set of output objectives corresponds to a physical quantity defined on a discretised spatial field. In this case, the intersection subspace $\mathcal{V}_{int}$ is trivial (contains only the zero vector). At a high level, we require a subspace where *approximate* invariance less stringent than that given by the intersection approach is achieved.

In Zahm et al. [163], dimension reduction of vector-valued functions is considered by studying solutions to a controlled approximation problem. They consider finding a ridge profile $\mathbf{g}$ and ridge directions $\boldsymbol{W}$ such that the approximation error is bounded,

$$\left\| \mathbf{f}(\mathbf{x}) - \mathbf{g}(\boldsymbol{W}^\top \mathbf{x}) \right\|_V \leq \varepsilon, \tag{4.8}$$

for some $\varepsilon \geq 0$ that should be kept small. This is similar to the case for scalar objectives except that the ridge profile is a vector-valued function. A salient difference here is the definition of the function norm used to quantify the approximation error, which is

defined as

$$\|\mathbf{u}\|_V = \sqrt{(\mathbf{u},\mathbf{u})_V}, \qquad (\mathbf{u},\mathbf{w})_V = \int \mathbf{u}(\mathbf{x})^\top \, \boldsymbol{R} \, \mathbf{w}(\mathbf{x}) \, \omega(\mathbf{x}) \, d\mathbf{x}. \tag{4.9}$$

Here, $\boldsymbol{R} \in \mathbb{R}^{N\times N}$ is a symmetric positive semi-definite *weight matrix*. The resultant dimension-reducing subspace is dependent on the definition of $\boldsymbol{R}$, which influences the approximation error metric. For a fixed $\boldsymbol{W}$, it is shown that the conditional expectation function

$$\mathbf{g}(\mathbf{x}) = \mathbb{E}[\mathbf{f} \mid \boldsymbol{W}^\top \mathbf{x}] \tag{4.10}$$

is the unique minimiser of the approximation error $\left\|\mathbf{f}(\mathbf{x}) - \mathbf{g}(\boldsymbol{W}^\top \mathbf{x})\right\|_V$; an analogous result is known for the scalar case [31]. Similar to the active subspace method, the authors arrive at a procedure that allows an expression for ε to be written in terms of the dimension-reducing subspace represented by $\boldsymbol{W}$, guaranteeing the boundedness of the approximation error. This is achieved using a Poincaré inequality which bounds the average deviation of a continuously differentiable function (a property assumed for each output component) with the average norm of the gradient. From that, an expression for ε can be formulated by defining the *vector gradient covariance matrix*

$$\boldsymbol{H} = \int_{\mathcal{D}} \boldsymbol{J}(\mathbf{x}) \, \boldsymbol{R} \, \boldsymbol{J}(\mathbf{x})^\top \, \omega(\mathbf{x}) \, d\mathbf{x}, \tag{4.11}$$

where $\boldsymbol{J}(\mathbf{x})$ is the Jacobian matrix,

$$\boldsymbol{J} = \left[\frac{\partial f_1}{\partial \mathbf{x}}, \frac{\partial f_2}{\partial \mathbf{x}}, \ldots, \frac{\partial f_N}{\partial \mathbf{x}}\right] \in \mathbb{R}^{d\times N}. \tag{4.12}$$

Proposition 2.6 of [163] shows that the approximation error can be bounded by

$$\left\|\mathbf{f}(\mathbf{x}) - \mathbf{g}(\boldsymbol{W}^\top \mathbf{x})\right\|_V \leq \sqrt{\sum_{i=n+1}^{d} \lambda_i}, \tag{4.13}$$

where λ_i is the i-th eigenvalue of $\boldsymbol{H}$, when the dimension-reducing subspace (the column span of $\boldsymbol{W}$) is taken to be the span of the first n eigenvectors of $\boldsymbol{H}$ by partitioning the eigendecomposition with eigenvalues sorted in descending order,

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{W} & \boldsymbol{V} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Lambda}_1 & \\ & \boldsymbol{\Lambda}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{W}^\top \\ \boldsymbol{V}^\top \end{bmatrix}. \tag{4.14}$$

Again, an analogous result is established for active subspaces (2.52), so this method can be seen as a generalisation of active subspaces to multiple objectives. Note that in its original form in [163], a Gaussian distribution is assumed for the input vector, which may account for input correlations via a covariance matrix $\boldsymbol{\Sigma}$ parameterising the Gaussian. The result is stated in terms of the generalised eigenpairs of the matrix pair $(\boldsymbol{H}, \boldsymbol{\Sigma})$. However, a similar result can be established for distributions defined on convex input domains using results from [5] (e.g. uniform distribution on the hypercube), arriving at a form analogous to (2.52) with an additional proportionality constant.

Equipped with a dimension-reducing subspace by taking the leading eigenvectors of $\boldsymbol{H}$, one can derive the invariant subspace of $\mathbf{f}$ by taking the orthogonal complement of $\boldsymbol{W}$, spanned by the columns of $\boldsymbol{V}$ in (4.14). It is worthwhile to compare this to the invariant subspace obtained with the intersection method. Instead of its dimensionality being set by the direct sum of the dimension-reducing subspaces of each objective, the invariant subspace depends on the eigenvalue decay of the vector gradient covariance matrix. Taking only the subspace corresponding to *small* eigenvalues of $\boldsymbol{H}$, the invariant subspace obtained here represents a notion of *weak* invariance that is less stringent than one obtained through the intersection method. Depending on application, the threshold between *small* and *non-small* can be tuned, giving more flexibility in implementation.

An important difference between scalar active subspaces and vector-valued dimension reduction is the presence of the weight matrix $\boldsymbol{R}$ defining the output norm in (4.9). One example in Section 5.2.2 of [163] defines $\boldsymbol{R}$ as a diagonal matrix

$$\boldsymbol{R} = \begin{bmatrix} \theta_1 & & & \\ & \theta_2 & & \\ & & \ddots & \\ & & & \theta_N \end{bmatrix}, \tag{4.15}$$

where $\theta_i \geq 0$ for $i = 1, 2, ..., N$. This has a natural interpretation of weighing the importance of each of the N output components. In fact, with $\boldsymbol{R}$ as a diagonal matrix, $\boldsymbol{H}$ is the accordingly weighted sum of the gradient covariance matrices of each output objective with the same input distribution $\omega(\mathbf{x})$. That is, if

$$\boldsymbol{C}_i = \int_{\mathcal{D}} \nabla f_i(\mathbf{x}) \, \nabla f_i(\mathbf{x})^\top \, \omega(\mathbf{x}) \, d\mathbf{x}, \tag{4.16}$$

then

$$\boldsymbol{H} = \sum_{i=1}^{N} \theta_i \boldsymbol{C}_i. \tag{4.17}$$

The larger the weight θ_i for f_i, the greater its influence on the eigenvectors defining the subspaces. Stronger invariance is expected for output components where the corresponding weights are large compared to those with smaller weights. This formulation allows further flexibility to control the degree of tolerance for different output components. Where possible, one can relax invariance constraints for output components which are not critical.

4.2 Embedded ridge approximations

One class of vector-valued functions of particular interest consists of parameterised scalar fields defined on a spatial domain. These often arise from modelling the governing physics using systems of partial differential equations (PDEs) under the influence of a set of parameters. These parameters can represent boundary conditions, initial conditions and the geometry of the domain. When the PDE system is discretised on a finite mesh within the spatial domain, it admits a natural representation as a vector-valued function of the parameters. To understand the effects of input parameters on the field for tasks such as uncertainty quantification, design optimisation and sensitivity analysis, surrogate models are often used as emulators of the computational model. This is due to the sizeable number of parameters required to capture the effects of geometric variability, as well as the high computational and storage costs of running simulation models repeatedly.

Emulation of scalar fields is not a novel area of research, and a range of methods and algorithms have been developed in the past. One prominent example is proper orthogonal decomposition (POD) [140, 7], which consists of applying PCA to a set of snapshots of a time-dependent simulation to construct a reduced-order surrogate model. A key feature that is leveraged in POD is the presence of spatial and temporal correlations within the flow field, which can also be interpreted as the presence of *smoothness* in the spatio-temporal variation encoded within the model physics. This is one motivation behind the use of convolutional neural networks (CNNs) as surrogate models in recent works [71, 8, 148]. CNNs have been deployed with great success in computer vision and image processing because of correlation structures arising from smoothness. Another approach is to train a physics-informed neural network by evaluating loss functions against analytical PDE expressions [125, 105].

In this work, we examine the approximation of parameterised scalar fields using polynomial ridge functions defined over discrete computational nodes, which we call *embedded ridge approximations*. As will be shown in this section, the advantages of using polynomial ridges include the following:

- **Synthesised emulators for integral quantities of interest**: Placing assumptions on the local properties of the flow field, it is shown that embedded ridge approximations can be used to derive low-dimensional approximations for scalar quantities of interest that can be expressed as spatial integrals of an underlying field.
- **Compression of flow field representations**: By exploiting spatial smoothness, a procedure for compressing flow field approximations, which retains only select nodes of importance, is introduced.

4.2.1 Approximation of scalar fields and their integrals

We start by defining the parameterised scalar field $f(\mathbf{x}, \mathbf{s})$, where $\mathbf{s} \in \mathbb{R}^K$ denotes the spatial location in K-dimensional space. Apart from assumptions on the variation of f with respect to $\mathbf{x}$ delineated in Chapter 2, assume also that, for each $\mathbf{s}$, $f(\mathbf{x}, \mathbf{s})$ is Lipschitz continuous and has square integrable bounded partial first derivatives with respect to $\mathbf{x}$. A vector-valued function can be defined on a discrete mesh with N nodes $(\mathbf{s}_i)_{i=1}^N$,

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} f(\mathbf{x}, \mathbf{s}_1) \\ f(\mathbf{x}, \mathbf{s}_2) \\ \vdots \\ f(\mathbf{x}, \mathbf{s}_N) \end{bmatrix} = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_N(\mathbf{x}) \end{bmatrix}. \tag{4.18}$$

We wish to find ridge approximations at each node. For $i = 1, 2, ..., N$, the ridge profile $g_i : \mathbb{R}^{n_i} \to \mathbb{R}$ and ridge subspace matrix $\boldsymbol{W}_i \in \mathbb{R}^{d \times n_i}$ can be computed such that

$$f_i(\mathbf{x}) \approx g_i(\boldsymbol{W}_i^\top \mathbf{x}), \tag{4.19}$$

using a set of input/output points $\left\{\mathbf{x}^{(m)}, \mathbf{f}(\mathbf{x}^{(m)})\right\}$. This can be achieved with a scalar-valued model-based dimension reduction method, such as those reviewed in Section 2.2.4.

Now, consider a quantity $h(\mathbf{x})$ that can be expressed as

$$h(\mathbf{x}) = \int_{\mathcal{D}} \theta(\mathbf{s})\, f(\mathbf{x}, \mathbf{s})\, d\mathbf{s} \tag{4.20}$$

Table 4.1 Example scalar fields and integral quantities of interest.

Scalar field $f(\mathbf{x}, \mathbf{s})$	Integral quantities of interest $h(\mathbf{x})$
Static pressure	Lift and drag coefficients
Pressure and velocity	Isentropic efficiency loss
Stagnation flow quantities	Area/Mass averaged flow quantities
Local displacement	Total displacement

where $\theta : \mathbb{R}^K \to \mathbb{R}$ is a spatial weight function. The integral can be approximated by evaluation on the mesh, expressing it as a weighted sum of the output components,

$$h(\mathbf{x}) \approx \sum_{i=1}^{N} \theta_i f(\mathbf{x}, \mathbf{s}_i). \tag{4.21}$$

Examples of such *integral quantities of interest* are listed in Table 4.1. Here, we are interested in constructing ridge approximations of these quantities. Suppose that we are equipped with the ridge subspaces of the scalar field at each node as in (4.19) and the spatial weights θ_i, how can we combine these ridge subspaces to compute a ridge subspace for $h(\mathbf{x})$?

Before continuing, we note that for the purpose of *emulation*, i.e. constructing a low-cost surrogate model to obtain approximate outputs rapidly, it is sufficient to simply evaluate the embedded ridge approximations $g_1(\boldsymbol{W}_1^\top \mathbf{x}), g_2(\boldsymbol{W}_2^\top \mathbf{x}), ..., g_N(\boldsymbol{W}_N^\top \mathbf{x})$, which are low-cost surrogates, and compute an approximation for $h(\mathbf{x})$ using the weighted sum expression. However, the utility of ridge approximations extends beyond surrogate modelling. Ridge subspaces can aid optimisation [67, 106], visualisation [36] (if the subspace is one- or two-dimensional), sensitivity analysis (see Chapter 3) and the discovery of physical insights [41]. The work in [131] provides studies that illustrate the design utility of sufficient summary plots from embedded ridge approximations of various quantities in a flow field. Moreover, as mentioned in Section 4.1 and will be demonstrated in the next chapter, dimension-reducing subspaces can be used to explore regions of the design space yielding output invariance, which finds utility in tolerance design.

The present approach is based on evaluating the gradient covariance matrix of $h(\mathbf{x})$,

$$\boldsymbol{C}_h = \mathbb{E}\left[\nabla_{\mathbf{x}} h(\mathbf{x}) \nabla_{\mathbf{x}} h(\mathbf{x})^\top\right], \tag{4.22}$$

since the eigendecomposition of this matrix can be used to identify the active subspace for $h(\mathbf{x})$, a dimension-reducing subspace with certified approximation accuracy de-

pending on eigenvalue decay. The subscript $\mathbf{x}$ is added to ∇ to indicate differentiation with respect to $\mathbf{x}$. This matrix can be directly estimated when provided with a sample of gradient evaluations ($\nabla_{\mathbf{x}} h(\mathbf{x}^{(m)})$) with methods such as adjoints. However, in this section we do not assume that automatic differentiation methods are available. Instead, from the embedded ridge approximations on the mesh, one can write

$$\nabla_{\mathbf{x}} h(\mathbf{x}) \approx \sum_{i=1}^{N} \theta_i \nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{s}_i) \approx \sum_{i=1}^{N} \theta_i \boldsymbol{W}_i \nabla^i_{\mathbf{u}} g_i(\boldsymbol{W}_i^\top \mathbf{x}), \tag{4.23}$$

where

$$\nabla^i_{\mathbf{u}} g_i(\boldsymbol{W}_i^\top \mathbf{x}) = \frac{\partial g_i(\mathbf{u}_i)}{\partial \mathbf{u}_i}, \quad \mathbf{u}_i = \boldsymbol{W}_i^\top \mathbf{x}. \tag{4.24}$$

So, the covariance matrix $\boldsymbol{C}_h$ can be expressed as

$$\begin{aligned} \boldsymbol{C}_h &= \mathbb{E}\left[\nabla_{\mathbf{x}} h(\mathbf{x}) \nabla_{\mathbf{x}} h(\mathbf{x})^\top\right] \\ &\approx \mathbb{E}\left[\left(\sum_{i=1}^{N} \theta_i \boldsymbol{W}_i \nabla^i_{\mathbf{u}} g_i\left(\boldsymbol{W}_i^\top \mathbf{x}\right)\right)\left(\sum_{j=1}^{N} \theta_j \boldsymbol{W}_j \nabla^j_{\mathbf{u}} g_j\left(\boldsymbol{W}_j^\top \mathbf{x}\right)\right)^\top\right] \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} \theta_i \theta_j \boldsymbol{W}_i \mathbb{E}\left[\nabla^i_{\mathbf{u}} g_i \nabla^j_{\mathbf{u}} g_j^\top\right] \boldsymbol{W}_j^\top. \end{aligned} \tag{4.25}$$

The expression in the last line provides the basis of an algorithm for finding the active subspaces for integral QoIs. Instead of directly approximating the dimension-reducing subspace of h through input/output pairs (direct approximation), ridge approximations are first formed for the scalar field underlying the computation of the QoI, whose ridge subspaces and profiles are then used to evaluate the gradient covariance matrix of $h(\mathbf{x})$.

In fact, the matrix $\boldsymbol{C}_h$ can be expressed as a special case of the vector gradient covariance matrix $\boldsymbol{H}$ for $\mathbf{f}$ (see (4.11)). Setting

$$\boldsymbol{R} = \boldsymbol{\theta}\boldsymbol{\theta}^\top, \quad \text{where } \boldsymbol{\theta} = [\theta_1, \theta_2, ..., \theta_N]^\top, \tag{4.26}$$

it follows that

$$\begin{aligned}
\boldsymbol{H} &= \mathbb{E}\left[\boldsymbol{J}(\mathbf{x})\ \boldsymbol{\theta}\boldsymbol{\theta}^{\top}\ \boldsymbol{J}(\mathbf{x})^{\top}\right] \\
&= \mathbb{E}\left[\left(\sum_{i=1}^{N}\theta_i \nabla f_i\right)\left(\sum_{j=1}^{N}\theta_j \nabla f_j\right)^{\top}\right] \\
&\approx \mathbb{E}\left[\left(\sum_{i=1}^{N}\theta_i \boldsymbol{W}_i \nabla^i_{\mathbf{u}} g_i\right)\left(\sum_{j=1}^{N}\theta_j \boldsymbol{W}_j \nabla^j_{\mathbf{u}} g_j\right)^{\top}\right] \\
&= \sum_{i=1}^{N}\sum_{j=1}^{N}\theta_i\theta_j \boldsymbol{W}_i \mathbb{E}\left[\nabla^i_{\mathbf{u}} g_i \nabla^j_{\mathbf{u}} g_j^{\top}\right]\boldsymbol{W}_j^{\top} \approx \boldsymbol{C}_h.
\end{aligned} \tag{4.27}$$

This provides a concise way of summarising the evaluation of $\boldsymbol{C}_h$ using nodal ridge subspaces. In Algorithm 4.1, the procedure for calculating the ridge approximation of $h(\mathbf{x})$ is described in terms of the associated vector gradient covariance matrix of the underlying scalar field. The finite sample estimate of a quantity is denoted by placing a hat $\widehat{\cdot}$ on the symbol.

Algorithm 4.1 Embedded ridge function approximation.

Input

Input/output pairs $\left\{\mathbf{x}^{(m)}, \mathbf{f}^{(m)}\right\}_{m=1}^{M}$, $\mathbf{f}^{(m)} = \left[f_1\left(\mathbf{x}^{(m)}\right), ..., f_N\left(\mathbf{x}^{(m)}\right)\right]^\top$, and output weight vector $\boldsymbol{\theta}$.

Output

Ridge profile $\widehat{g}$ and ridge directions $\widehat{\boldsymbol{W}}$ such that $h(\mathbf{x}) \approx \widehat{g}\left(\widehat{\boldsymbol{W}}^\top \mathbf{x}\right)$

for $i = 1, ..., N$ **do**

Find $\widehat{\boldsymbol{W}}_i$ using a gradient-free ridge approximation strategy.[1]

Fit an approximate ridge profile $\widehat{g}_i$ using

$$\left\{\left(\widehat{\boldsymbol{W}}_i^\top \mathbf{x}^{(1)}, f_i^{(1)}\right), \ldots, \left(\widehat{\boldsymbol{W}}_i^\top \mathbf{x}^{(M)}, f_i^{(M)}\right)\right\} \tag{4.28}$$

as training data.

Evaluate $\nabla_{\mathbf{x}} \widehat{f}_i\left(\mathbf{x}^{(m)}\right) = \widehat{\boldsymbol{W}}_i \nabla^i_{\mathbf{u}} \widehat{g}_i(\widehat{\boldsymbol{W}}_i^\top \mathbf{x}^{(m)})$ for $m = 1, ..., M$.

end for

Form $\widehat{\boldsymbol{J}}\left(\mathbf{x}^{(m)}\right)$ from (4.12).

Calculate

$$\widehat{\boldsymbol{C}_h} = \frac{1}{M} \sum_{m=1}^{M} \widehat{\boldsymbol{J}}\left(\mathbf{x}^{(m)}\right) \boldsymbol{R}\, \widehat{\boldsymbol{J}}\left(\mathbf{x}^{(m)}\right)^\top \tag{4.29}$$

where

$$\boldsymbol{R} = \boldsymbol{\theta}\boldsymbol{\theta}^\top. \tag{4.30}$$

Find the eigendecomposition of $\widehat{\boldsymbol{C}_h}$ with descending eigenvalues,

$$\widehat{\boldsymbol{C}_h} = \begin{bmatrix} \widehat{\boldsymbol{W}} & \widehat{\boldsymbol{V}} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\Lambda}_1} & \\ & \widehat{\boldsymbol{\Lambda}_2} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{W}}^\top \\ \widehat{\boldsymbol{V}}^\top \end{bmatrix}, \tag{4.31}$$

and choose the leading eigenvectors corresponding to the largest eigenvalues $\widehat{\boldsymbol{\Lambda}_1}$ to form $\widehat{\boldsymbol{W}}$.

Fit a low-dimensional ridge approximation $\widehat{g}$ using

$$\left\{\left(\widehat{\boldsymbol{W}}^\top \mathbf{x}^{(1)}, \boldsymbol{\theta}^\top \mathbf{f}^{(1)}\right), \ldots, \left(\widehat{\boldsymbol{W}}^\top \mathbf{x}^{(M)}, \boldsymbol{\theta}^\top \mathbf{f}^{(M)}\right)\right\} \tag{4.32}$$

as training data.

[1] Note that one can use the same set of input values for each component.

4.2.1.1 Accuracy of embedded ridge approximation

When can ridge subspaces of integrated QoIs be found more efficiently via embedded ridge approximations than direct approximation? Placing the assumption that the field has a *localised range of influence*, embedded ridge approximations can arrive at a ridge approximation for h more efficiently than direct approximation when the input parameters $\mathbf{x}$ represent variations of local spatial properties. Summarised briefly, this is when the field quantity at a given location is only strongly affected by changes in input parameters representing nearby perturbations, and the dependence of the field quantity on the input parameters is strongly anisotropic. Examples of this include the following.

- Over an airfoil in subsonic flow, the static pressure of a point near the leading edge is unlikely to be strongly affected by small geometric perturbations far downstream. Supposing that separation of flow does not occur, the effect of small changes in boundary geometry decays as a function of distance, especially if the geometry deviation is downstream of the flow measurement point.
- The local deflection of a structure is unlikely to be affected by small local changes in elastic properties far from the point of measurement. This is because the local strain rate is dependent on local material properties.

The presence of this anisotropy implies the presence of a low-dimensional parameterisation that can be discovered by dimension reduction methods. While this anisotropy of dependence may not be present in integral QoIs, it can be physically motivated in their corresponding scalar fields.

A scalar field with localised range of influence contains node quantities that can be well-approximated by low-dimensional ridge subspaces, as their functional dependence on input perturbations can be summarised with fewer parameters. This implies that the subspace sought by ridge approximations working on embedded computational nodes tends to be of a lower dimensionality than those with integral quantities. As a result, the desired ridge subspaces can be found more easily; e.g. optimisation steps in polynomial variable projection benefit from working within a Grassmann manifold with fewer parameters.

In the finite sample regime, the accuracy of embedded ridge approximation can be related to the accuracy to which node subspaces are found. Below, we establish a bound on the quantity $\mathbb{E}\left\|\widehat{\boldsymbol{C}_h} - \boldsymbol{C}_h\right\|_2$ in terms of the number of training sample points M and the accuracy to which embedded ridge subspaces are found, $\left\|\widehat{\boldsymbol{W}_i} - \boldsymbol{W}_i\right\|_2$, assuming that the ridge subspaces are of constant dimension n spatially.

Theorem 4.2.1. *Assume that* $\nabla^i_{\mathbf{u}} g_i$ *defined in* (4.24) *satisfies* $\left\|\nabla^i_{\mathbf{u}} g_i\right\|_2 \leq L$ *for all* $1 \leq i \leq N$,

$$M \geq \frac{2L^2 \log(2r)}{\varepsilon^2 \left\|\mathbb{E}\left[\nabla^i_{\mathbf{u}} g_i \nabla^j_{\mathbf{u}} g_j^\top\right]\right\|_2}, \tag{4.33}$$

for all $1 \leq i, j \leq N$, *and*

$$\left\|\widehat{\boldsymbol{W}}_i - \boldsymbol{W}_i\right\|_2 \leq \eta_M,$$

with $\boldsymbol{W}_i \in \mathbb{R}^{d \times n}$ *for all* $i = 1, 2, ..., N$. *Then,*

$$\mathbb{E}\left\|\widehat{\boldsymbol{C}}_h - \boldsymbol{C}_h\right\|_2 \leq L^2 C\left(2\eta_M + \varepsilon + \varepsilon^2\right),$$

where $C := \sum_{ij} |\theta_i \theta_j|$.

Proof. By the matrix Bernstein inequality, it can be shown that (4.33) implies that [154, Section 1.6.3]

$$\mathbb{E}\left\|\frac{1}{M}\sum_{m=1}^{M} \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} - \mathbb{E}[\nabla^i_{\mathbf{u}} g_i \nabla^j_{\mathbf{u}} g_j^\top]\right\|_2 \leq (\varepsilon + \varepsilon^2)\left\|\mathbb{E}[\nabla^i_{\mathbf{u}} g_i \nabla^j_{\mathbf{u}} g_j^\top]\right\|_2, \tag{4.34}$$

where $\nabla^i_{\mathbf{u}} g_i^{(m)}$ is the estimate of $\nabla^i_{\mathbf{u}} g_i$ at $\mathbf{x}^{(m)}$. We can then write

$$\begin{aligned}
\mathbb{E}\left\|\widehat{\boldsymbol{C}}_h - \boldsymbol{C}_h\right\|_2 &= \mathbb{E}\left\|\frac{1}{M}\sum_{m=1}^{M}\sum_{ij} \theta_i\theta_j \widehat{\boldsymbol{W}}_i \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \widehat{\boldsymbol{W}}_j^\top - \sum_{ij} \theta_i\theta_j \boldsymbol{W}_i \mathbb{E}[\nabla^i_{\mathbf{u}} g_i \nabla^j_{\mathbf{u}} g_j^\top] \boldsymbol{W}_j^\top\right\|_2 \\
&= \mathbb{E}\left\|\sum_{ij} \theta_i\theta_j \boldsymbol{E}_{ij}\right\|_2 \\
&\leq \sum_{ij} |\theta_i\theta_j| \mathbb{E}\left\|\boldsymbol{E}_{ij}\right\|_2,
\end{aligned}$$

where

$$
\begin{aligned}
\boldsymbol{E}_{ij} &= \frac{1}{M}\sum_{m=1}^{M} \widehat{\boldsymbol{W}}_i \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \widehat{\boldsymbol{W}}_j^\top - \boldsymbol{W}_i \mathbb{E}[\nabla^i_{\mathbf{u}} g_i \nabla^j_{\mathbf{u}} g_j^\top] \boldsymbol{W}_j^\top \\
&= \frac{1}{M}\sum_{m=1}^{M} \left(\widehat{\boldsymbol{W}}_i \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \widehat{\boldsymbol{W}}_j^\top - \boldsymbol{W}_i \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \widehat{\boldsymbol{W}}_j^\top \right) \\
&\quad + \frac{1}{M}\sum_{m=1}^{M} \left(\boldsymbol{W}_i \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \widehat{\boldsymbol{W}}_j^\top - \boldsymbol{W}_i \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \boldsymbol{W}_j^\top \right) \\
&\quad + \frac{1}{M}\sum_{m=1}^{M} \left(\boldsymbol{W}_i \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \boldsymbol{W}_j^\top \right) - \boldsymbol{W}_i \mathbb{E}[\nabla^i_{\mathbf{u}} g_i \nabla^j_{\mathbf{u}} g_j^\top] \boldsymbol{W}_j^\top \\
&= \left(\widehat{\boldsymbol{W}}_i - \boldsymbol{W}_i \right) \left(\frac{1}{M}\sum_{m=1}^{M} \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \right) \widehat{\boldsymbol{W}}_j^\top \\
&\quad + \boldsymbol{W}_i \left(\frac{1}{M}\sum_{m=1}^{M} \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \right) \left(\widehat{\boldsymbol{W}}_j^\top - \boldsymbol{W}_j^\top \right) \\
&\quad + \boldsymbol{W}_i \left(\frac{1}{M}\sum_{m=1}^{M} \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} - \mathbb{E}[\nabla^i_{\mathbf{u}} g_i \nabla^j_{\mathbf{u}} g_j^\top] \right) \boldsymbol{W}_j^\top .
\end{aligned}
$$

Note that

$$
\mathbb{E}\left\| \frac{1}{M}\sum_{m=1}^{M} \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \right\|_2 \leq \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}\left\| \nabla^i_{\mathbf{u}} g_i^{(m)} \nabla^j_{\mathbf{u}} g_j^{(m)\top} \right\|_2 \leq L^2 \tag{4.35}
$$

because $\nabla^i_{\mathbf{u}} g_i^{(m)}$ are identically distributed copies of $\nabla^i_{\mathbf{u}} g_i$ and their norms are upper bounded by L. Using the triangle inequality and sub-multiplicativity of the norm, we get

$$
\sum_{ij} |\theta_i \theta_j| \mathbb{E}\left\| \boldsymbol{E}_{ij} \right\|_2 \leq C \left(2\eta_M L^2 + \left(\varepsilon + \varepsilon^2 \right) L^2 \right) \tag{4.36}
$$

from which the theorem follows. □

There are several remarks to make regarding (4.33). It should be clear that the number of samples M required scales as a function of n instead of d. This encapsulates the advantage brought about by the localised range of influence of the scalar field. It is possible to derive a bound related to the subspace error of $\widehat{\boldsymbol{W}}$ in Algorithm 4.1 as a corollary to Theorem 4.2.1. This involves steps very similar to Lemma 3.9 and Corollary 3.10 in [28], which are based on Corollary 8.1.11 in [60].

4.2.2 Compression of embedded ridge approximations

Assuming spatial smoothness of the underlying field, it is likely that neighbouring nodes will have similar values of these quantities, depending on the overall resolution of the mesh. More specifically, one can think of these quantities as being strongly correlated with their neighbours. When approximating each component f_i as a ridge function, this correlation can be interpreted as similarity in both the ridge directions $\boldsymbol{W}_i$ and the ridge profile g_i. In this section, we assume that the number of ridge directions for each $\boldsymbol{W}_i$ is constant, and equal to n, for simplicity.

Given *a priori* knowledge of the similarity between neighbouring ridge directions, the embedded ridges of a scalar field can be compressed to avoid storing the ridge directions for all output components. This is useful for re-creating large scalar fields from a smaller subset of nodes. To this goal, we propose the *ridge compression and recovery* algorithms. The former allows one to retain only a subset of suitably subsampled output nodes; the latter recovers the remaining nodes from these subsampled nodes. Related work by Ji et al. [83] proposes a procedure to find a unified ridge subspace for multiple objectives. Instead of yielding one subspace, our approach focuses on adaptively selecting a problem-dependent subset of subspaces to effectively represent the entire field.

The algorithm for ridge compression is detailed in Algorithm 4.2, where each removed component will be reconstructed by the average of two of its closest neighbours, measured by the *subspace distance* metric. For two subspaces of $\mathbb{R}^d$, $\mathcal{S} = \text{colspan}(\boldsymbol{W})$ and $\widetilde{\mathcal{S}} = \text{colspan}(\widetilde{\boldsymbol{W}})$, the subspace distance is given by

$$dist(\mathcal{S}, \widetilde{\mathcal{S}}) = \left\| \boldsymbol{W}\boldsymbol{W}^\top - \widetilde{\boldsymbol{W}}\widetilde{\boldsymbol{W}}^\top \right\|_2. \tag{4.37}$$

The subspace distance is invariant to the choice of the basis matrices as long as they have orthonormal columns.

Given an embedded ridge approximation and the number of components that need to be removed k, the algorithm iterates through all N nodes and identifies two neighbours per node. These neighbours are chosen based on the smallest subspace distance between successive nodes; see lines 10 and 11 in Algorithm 4.2. In line 11, it is required that the second closest neighbour must be closer to the candidate to be removed than the first neighbour in line 10. If this step is not enforced, the average between the neighbours is a poor approximation of the removed candidate. Following this, in line 14, candidates for removal are sorted by considering the sum of their distances to their two neighbours. Removing a candidate with smaller total distance is

prioritised over removing one with larger total distance. From line 15 onwards, the candidates are removed according to the order determined in line 14. The recovery algorithm (Algorithm 4.3) reconstructs the missing components based on the list of nearest neighbours (the output from Algorithm 4.2). We only consider the case where $n = 1$ here, permitting us to estimate the missing node's ridge subspace as a linear combination of the neighbouring components.

Algorithm 4.2 Ridge compression algorithm for embedded ridge approximations.

1: **Input**
2: List of ridge directions $\boldsymbol{W}_1, ..., \boldsymbol{W}_N$ corresponding to $g_1, ..., g_N$ (but the ridge profiles are not needed), and the number of components to retain k.
3: **Output**
4: List of subsampled ridge directions $\boldsymbol{W}_{N_1}, ..., \boldsymbol{W}_{N_k}$, and a list of nearest neighbours $L \in \mathbb{N}^{(N-k)\times 2}$ corresponding to missing components $m \in \mathbb{N}^{N-k}$.
5: Initialise empty list m and array L, and $I_s = (1, ..., N)$.
6: **while** length of m is smaller than $N - k$ and $I_s \neq \varnothing$ **do**
7: $I' = I_s \backslash (m \cup L)$[2] ▷ Gather the remaining non-removed, non-paired components.
8: $A = L \cup I'$. ▷ Gather the available neighbours.
9: **for** i up to the length of I' **do**
10: $L'[i, 1] = \text{argmin}_{j \in A \backslash i} dist(\boldsymbol{W}_i, \boldsymbol{W}_j)$ ▷ Find the best neighbours for each index.
11: $L'[i, 2] = \text{argmin}_{j \in A \backslash \{L'[i,1], i\}} dist(\boldsymbol{W}_i, \boldsymbol{W}_j)$ s.t. $dist(\boldsymbol{W}_i, \boldsymbol{W}_j) < dist(\boldsymbol{W}_j, \boldsymbol{W}_{L'[i,1]})$
12: $D'[i] = dist(\boldsymbol{W}_i, \boldsymbol{W}_{L'[i,1]}) + dist(\boldsymbol{W}_i, \boldsymbol{W}_{L'[i,2]})$. ▷ Compute total distances.
13: **end for**
14: Sort I' and L' columnwise in ascending order of D' to give I_s and L_s.
15: **for** i up to the length of I_s **do**
16: **if** $I_s[i] \notin m \cup L$ and $L_s[i, 1], L_s[i, 2] \notin m$ **then**
17: $m \leftarrow m \cup \{I_s[i]\}$.
18: $L \leftarrow L \cup \{L_s[i, 1], L_s[i, 2]\}$.
19: **end if**
20: **end for**
21: **end while**
22: $k = N - \text{length}(m)$.[3]
23: $(N_1, \ldots, N_k) = (1, ..., N) \backslash m$.
24: Retain $\boldsymbol{W}_{N_1}, ..., \boldsymbol{W}_{N_k}$ and discard the rest.

[2] Note that lists are converted to sets before set operations are performed on them; that is, duplicate elements are removed and the order which was present in the list is no longer enforced. For a two-dimensional array, we flatten the array and consider all distinct elements.

Algorithm 4.3 Ridge recovery algorithm for a compressed embedded ridge approximation.

Input
  Subsampled points $(N_1, ..., N_k)$, list of matrices $\boldsymbol{W}_{N_1}, \ldots, \boldsymbol{W}_{N_k} \in \mathbb{R}^d$, and a list of nearest neighbours $L \in \mathbb{N}^{(N-k)\times 2}$ corresponding to missing components $m \in \mathbb{N}^{N-k}$.
Output
  Recovered approximate ridge subspace matrices $\boldsymbol{U}_1, \ldots, \boldsymbol{U}_N$.
for $i = 1, ..., N$ **do**
  if $i \in (N_1, \ldots, N_k)$ **then**
    $\boldsymbol{U}_i = \boldsymbol{W}_i$.
  else
    Find j such that $m[j] = i$.
    $\boldsymbol{U}'_{i1} = \boldsymbol{W}_{L[j,1]} + \boldsymbol{W}_{L[j,2]}$
    $\boldsymbol{U}'_{i2} = \boldsymbol{W}_{L[j,1]} - \boldsymbol{W}_{L[j,2]}$
    $P = \text{argmin}_{p=1,2} dist(\boldsymbol{U}'_{ip}, \boldsymbol{W}_{L[j,1]})$
    $\boldsymbol{U}_i = \text{column normalise}(\boldsymbol{U}'_{iP})$
  end if
end for

In the ridge compression algorithm, once a node is marked as one of the neighbours of a removed node, it can no longer be removed. This sets a hard limit on how many nodes can be removed before all stored nodes are marked. One way to circumvent this difficulty is to apply the compression and recovery algorithms *recursively*. Figure 4.1 illustrates this idea. At each compression stage, up to S components are compressed, where S can be set by the user. After this, the remaining components are input to the next stage to remove up to a further S components and so on. To recover the removed components, the recovery algorithm is applied stagewise, similar to the compression process, in the reverse direction. Note that upon passing the remaining components to the next stage, even though some of the remaining components are neighbours to removed components in the previous stage, it is possible to remove them in the next stage. This is because their ridge directions are not required until the corresponding stage in the recovery process, when these components will be reconstructed by the previous recovery stage. In this way, the extent of compression can be increased.

Both the ridge compression and recovery algorithms presented in this section are greedy algorithms, which may therefore not result in the storage configuration that globally minimises the distance between the missing components and their neighbours.

[3]This may be larger than the input k because no further components can be removed without compromising the neighbours of the already removed ones.

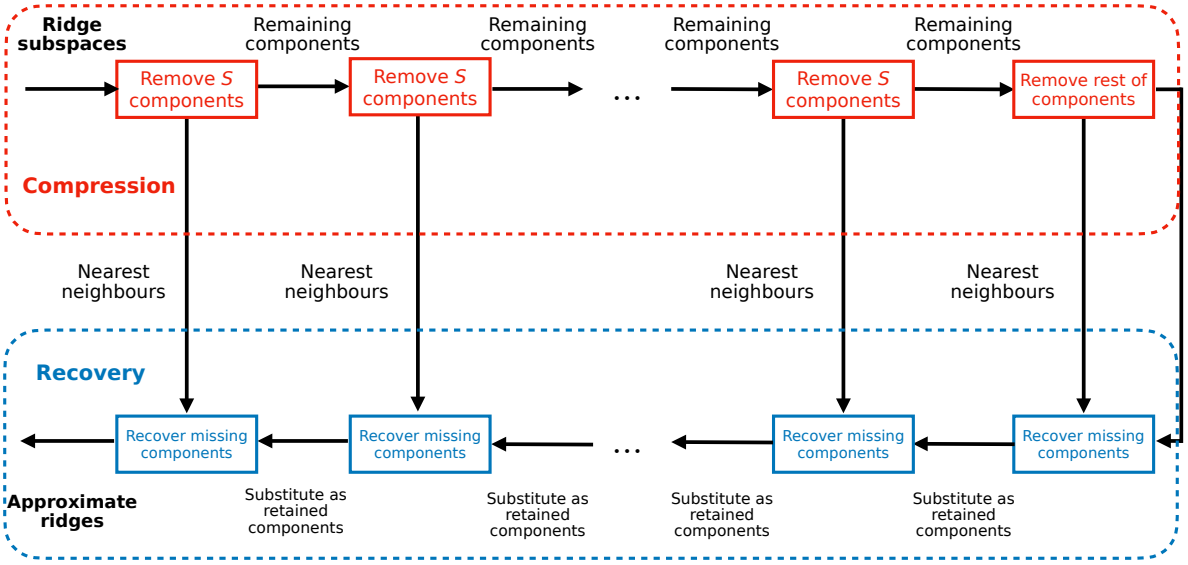


Figure 4.1 Schematic for applying the ridge compression algorithm (Algorithm 4.2) and recovery algorithm (Algorithm 4.3) recursively, removing at most S components at a time.

However, to determine the globally optimal solution requires a combinatorial search over every storage configuration, which is computationally prohibitive.

Note that the compression problem can be interpreted as a clustering task, where cluster centres are retained and all other ridges can be recovered by identifying each with the closest cluster or a linear combination of the two closest centres. Operating with a non-Euclidean metric defined by the subspace distance, clustering algorithms such as k-medoids[4] can be used. Algorithm 4.4 describes an algorithm for clustering ridge directions using k-medoids, based on its implementation in [119]. Algorithm 4.3 can be used as the corresponding recovery algorithm. Alternatively, each removed component can be replaced by its nearest medoid. In Section 4.2.3.3, the ridge compression algorithm is compared with k-medoids for compressing the flow field of a CFD simulation case study.

[4]This algorithm can be seen as a generalisation of k-means clustering by measuring the dissimilarity between objects using a non-Euclidean metric. Instead of centroids, the cluster centres are referred to as *medoids*.

Algorithm 4.4 Clustering ridge directions with k-medoids.

Input
List of ridge directions $\boldsymbol{W}_1, ..., \boldsymbol{W}_N$ corresponding to $g_1, ..., g_N$ (but the ridge profiles are not needed), and the number of components to retain k.
Output
List of subsampled ridge directions $\boldsymbol{W}_{N_1}, ..., \boldsymbol{W}_{N_k}$, and a list of nearest neighbours $L \in \mathbb{N}^{(N-k)\times 2}$ corresponding to missing components.
Initialise list of medoids $D \subset (1, ..., N)$ randomly, where the number of medoids is equal to k.
Assign each non-medoid component to its closest medoid.
$\Sigma_d =$ sum of distances of each non-medoid component to its nearest medoid.
while the value of Σ_d is different from its value in previous iteration **do**
Find a new medoid from each cluster which minimises the sum of distances to all other components in the cluster.
Assign each non-medoid component to its new closest medoid and calculate Σ_d.
end while
for each non-medoid component $\boldsymbol{W}_i$ **do**
$L[i, 1] = \text{argmin}_{j \in D \setminus i} dist(\boldsymbol{W}_i, \boldsymbol{W}_j)$
$L[i, 2] = \text{argmin}_{j \in D \setminus \{L'[i,1], i\}} dist(\boldsymbol{W}_i, \boldsymbol{W}_j)$ s.t. $dist(\boldsymbol{W}_i, \boldsymbol{W}_j) < dist(\boldsymbol{W}_j, \boldsymbol{W}_{L[i,1]})$
end for
Retain $\boldsymbol{W}_{N_1}, ..., \boldsymbol{W}_{N_k}$ where $D = (N_1, ..., N_k)$ are the medoids, and discard the rest.

4.2.2.1 A perturbation bound on the mean squared error

The ridge compression and recovery algorithms reduce the number of ridge subspaces stored in the scalar field representation. In the process, some subspaces are replaced by similar ones. Below, we establish the stability of this procedure by considering the mean squared error on each node. Given a ridge function $g(\boldsymbol{W}^\top \mathbf{x})$, consider a perturbation of the ridge subspace from $\mathcal{S} = \text{colspan}(\boldsymbol{W})$ to $\widetilde{\mathcal{S}} = \text{colspan}(\widetilde{\boldsymbol{W}})$. The goal is to bound the mean squared error incurred by the perturbation. Consider the Taylor expansion

$$g\left(\widetilde{\boldsymbol{W}}^\top \mathbf{x}\right) = g\left(\boldsymbol{W}^\top \mathbf{x}\right) + \left((\widetilde{\boldsymbol{W}} - \boldsymbol{W})^\top \mathbf{x}\right)^\top \underbrace{\nabla g(\mathbf{u})|_{\mathbf{u}=\boldsymbol{W}^\top \mathbf{x}}}_{\nabla_{\mathbf{u}} g(\boldsymbol{W}^\top \mathbf{x})} + \text{h.o.t.} \tag{4.38}$$

If the subspace perturbation is small enough, higher-order terms (h.o.t.) can be neglected and the mean squared error can be approximated as

$$\mathbb{E}\left[\left(g\left(\widetilde{\boldsymbol{W}}^\top \mathbf{x}\right) - g\left(\boldsymbol{W}^\top \mathbf{x}\right)\right)^2\right] \approx \varepsilon = \mathbb{E}\left[\left(\mathbf{x}^\top(\widetilde{\boldsymbol{W}} - \boldsymbol{W})\nabla_{\mathbf{u}} g\left(\boldsymbol{W}^\top \mathbf{x}\right)\right)^2\right]. \tag{4.39}$$

Clearly, the quantity ε depends on the specification of basis matrices, which are not fixed for given subspaces. In the following theorem, we show that it is possible to select basis matrices that allow ε to be bounded by a function of the subspace distance of the perturbation, which is independent of the basis matrices.

Theorem 4.2.2. *Let $\mathcal{S} = colspan(\boldsymbol{W})$, and $\widetilde{\mathcal{S}}$ be a perturbation of $\mathcal{S}$, with associated principal angles $\{\phi_i\}_{i=1}^n$ where $0 \leq \phi_1 \leq \phi_2 \leq ... \leq \phi_n \leq \pi/2$ (see Section 6.4.3 of [60] for definition). Assume that the square of the gradient is bounded as $\nabla_{\mathbf{u}} g^\top \nabla_{\mathbf{u}} g \leq G^2$ and $\mathbb{E}[\mathbf{x}\mathbf{x}^\top] = \sigma_x \boldsymbol{I}_d$ (i.e. inputs are independent and identically distributed). Then, we can pick $\boldsymbol{W}, \widetilde{\boldsymbol{W}} \in \mathbb{R}^{d\times n}$ where $\mathcal{S} = colspan(\boldsymbol{W})$ and $\widetilde{\mathcal{S}} = colspan(\widetilde{\boldsymbol{W}})$ such that*

$$\varepsilon \leq G^2 \sigma_x \sum_{i=1}^{n} (2 - 2\cos(\phi_i)), \tag{4.40}$$

where ε is the first-order approximation to the mean squared error (4.39)*.*

Proof. Applying the Cauchy-Schwarz inequality to $\left(\mathbf{x}^\top(\widetilde{\boldsymbol{W}} - \boldsymbol{W})\nabla_{\mathbf{u}} g(\boldsymbol{W}^\top \mathbf{x})\right)^2$ in (4.39) yields the following:

$$\begin{aligned} \varepsilon &\leq \mathbb{E}\left[\left(\nabla_{\mathbf{u}} g^\top \nabla_{\mathbf{u}} g\right)\left(\mathbf{x}^\top(\widetilde{\boldsymbol{W}} - \boldsymbol{W})(\widetilde{\boldsymbol{W}} - \boldsymbol{W})^\top \mathbf{x}\right)\right] \\ &\leq G^2 \underbrace{\mathbb{E}\left[\mathbf{x}^\top(\widetilde{\boldsymbol{W}} - \boldsymbol{W})(\widetilde{\boldsymbol{W}} - \boldsymbol{W})^\top \mathbf{x}\right]}_{:=\eta}. \end{aligned} \tag{4.41}$$

Let $\boldsymbol{E} = \widetilde{\boldsymbol{W}} - \boldsymbol{W}$, $\mathbf{y} = \boldsymbol{E}^\top \mathbf{x}$ and $\mathbf{e}_i$ be the i-th column of $\boldsymbol{E}$. Then,

$$\eta = \mathbb{E}\left[\mathbf{y}^\top \mathbf{y}\right] = \sum_{i=1}^{n} \mathbf{e}_i^\top \mathbb{E}\left[\mathbf{x}\mathbf{x}^\top\right] \mathbf{e}_i = \sigma_x \sum_{i=1}^{n} \mathbf{e}_i^\top \mathbf{e}_i. \tag{4.42}$$

However, we have $\mathbf{e}_i = \widetilde{\mathbf{w}}_i - \mathbf{w}_i$. So,

$$\begin{aligned} \mathbf{e}_i^\top \mathbf{e}_i &= (\widetilde{\mathbf{w}}_i - \mathbf{w}_i)^\top (\widetilde{\mathbf{w}}_i - \mathbf{w}_i) \\ &= 2 - 2\widetilde{\mathbf{w}}_i^\top \mathbf{w}_i, \end{aligned} \tag{4.43}$$

where we have used the fact that $\widetilde{\mathbf{w}}_i^\top \widetilde{\mathbf{w}}_i = \mathbf{w}_i^\top \mathbf{w}_i = 1$. So, substituting (4.42) and (4.43) into (4.41), we have:

$$\varepsilon \leq G^2 \sigma_x \sum_{i=1}^{n} \left(2 - 2\widetilde{\mathbf{w}}_i^\top \mathbf{w}_i\right). \tag{4.44}$$

Choosing $W, \widetilde{W}$ to contain the n principal vectors of the pair $(\mathcal{S}, \widetilde{\mathcal{S}})$, we have

$$\widetilde{\mathbf{w}}_i^\top \mathbf{w}_i = \cos(\phi_i) \tag{4.45}$$

for every $i = 1, ..., n$, which completes the proof. □

Theorem 4.2.2 establishes a stability bound on the approximation error of a ridge function with a small perturbation of the associated subspace. Given the Lipschitz continuity of the underlying field with respect to the spatial domain, it is reasonable to assume that neighbouring nodes are determined by ridge subspaces that are closely related to each other. This implies that the error incurred by compression is bounded.

In fact, note that the actual approximation error incurred via compression can be smaller than suggested by Theorem 4.2.2, since the change in the ridge profile g as a result of the perturbation in the subspace is not accounted for. In practice, after approximating the subspace by a perturbed version of its original value, the ridge profile can be refitted to data projected to the new subspace, minimising the mean squared error in the process. The new error can be smaller than simply applying g to the data projected to the new subspace without changing the compression level.

4.2.3 Numerical examples

In this section, we illustrate the embedded ridge approximation approach with two numerical examples.

4.2.3.1 Analytical function

Consider the function

$$h(\mathbf{x}) = \begin{bmatrix} 2 & 3 & 5 \end{bmatrix} \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ f_3(\mathbf{x}) \end{bmatrix} \tag{4.46}$$

where

$$f_1(\mathbf{x}) = \left(\mathbf{w}_1^\top \mathbf{x}\right)^2 + \left(\mathbf{w}_1^\top \mathbf{x}\right)^3,$$
$$f_2(\mathbf{x}) = \exp\left(\mathbf{w}_2^\top \mathbf{x}\right),$$
$$f_3(\mathbf{x}) = \sin\left(\left(\mathbf{w}_3^\top \mathbf{x}\right)\pi\right),$$

defined over the domain $\mathcal{D} = [-1, 1]^{10}$ where the inputs $\mathbf{x}$ are independent and have uniform marginals. Note that $h(\mathbf{x})$ is an exact ridge function with three ridge directions spanned by the columns of $\boldsymbol{W} = [\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3]$. We draw $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3$ as random vectors with unit Euclidean norm and compare the recovered ridge directions to the drawn vectors using the subspace distance (see (4.37)). Polynomial variable projection is used for finding ridge directions, where the polynomials have a maximum total degree of 7. For the optimisation loop over the Grassmann manifold, the convergence criterion is set to be when the subspace distance between the ridge directions of the previous and current iterations is smaller than 10^{-7}.

For embedded ridge approximation, polynomial variable projection is used to estimate the ridge directions for each component function $f_1(\mathbf{x}), f_2(\mathbf{x})$ and $f_3(\mathbf{x})$, which are then applied to calculate the first three leading eigenvectors of the vector gradient covariance matrix of $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x})]^\top$. The weights are set as $\boldsymbol{\theta} = [2, 3, 5]^\top$ to find an estimate of the dimension-reducing subspace of $h(\mathbf{x})$. For direct ridge approximation, we use polynomial variable projection to estimate the three-dimensional dimension-reducing subspace of $h(\mathbf{x})$ directly. The number of observations used for each method is varied and the subspace distance between the recovered directions and the true directions is calculated. It is observed that the results are binary; we either get a small subspace distance from successful recovery or a large subspace error from failure in recovery. Thus, the probability of successful recovery—where the subspace distance is below 0.005—across 40 trials is plotted on the left of Figure 4.2. It was found via perturbation studies that 40 trials yield representative results. This plot shows that recovery using embedded ridge approximations is more stable and requires fewer observations than direct ridge approximation for a given recovery probability.

To achieve successful recovery of $\boldsymbol{W}$ from the embedded ridge approximation, one needs to be able to successfully recover the ridge directions in each individual function, as reflected from the right plot of Figure 4.2. Despite the need to successfully find three sets of ridge directions concurrently, the probability of recovery is still significantly higher for the embedded ridge approximation method. This is because the optimisation over three-dimensional subspaces required in the direct method

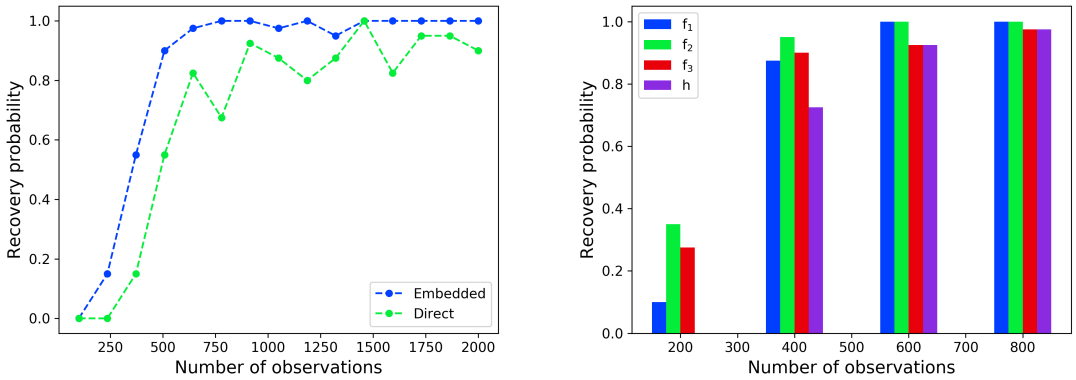


Figure 4.2 (Left) Comparing the average recovery probability for embedded and direct ridge approximation for $h(\mathbf{x})$ in (4.46). (Right) Recovery probability of component ridges and QoI ridge when using an embedded ridge approximation. A successful recovery is defined to be when the subspace error is smaller than 0.005. Forty trials are performed.

is much more challenging than their one-dimensional counterparts required in the embedded method (see Table 3 in [77]).

4.2.3.2 Embedded ridges of a subsonic airfoil

Embedded ridge approximation is applied to analyse the flow around the NACA0012 airfoil profile with shape deformations. The deformation is parameterised using $d = 50$ Hicks-Henne bump functions around the airfoil, and the variation in the static pressure around the airfoil is computed. An entry Mach number of 0.3 (subsonic) and an angle of attack of 1.25° are fixed, with free-stream temperature and pressure at 273.15 K and 101325 Pa, respectively. The pressure profile is found using the compressible Euler flow solver in the open-source CFD suite SU2 [48]. The coefficients of lift and drag[5] are known to be linear functions of the pressure around the airfoil [2, Ch. 1], given by

[5]Ignoring skin friction and assuming a unit reference area.

$$
\begin{aligned}
C_l &= \frac{1}{\frac{1}{2}\rho_0 v_\infty^2} \oint p(\mathbf{x})\mathbf{n}\cdot\mathbf{k}\,\mathrm{d}s & C_d &= \frac{1}{\frac{1}{2}\rho_0 v_\infty^2} \oint p(\mathbf{x})\mathbf{n}\cdot\mathbf{j}\,\mathrm{d}s \\
&\approx \frac{1}{\frac{1}{2}\rho_0 v_\infty^2} \sum_{i=1}^{N_s} p_{s,i}(\mathbf{x})\mathbf{n}_i\cdot\mathbf{k}\,\Delta s_i & &\approx \frac{1}{\frac{1}{2}\rho_0 v_\infty^2} \sum_{i=1}^{N_s} p_{s,i}(\mathbf{x})\mathbf{n}_i\cdot\mathbf{j}\,\Delta s_i \\
&= \boldsymbol{\theta}_l^\top \mathbf{p}_s(\mathbf{x}), & &= \boldsymbol{\theta}_d^\top \mathbf{p}_s(\mathbf{x}),
\end{aligned}
$$

where $\oint$ denotes an integral around the airfoil surface. The input variable $\mathbf{x} \in \mathbb{R}^d$ contains the Hicks-Henne bump amplitudes; $\mathbf{n}$ is the surface normal, $\mathbf{k}$ the direction perpendicular to the flow, and $\mathbf{j}$ the direction parallel to the flow. In the normalising factors, ρ_0 is the free-stream density, and v_∞ is the free-stream speed. The computational domain consists of $N = 5233$ mesh elements, of which $N_s = 200$ reside on the surface and are accounted for in lift and drag calculations. The surface pressure profile can accordingly be represented by the vector-valued function $\mathbf{p}_s : \mathbb{R}^d \to \mathbb{R}^{N_s}$. Note that the approximation in the second line of both expressions comes not solely from the discretisation but also from the assumption that $\mathbf{n}$ is independent of $\mathbf{x}$—a good approximation when the geometric perturbations are small. Under this approximation, the coefficients of lift and drag can then be expressed as linear functions of the components of $\mathbf{p}_s(\mathbf{x})$.

As the flow is entirely subsonic and inviscid, the bumps are expected to have a strongly local influence. Hence, the pressure profile $\mathbf{p}_s(\mathbf{x})$ is well-approximated by embedded ridge functions. The following procedure applies the steps in Algorithm 4.1 to form ridge approximations of C_l and C_d, assuming the pressure at each node is approximated by a one-dimensional ridge function.

1. Using a gradient-free computational strategy, find the leading ridge direction $\widehat{\boldsymbol{W}}_i$ for each $p_{s,i}(\mathbf{x})$.

2. Fit a low-dimensional surrogate using this leading mode for each $p_{s,i}$,

$$
p_{s,i}(\mathbf{x}) \approx \widehat{g}_i(\widehat{\boldsymbol{W}}_i^\top \mathbf{x}), \tag{4.47}
$$

   for the i-th component of $\mathbf{p}_s$. Univariate orthogonal polynomials are used for the profiles $\widehat{g}_i(\cdot)$.

3. The elements of the Jacobian can be computed via these ridge approximations:

$$
\widehat{\boldsymbol{J}}(\mathbf{x})_{ij} = \widehat{W}_{ji}\widehat{g}_j'(\widehat{\boldsymbol{W}}_j^\top \mathbf{x}), \tag{4.48}
$$

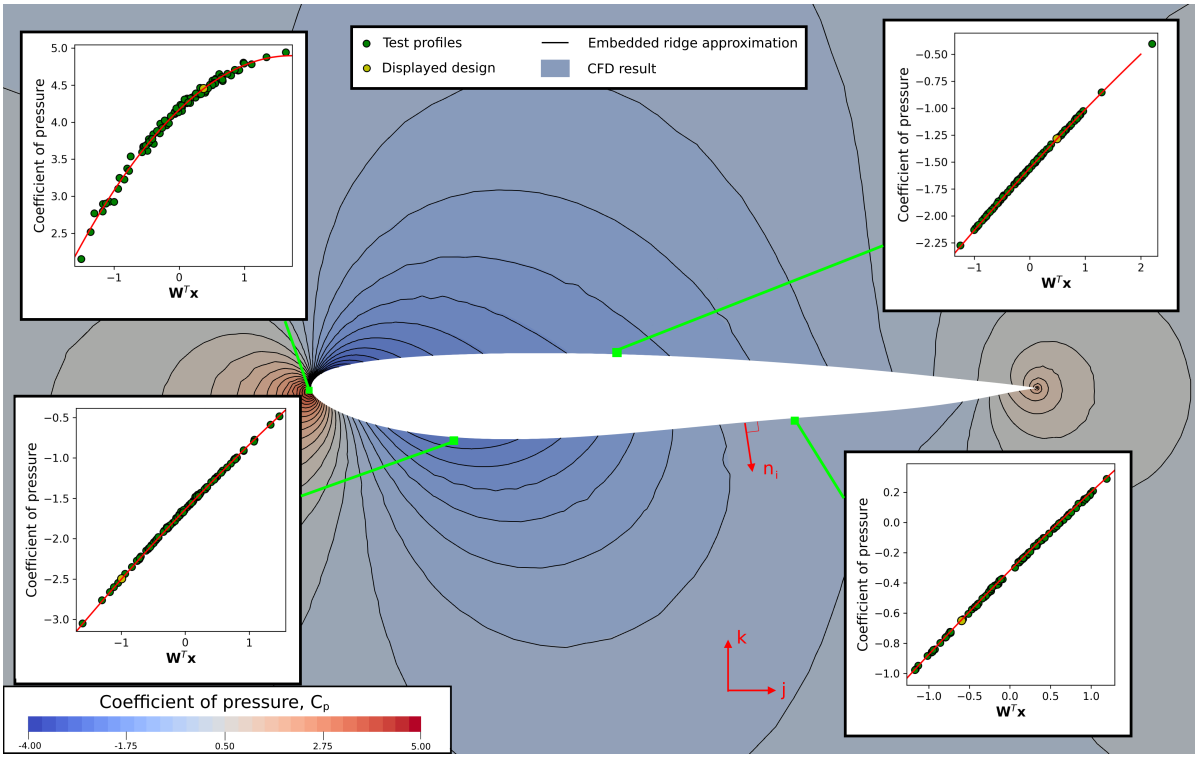


Figure 4.3 Static pressure field around a deformed airfoil not used in training. In the background, coloured contours represent the CFD calculation of the field, which is compared against the black isolines representing an embedded ridge approximation fitted with 400 flow field observations. The embedded ridge approximation has also been compressed by removing 3000 nodes out of 5233 using the ridge compression algorithm. Moreover, inset plots show the sufficient summary plots of nodal pressure at various locations around the airfoil, where unseen test data is projected to the fitted ridge subspace along with the best fit polynomial.

where $\widehat{W}_{ji}$ is the i-th element of $\widehat{\boldsymbol{W}}_j$. Gradients here are furnished by the polynomial approximation analytically.

4. Compute the gradient covariance matrix with (4.29) by substituting $\boldsymbol{\theta}_l$ and $\boldsymbol{\theta}_d$, from which the dimension-reducing subspaces can be computed for the integral QoIs C_l and C_d.

From the ridge subspaces at each node, sufficient summary plots can be used to visualise the pressure variation with respect to the active variable $\boldsymbol{W}_i^\top \mathbf{x}$, as in Figure 4.3.

To apply the embedded ridge approximation approach, 1-D ridge functions with quadratic ridge profiles are fitted for each component of the surface pressure profile $p_{s,i}$. Three gradient-free dimension-reducing strategies to find the ridge subspace at each node are compared:

1. Fitting global linear models for each node, and taking the ridge direction as the normalised parameters of the linear model (see Section 2.2.4.1). Note that the *ridge profiles* are still quadratic; the linear models are only used to find the ridge directions. This will be referred to as "Embedded linear".

2. As above, but using quadratic polynomial variable projection for nodes close to the leading edge, noting that pressure variation near the leading edge tends to be non-linear. The ridge subspace remains one-dimensional for all nodes. This will be referred to as "Embedded VP".

3. As above, but using MAVE only for nodes close to the leading edge to extract a one-dimensional ridge subspace. Then, a quadratic polynomial is fitted in this one-dimensional subspace for these nodes. This will be referred to as "Embedded MAVE".

Embedded ridge approximation is compared with the direct ridge approximation approach, where observations for C_l and C_d are used to find a ridge approximation directly and without the use of gradients. For the direct approach, three dimension-reducing strategies are compared—polynomial variable projection, MAVE and the linear model. For both the embedded and direct approaches, one dimension is used for the ridge approximation of C_l, and two dimensions for C_d. Note that the linear model used in the direct approach is unable to estimate more than one dimension, so only one is used for C_d in this case.

In Figure 4.4, the mean squared errors of the surrogate models fitted using the dimension-reducing subspaces resulting from embedded and direct ridge approximation are plotted. The mean squared error of approximating the QoI $h(\mathbf{x})$ with $\widehat{h}(\mathbf{x})$ is

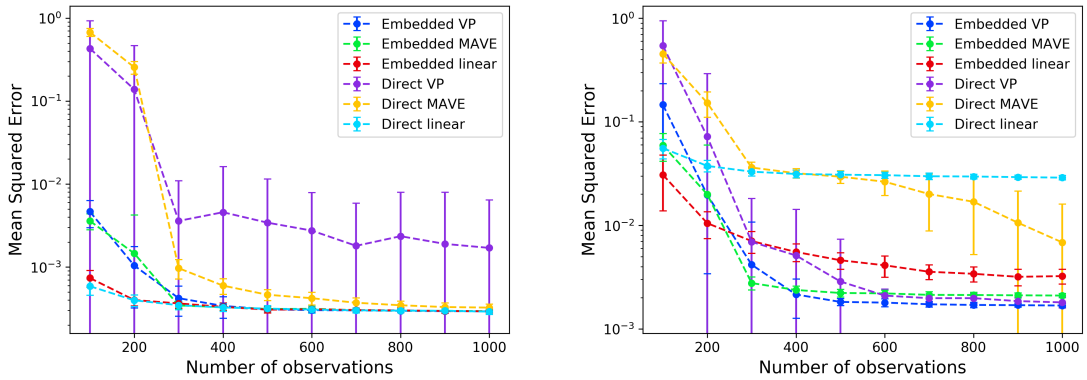


Figure 4.4 Mean squared error of C_l (left) and C_d (right) for surrogate models with VP, MAVE and linear models via direct and embedded ridge approximations.

evaluated as

$$\varepsilon_h = \frac{1}{M'} \sum_{j=1}^{M'} \frac{\left(h(\mathbf{y}^{(j)}) - \widehat{h}(\mathbf{y}^{(j)})\right)^2}{\sigma_h}, \tag{4.49}$$

where $\mathbf{y}^{(j)} \in \mathbb{R}^d$ are verification samples drawn independently from data used to train the response surfaces. The ridge approximation of $h(\mathbf{x})$ evaluated at $\mathbf{y}^{(j)}$ is denoted $\widehat{h}(\mathbf{y}^{(j)}) = \widehat{g}_i\left(\widehat{\boldsymbol{W}}^\top \mathbf{y}^{(j)}\right)$, and σ_h is the sample variance of $h(\mathbf{y})$ across all M' verification samples.

It is shown that using embedded ridge approximation reduces the mean squared error compared to direct estimation when the number of samples is limited. The errors for embedded VP and MAVE approximately reach convergence in 300 observations for C_l, and 400 observations for C_d. Although the linear models (in both the direct and embedded cases) suffice for estimating C_l, for functions with stronger non-linear dependencies such as C_d, the linear model is shown to have a larger error compared to VP and MAVE. Also note that the use of embedded ridge approximation permits us to extend the capability of linear models to estimate more than one mode in the scalar QoIs, improving its performance as seen on the right of Figure 4.4.

4.2.3.3 Sparse storage of pressure field

Because of its smoothness, the pressure field can be effectively compressed by the ridge compression algorithm. It is observed that each node out of the $N = 5233$ nodes in the flow field can be well-approximated by a 1-D ridge function. The efficacy of

the ridge compression and recovery algorithms is tested by selecting a subset of the components to store and recovering the missing components from it.

A range of components is removed using the ridge compression algorithm (Algorithm 4.2) applied recursively, where in each stage at most $S = 500$ components are removed. Then, we reconstruct the missing ridge directions using the ridge recovery algorithm (Algorithm 4.3), again applied recursively as in Figure 4.1. The reconstruction quality is evaluated using the average mean squared error ε_R, defined as

$$\varepsilon_R = \frac{1}{N'M'} \sum_{i=1}^{N'} \sum_{j=1}^{M'} \frac{\left(p_i \left(\mathbf{y}^{(j)} \right) - \widehat{p}_i \left(\mathbf{y}^{(j)} \right) \right)^2}{\sigma_{pi}}, \tag{4.50}$$

where N' is the number of recovered components, and other variables are defined similarly as before. For comparison, the k-medoids clustering algorithm (Algorithm 4.4) is run under the same settings, with the same recovery algorithm (Algorithm 4.3). In addition, a random deletion strategy is run, where the removed nodes are selected randomly, and missing modes are recovered by substituting the nearest neighbour in terms of subspace distance.

In Figure 4.5, the mean squared error averaged across all recovered components is plotted for the three methods: the ridge compression algorithm, k-medoids clustering and random deletion. The plot shows that applying compression recursively allows recovery of missing ridge subspaces with greater accuracy than the other methods up to approximately 4700 components, which covers almost all of the nodes.

In Figure 4.6 and Figure 4.3, the pressure profiles on the surface of the airfoil and the entire flow field are compared for two cases: full CFD results and the reconstruction after removal of 3000 nodes using the compression algorithm respectively. The plots are for an airfoil geometry which was not used in the computation of the embedded ridge approximation. It can be seen that the pressure field is well approximated, including the region near the leading edge where larger non-linear pressure variations are present. Figure 4.7 shows the locations of the removed nodes at different levels of compression. Nodes at the far-field are prioritised for removal, and, at high compression levels, nodes near the leading and trailing edges tend to be retained.

4.3 Conclusions for chapter

In this chapter, dimension-reducing strategies for multi-objective functions are studied focusing on two goals: finding invariant subspaces for multiple quantities, and the

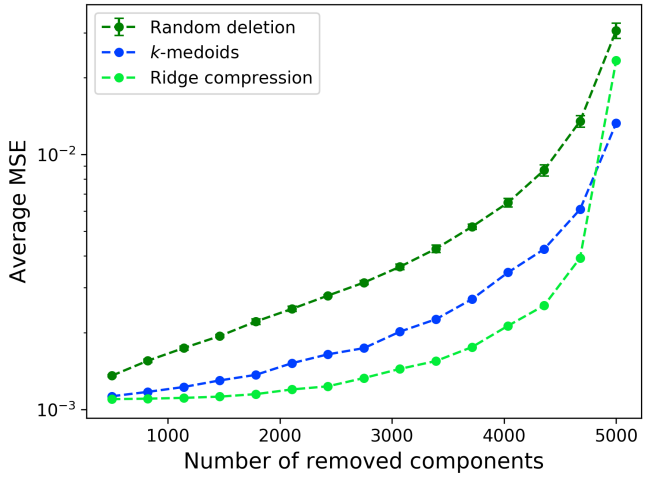


Figure 4.5 Average mean squared error after removing various numbers of field components using the ridge compression algorithm (Algorithm 4.2) applied recursively with a stride $S = 500$, k-medoids clustering (Algorithm 4.4) and random deletion.

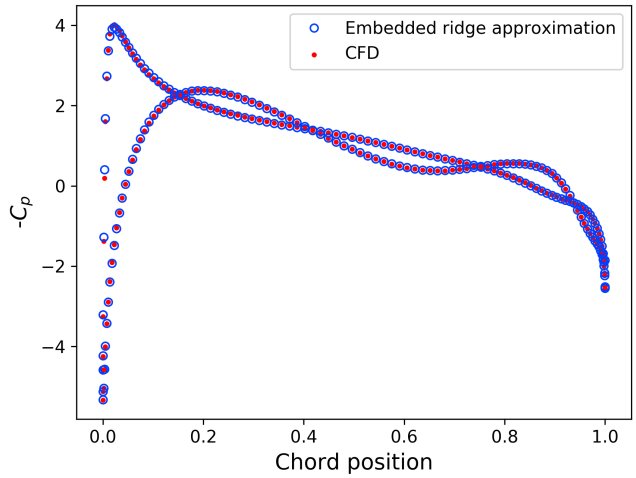


Figure 4.6 Comparing the C_p profile on the surface of the airfoil before and after removing 3000 nodes with ridge compression using an embedded ridge approximation formed from 400 observations.

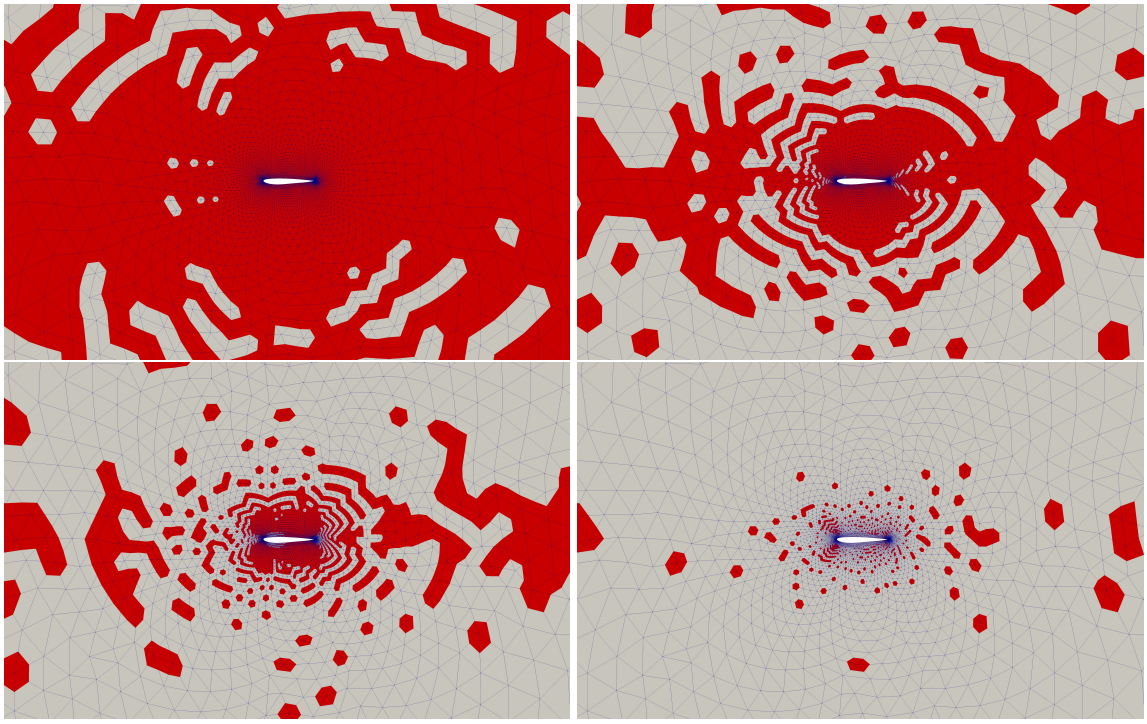


Figure 4.7 Ridge compression scheme when 500 (top left), 1500 (top right), 2500 (bottom left) and 4300 (bottom right) nodes are removed. Red nodes are retained nodes, while grey ones are removed.

approximation of scalar fields with embedded ridges. For the former, the intersection approach is compared with an approach based on vector-valued dimension reduction. For the latter, the method of embedded ridge approximation is proposed. Scalar fields and associated integral QoIs can be efficiently approximated assuming localised scale of influence. The ridge compression and recovery algorithms are proposed to mitigate possible storage issues. Embedded ridge approximation is demonstrated on numerical examples, showing good accuracy at reconstructing a flow field around an airfoil. In the next chapter, multi-objective invariance strategies will be applied in a novel computational framework for automatic tolerance design.

Chapter 5

Tolerance design with blade envelopes

Manufacturing variations and in-service degradation of the blade geometries of rotating components have a sizeable impact on the aerodynamic performance of a jet engine (see Figure 5.1). To a design engineer, it is therefore important to be able to understand and mitigate the associated losses and risks. However, it is challenging to gauge the impact of manufacturing variabilities prior to manufacture, let alone trying to predict what the amplified impact of any in-service degradation might be. Currently, a two-pronged approach is usually taken. First, components are being designed to operate over a range of conditions (and uncertainties therein) via robust optimisation techniques [87, 136] as well as more traditional design guides such as loss buckets—i.e., loss across a range of positive and negative incidence angles [64]. In parallel, there has been a growing research effort to assess 3-D manufacturing variations and in-service degradation by optically scanning (via GOM) the manufactured blades, meshing them, and running them through a flow solver [92]. Both approaches, while useful in extracting aerodynamic inference, are limiting. One of the key bottlenecks is the cost of evaluating flow quantities of interest via CFD, as the dimensionality of the space of manufactured geometries is too large to fully explore, even with an appropriately tailored design of experiment (DoE). Dimension reduction methods such as PCA have been used to extract features that constitute the largest geometric variations [54, 92]. However, these features are not performance-based, i.e., the mode of greatest geometric variability need not correspond to the mode of greatest performance scatter, a point raised by Dow and Wang [46].

In this chapter, we propose a novel computational framework known as a *blade envelope*. This framework offers a performance-based analysis of geometric variations which leads to principled manufacturing guidelines. Upon scanning a manufactured geometry, one is able to judge whether it should be discarded or kept and installed

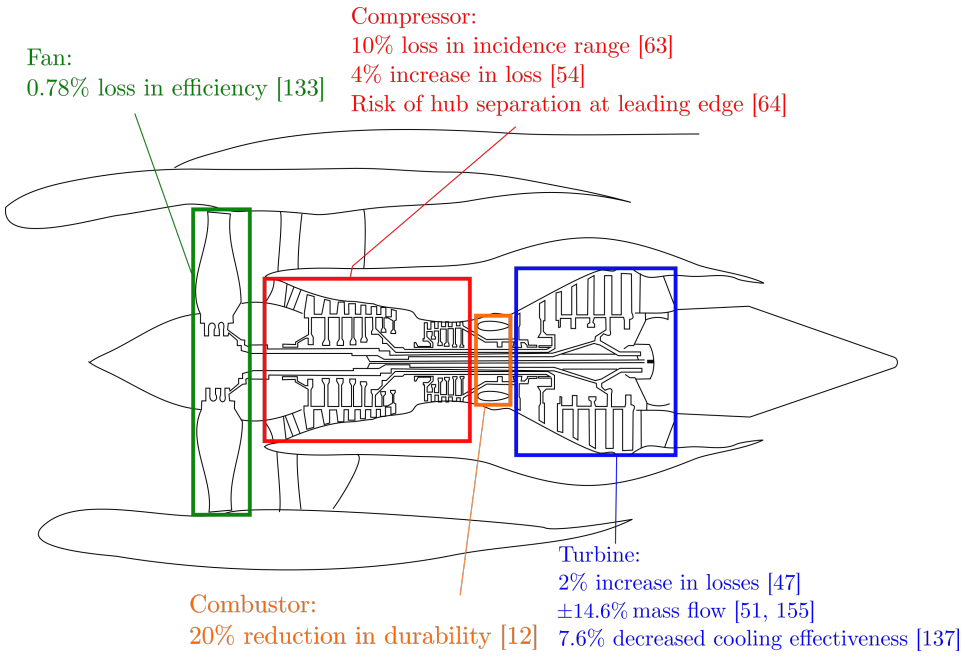


Figure 5.1 Impacts associated with manufacturing variations in a jet engine; see [135, 63, 54, 64, 12, 47, 51, 157, 139].

in an engine. This is done through a scrap-or-use confidence value, supported by guarantees on in-service performance which are furnished through model-based dimension reduction methods.

5.1 Overview and related work

A blade envelope offers a quantitative manufacturing guide for a blade that can be used to instruct manufacturers whether a scanned component should be used or scrapped. It is defined as a statistical distribution of perturbed geometries around a mean design geometry $\boldsymbol{\mu} \in \mathbb{R}^N$,

$$BE \sim \mathrm{p}(\boldsymbol{\mu}, \boldsymbol{S}). \tag{5.1}$$

This distribution contains geometries that result in close-to-nominal performance. Provided that a geometry adheres to the *tolerance covariance* $\boldsymbol{S} \in \mathbb{R}^{N\times N}$ of the distribution, it is considered to reside *within* the blade envelope, and is guaranteed to offer near-identical performance to nominal.

Rigorous definitions of these concepts are offered in subsequent sections. Here, we first provide an intuitive overview using a visual example summarised in Figure 5.2.

The *control zone*, showing the pointwise two-standard-deviation intervals of the blade envelope distribution (5.1) at each location on the blade, is shown in grey. However, this alone is not sufficient to capture the key characteristics of the tolerance covariance criterion; specifically, it is unable to capture the correlation between the pointwise displacements. Instead, they can be visualised by plotting the displacements of performance-invariant profiles within the blade envelope. Referring to Figure 5.2, it can be seen that leading edge (LE) displacements on both the suction and pressure sides obey a pattern: invariant profiles which are positively displaced near the LE tend also to be displaced positively in the immediate vicinity on the same side; in other words, these profiles adhere to a certain *curvature constraint* dictating that the displacement cannot vary too rapidly. Consider two scanned profiles shown as red and blue markers in Figure 5.2. Each marker corresponds to airfoil profile measurements, emulating those taken by a coordinate measurement machine (CMM). The measurements are interpolated using B-splines to obtain the airfoil profiles, both of which lie within the control zone. However, as can be expected, it is the profile variation within the control zone that truly dictates how adverse the change in performance will be. It is clear that the red profile, containing drastic variation in displacements over a short range, does not obey the geometric correlation shown by other profiles in the envelope, and therefore should be scrapped; indeed, this corresponds to the aerodynamic intuition that sharp geometric discontinuities are likely to have a severe impact on performance. Meanwhile, the blue geometry, having a milder curvature profile, adheres to the covariance criterion and can likely be used in an engine. This example serves to highlight the important point that pointwise tolerance ranges described by the control zone alone are not sufficient. Although we arrived at this conclusion by inspection based on rough heuristics, it should be stressed that in practice the suitability of a profile should be determined computationally using the mathematical definition of the tolerance covariance to be presented below.

Some works in the literature of tolerance design recognise that pointwise tolerance bounds are insufficient to identify unacceptable geometries. Consider the work of Lobato et al. [103] who offer a recipe for acceptance and rejection of airfoils based on the curvature variation within a chord-wise interval. Provided the curvature of the measured profile lies within a computed upper and lower tolerance band, neither of which are necessarily inferred from CFD, the profile is deemed acceptable. With blade envelopes, profile tolerance constraints on displacement and curvature are supported by computational and/or experimental evidence and pivoted on geometric correlations. This helps provide clarity as to where tolerances need to be tightened and

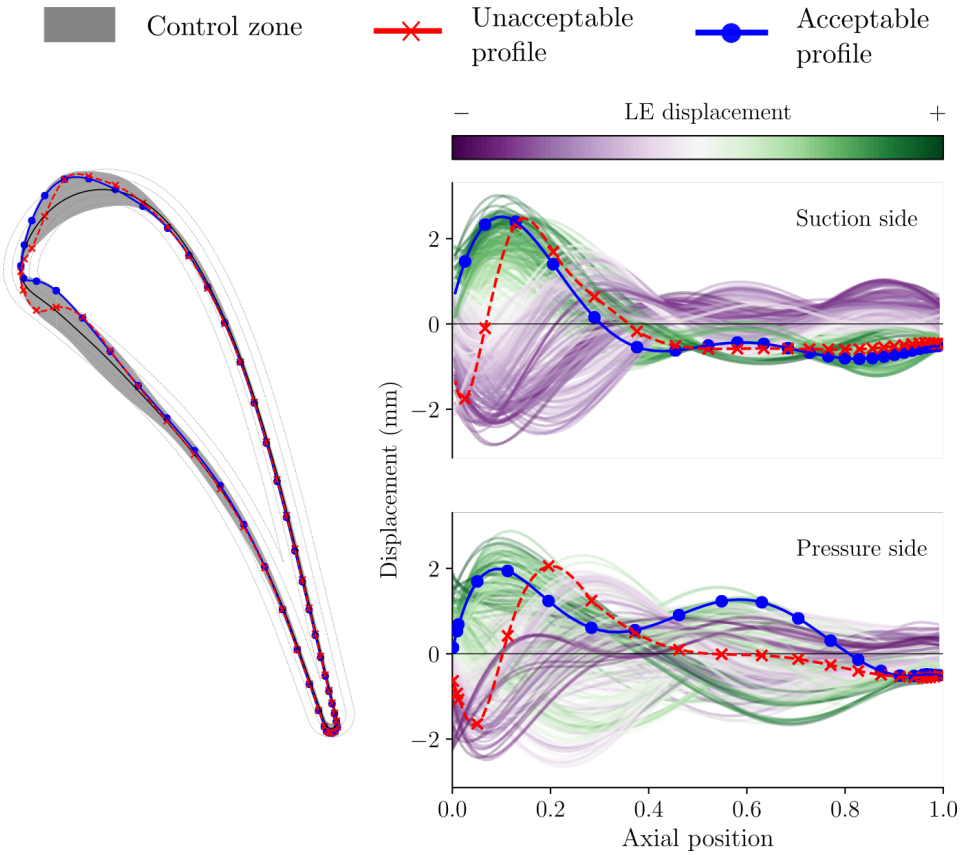


Figure 5.2 Visual representation of a blade envelope. The control zone (pointwise two-standard-deviation intervals) is shown in grey. The tolerance covariance is characterised by the airfoil displacement plots colour-coded according to average displacement in the first 5% of span. Dots denote airfoil coordinate measurements. Note: in all displacement plots in this chapter, the axial distance is normalised by the axial chord; in all figures in this chapter, displacements are drawn to scale.

where they can be relaxed, thus avoiding unnecessary scrapping due to potentially overly conservative bounds.

5.2 Computational test case

To demonstrate the implementation of blade envelopes, the von Karman Institute LS89 linear turbine cascade is selected for the computational examples. This transonic, highly-loaded blade serves as a rich experimental (and subsequently computational) repository with Schlieren flow visualisations, blade static pressure measurements, exit flow angle measurements, and even blade convective heat transfer values. An experimental campaign was carried out by Arts et al. [4] at a range of different

Figure 5.3 The FFD box (black dots) with some possible deformations within the design space (grey curves) and the baseline profile (black curve).

exit Mach numbers ranging from 0.70 to 1.10, exit Reynolds numbers ranging from 0.5×10^5 to 2.0×10^6 and freestream turbulence intensities of 1% to 6%. The overall chord of the tested profile was approximately 67 mm with a stagger angle of 55°.

To simulate deformations arising from manufacturing variations, a large design space around the baseline profile is defined. This space $\mathcal{D}$ is parameterised by $d = 20$ independent free-form deformation (FFD) design variables, scaled to lie within the range -1 to 1, such that each design vector $\mathbf{x} \in [-1, 1]^d$. The FFD nodes are constrained to move only in the direction perpendicular to the inflow; some possible deformations are captured in Figure 5.3. Note that the magnitude of geometric variations imposed here grossly exaggerates the typical scales of deviations seen in real turbine blades.

The CFD code SU2 is used for converting design vectors to geometry coordinates, deforming the mesh corresponding to various geometries and their subsequent numerical solves. The latter are obtained by solving the Reynolds-averaged Navier-Stokes (RANS) equations with a Spalart-Allmaras turbuence closure. No transition model is used and all the boundary layers are assumed to be turbulent. A stagnation pressure driven inlet and an exit static pressure outlet are used to set the passage pressure ratio. Periodic boundary conditions are imposed along the pitch-wise direction, and the freestream turbulent viscosity ratio was set to 100. The exact pressures, temperatures

Table 5.1 Flow properties used in the computational test case for this chapter.

Flow property	Symbol	Value
Inlet stagnation pressure	p_{01}	1.1×10^6 Pa
Inlet stagnation temperature	T_{01}	592.295 K
Inlet density	ρ	1.2866 kg m^{-3}
Freestream Reynolds number	Re	6.0×10^5
Exit static pressure	p_2	5.23×10^5 Pa
Heat capacity ratio	γ	1.4

and densities adopted for our studies are shown in Table 5.1, and fall within the regime of the conditions tested during the experimental campaign. A CFD-experimental validation study is omitted here since prior comparisons of this blade using SU2 are available in the literature [155].

5.3 Statistical methodology

In this section, we detail the statistical tools underlying the generation and deployment of the blade envelope.

5.3.1 Finding the invariant subspace

The concepts introduced in Section 4.1 form the foundation of the computational algorithm for blade envelopes. From the high-dimensional design space that contains all possible deformations from a base geometry, the first step is to gather an *input-output aerodynamic database* containing a set of points in the design space that forms a DoE and the quantities of interest evaluated at these points. From this, dimension reduction methods (those detailed in Section 2.2 coupled with strategies from Section 4.1 for multiple objectives) can be applied to find invariant subspaces. This yields a region that contains designs obeying output invariance constraints.

To complete the picture, geometric characteristics of the invariant designs need to be captured to form the blade envelope without dependence on the design space parameters. Moreover, this will need to be converted into a scrap-or-use criterion that intuitively quantifies whether a given test geometry is far from the blade envelope distribution or not.

5.3.2 Generating invariant geometries

The blade envelope is a statistical distribution of invariant geometries. To estimate parameters of this distribution, sample moments of invariant geometries can be calculated. Starting from basis matrices for the dimension-reducing subspace $\boldsymbol{W}$ and the invariant subspace $\boldsymbol{V}$, it was shown in (4.2) that the design vector $\mathbf{x}$ can be decomposed into coordinates along the respective subspaces,

$$\mathbf{x} = \boldsymbol{W}\boldsymbol{W}^\top\mathbf{x} + \boldsymbol{V}\boldsymbol{V}^\top\mathbf{x} = \boldsymbol{W}\mathbf{u} + \boldsymbol{V}\mathbf{z}, \tag{5.2}$$

where $\mathbf{u}$ and $\mathbf{z}$ are the active and inactive coordinates respectively. The goal is to generate sample points that are constrained in $\mathbf{u}$ but free to vary in $\mathbf{z}$. Additionally, these points should reside in the design space $\mathcal{D} = [-1, 1]^d$. For a fixed $\mathbf{u}$, this implies that

$$\begin{aligned} \boldsymbol{V}\mathbf{z} &\leq \mathbf{1} - \boldsymbol{W}\mathbf{u}, \\ -\boldsymbol{V}\mathbf{z} &\leq \mathbf{1} + \boldsymbol{W}\mathbf{u}, \end{aligned} \tag{5.3}$$

where the inequalities are taken componentwise. These linear constraints form a *polytope*, the high-dimensional generalisation of a polygon, in the invariant subspace. To generate sample points from this polytope, a strategy based on the hit-and-run method [142] is used. In this method, one first identifies a feasible point by locating the Chebyshev centre of the polytope[1] at the specified active coordinate $\mathbf{u}$, by solving a linear program. Then, starting from the Chebyshev centre, a random direction is selected and, in this direction, the longest line segment that lies within the polytope (and goes through the centre) is identified. Following this, a point on this line segment is selected at random yielding the first sample point. This procedure is repeated by starting with a new random direction and identifying the longest line segment along that line. These steps are captured in Figure 5.4. The obtained sample points converge in distribution to a uniform distribution over the polytope [141].

[1] The Chebyshev centre is the centre of the largest hypersphere that can fit within the polytope (see page 416 in [11]).

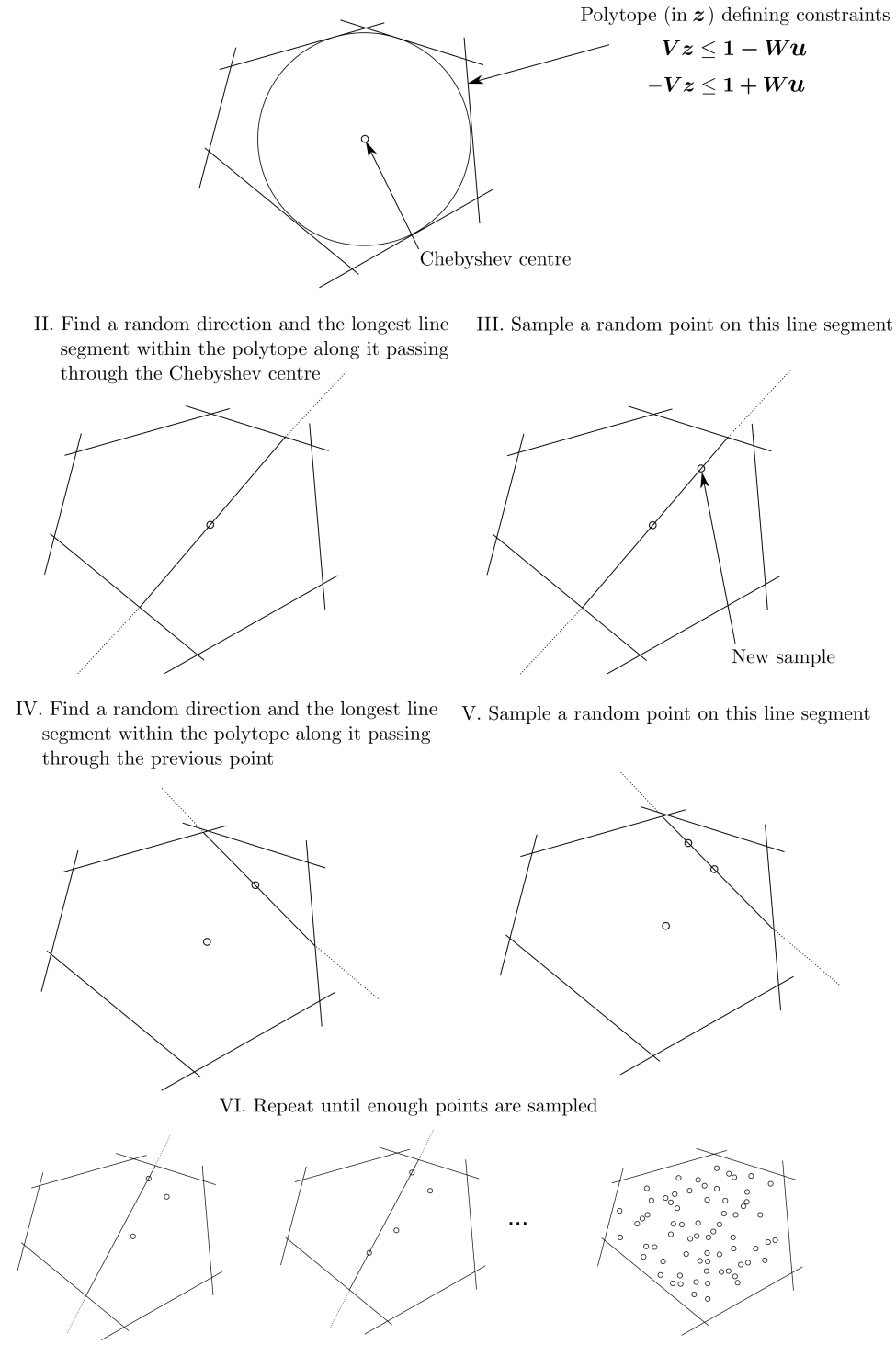


Figure 5.4 Schematic showing the hit-and-run sampling method for generating sample points in the invariant subspace.

5.3.3 Inferring the blade envelope distribution

The generated sample points in the parametric design space can be converted to corresponding deformed geometries, used to formulate the mean profile and tolerance covariance of the blade envelope. Let H be the number of invariant sample points generated via the hit-and-run method for the nominal value of $\mathbf{u}$ at the origin—corresponding to the undeformed geometry. Assume that blade geometries can be expressed with the coordinates $\mathbf{s} \in \mathbb{R}^N$ at N discrete locations on the surface (see Figure 5.5). This is compatible with the FFD design space of the axial turbine test case described in Section 5.2. Implementation details in the following will differ for other base geometries, but the computational framework remains applicable.

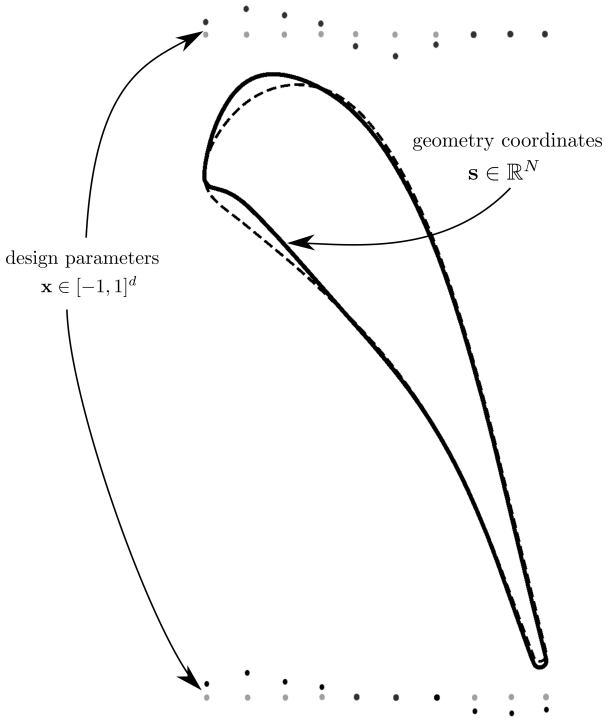


Figure 5.5 Distinguishing between design parameters $\mathbf{x}$ and geometry coordinates $\mathbf{s}$.

Equipped with a set of airfoil coordinates $\left\{\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, \ldots, \mathbf{s}^{(H)}\right\}$ corresponding to invariant designs, one can calculate the parameters of the blade envelope distribution (5.1) using sample moments. The *tolerance covariance* matrix quantitatively captures the geometric characteristics of invariant profiles. As demonstrated in Figure 5.2, it can be notionally interpreted as a constraint on the curvature variation on both the suction and pressure sides, though in practice one should adopt a computational characterisation in lieu of deriving qualitative heuristics based on inspection. Formally,

the tolerance covariance is defined as the matrix $\boldsymbol{S} \in \mathbb{R}^{N\times N}$ where each element is given by

$$\boldsymbol{S}_{ij} = \operatorname{cov}\left(s_i,\, s_j\right), \tag{5.4}$$

where the subscript i denotes the i-th discrete measurement coordinate. It can be approximated by the sample covariance using the airfoil coordinates

$$\boldsymbol{S} = \frac{1}{H}\sum_{i=1}^{H}(\mathbf{s}^{(i)} - \boldsymbol{\mu})(\mathbf{s}^{(i)} - \boldsymbol{\mu})^{\top}, \tag{5.5}$$

where the sample mean is given by

$$\boldsymbol{\mu} = \frac{1}{H}\sum_{i=1}^{H}\mathbf{s}^{(i)}. \tag{5.6}$$

The diagonal entries of this matrix refer to pointwise variance, related to the average variation of profiles at each measurement location, while the off-diagonal entries encode the pairwise covariance between the deviations at different locations.

When a manufactured component is scanned, we wish to determine whether it complies with a computed blade envelope for the design in question. This necessitates a method where one can compare the geometric coordinates of a blade with its blade envelope and distill a binary scrap-or-use output. It is assumed that, although the number of coordinate points may be smaller than N, one can linearly interpolate them to estimate the vertical displacements (or surface normal displacements) at the same horizontal coordinates (or chord-wise locations) as the blade envelope. Alternatively, one can do the opposite and downsample the coordinates associated with the blade profile, which are obtained solely from CFD.

A statistical metric known as the *Mahalanobis distance* is computed between the scanned blade and the distribution associated with the blade envelope. This metric measures the distance between a point $\widetilde{\mathbf{s}}$ and a distribution (see Figure 5.6),

$$\zeta\left(\widetilde{\mathbf{s}}\right) = \sqrt{\left(\widetilde{\mathbf{s}} - \boldsymbol{\mu}\right)^{\top} \boldsymbol{S}^{-1}\left(\widetilde{\mathbf{s}} - \boldsymbol{\mu}\right)}, \tag{5.7}$$

which is a Euclidean distance weighted by the inverse of the covariance matrix $\boldsymbol{S}$. In our case, the distribution in question is the blade envelope, a distribution defined over the space of geometries, characterised by the mean profile $\boldsymbol{\mu}$ and the tolerance covariance $\boldsymbol{S}$; a measured profile is a point in this space, and we wish to calculate its distance away from the blade envelope. From the sample moments (5.5) and (5.6), the

Mahalanobis distance is calculated according to (5.7). If the distance is small, then the test profile $\tilde{\mathbf{s}}$ is said to lie within the blade envelope and is thus deemed acceptable. On the other hand, if the distance is large, the test profile clearly lies far away from the distribution associated with the blade envelope and therefore should not be used.

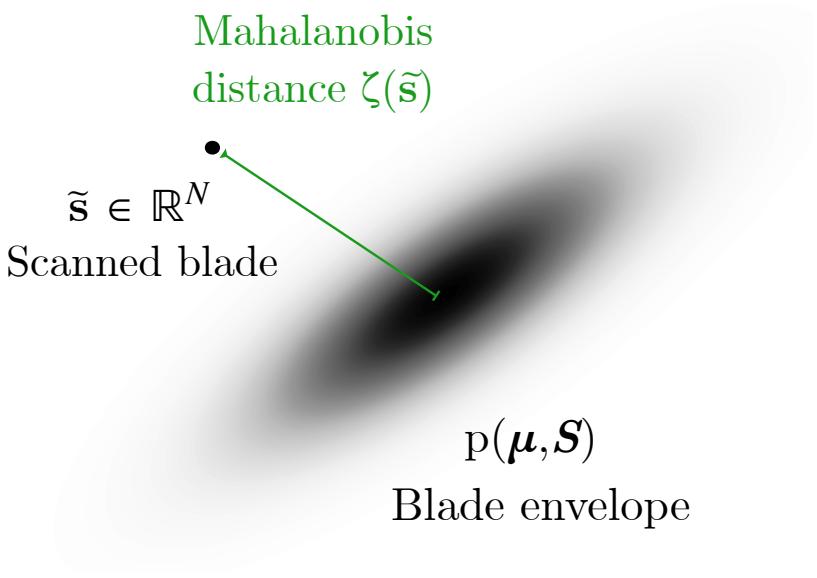


Figure 5.6 Illustrating the Mahalanobis distance metric.

This distance, which can only be interpreted relative to a base value, can be converted to an absolute confidence score via a logistic function,

$$g\left(\tilde{\mathbf{s}}\right) = \frac{1}{1 + \exp\left\{\beta_1\left(\zeta\left(\tilde{\mathbf{s}}\right) - \beta_2\right)\right\}} \tag{5.8}$$

where model parameters β_1 and β_2 are fitted to a training database consisting of geometries from the invariant subspace and random geometries from the design space, both requiring no additional CFDs to generate. To train the model parameters, invariant geometries are labelled with "1" and random geometries "0". A penalised maximum likelihood approach is used to optimise β_1 and β_2, by minimising the penalised cross-entropy loss function,

$$L(\beta_1, \beta_2) = -\frac{1}{M}\left[\sum_{i=1}^{M} y^{(i)} \log(g^{(i)}) + (1 - y^{(i)}) \log(1 - g^{(i)})\right] + C|\beta_1| \tag{5.9}$$

for some user-determined penalisation parameter $C \geq 0$ (dependent on the amount of uncertainty the user wants to propagate), where $g^{(i)} = g(\mathbf{s}^{(i)})$, the geometry of the i-th

training data point and $y^{(i)}$ is its label. The penalisation approach ensures a certain degree of uncertainty is expressed with the output predictions.

Once determined for a specific blade, expression (5.8) can be used on the factory floor to render a scrap-or-use judgment for a manufactured blade, where an output prediction close to 1 indicates confidence that a scanned geometry achieves performance standards; whereas a score close to 0 indicates the blade should be scrapped. The steps involved in generating a blade envelope and deploying it to query whether a scanned blade should be used are summarised in Figure 5.7.

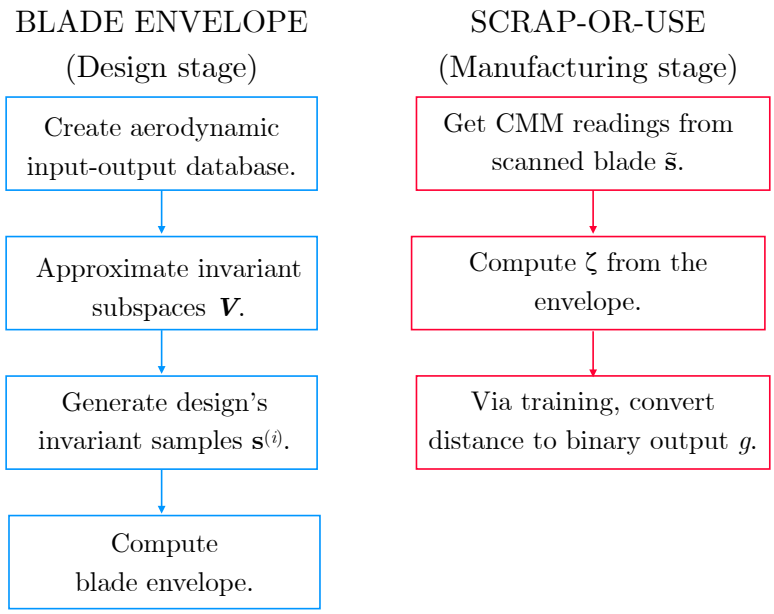


Figure 5.7 Flowchart summarising the key steps involved in generating a blade envelope and querying it to ascertain whether a scanned component should be used.

5.3.4 Finding key manufacturing modes

The Mahalanobis distance measures the deviation of a geometric profile from the distribution of invariant geometries, and forms a key component for designing tolerances. Alternatively, provided with information about the geometric variation imposed by a fixed manufacturing procedure, one may query the locations at which geometric variations cause the largest impact on aerodynamic performance. For this purpose, one can treat the Mahalanobis distance as a function of the pointwise displacement from nominal and find *key manufacturing modes* that reveal geometric modes that correspond to the largest performance deviations.

Consider the square of the Mahalanobis distance $\zeta(\mathbf{s})^2$ which is a quadratic function of the displacement $\mathbf{s}$. This is equivalent to considering the Mahalanobis distance itself since the distance is always positive. Its gradient with respect to $\mathbf{s}$ is given by

$$\nabla_{\mathbf{s}}(\zeta(\mathbf{s})^2) = 2\boldsymbol{S}^{-1}(\mathbf{s} - \boldsymbol{\mu}). \tag{5.10}$$

From this, one can form the gradient covariance matrix

$$\begin{aligned} \boldsymbol{C}_{geom} &= \mathbb{E}\left[\nabla_{\mathbf{s}}(\zeta(\mathbf{s})^2)\nabla_{\mathbf{s}}(\zeta(\mathbf{s})^2)^\top\right] \\ &= 4\boldsymbol{S}^{-1}\mathbb{E}\left[(\mathbf{s} - \boldsymbol{\mu})(\mathbf{s} - \boldsymbol{\mu})^\top\right]\boldsymbol{S}^{-1} \\ &= 4\boldsymbol{S}^{-1}\,\boldsymbol{S}_{geom}\,\boldsymbol{S}^{-1}, \end{aligned} \tag{5.11}$$

noting that $\boldsymbol{S}^{-1}$ is symmetric. The matrix $\boldsymbol{S}_{geom}$ is the covariance matrix of $\mathbf{s}$, and depends on the variation of the geometry under a certain manufacturing procedure. It can be estimated by a sample of geometries produced from this procedure, analogous to (5.5). From this, key manufacturing modes can be found by the leading eigenvectors of $\boldsymbol{C}_{geom}$. These modes thus indicate geometric variations that cause the largest variation of the Mahalanobis distance on average, which implies that they are the most likely modes to cause performance impacts.

5.4 Demonstration on test case

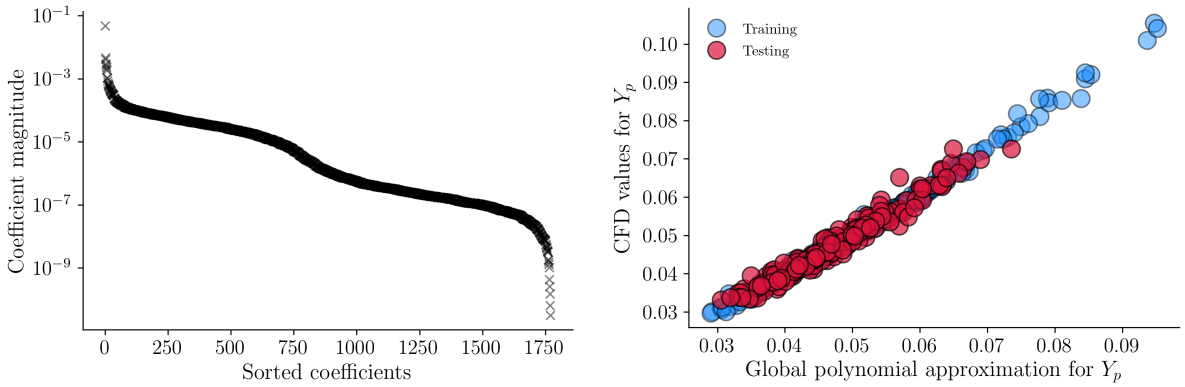
In the following, blade envelopes for the LS89 turbine are formed first with a singular objective, which is then extended to multiple scalar and vector objectives. The blade envelopes are then used to yield scrap-or-use criteria under different performance constraints.

5.4.1 Blade envelope for a single objective

Consider a blade envelope that contains geometries invariant with respect to the stagnation pressure loss coefficient,

$$Y_p = \frac{p_{02} - p_{01}}{p_{02} - p_2}, \tag{5.12}$$

where p_{02} denotes the circumferentially mass-averaged exit stagnation pressure, with the rest of the quantities defined according to Table 5.1. Using the FFD design space



(a) Polynomial coefficient magnitudes.

(b) Validation on testing points.

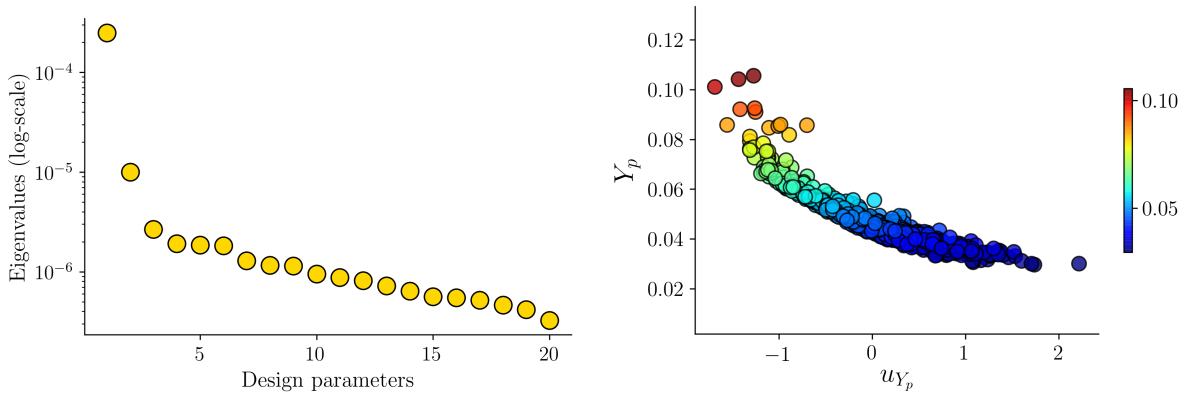
Figure 5.8 Global sparse polynomial approximation: (a) sorted coefficients; (b) validation of the model using the testing data.

$\mathcal{D} = [-1, 1]^d$ where $d = 20$, a DoE of 1000 deformed designs is formed by uniform random sampling, which is split into $M = 800$ training designs and 200 testing designs. For each design, CFD is used to calculate the corresponding loss value. A global sparse cubic polynomial on an isotropic total order basis (with 1771 basis terms) is fitted using the BPDN algorithm (see (2.43)) with the training points. The resulting coefficients are shown in Figure 5.8a. Deploying our model on the testing data, an R^2 value of 0.957 is obtained, indicating sufficient predictive accuracy (see Figure 5.8b). Using the polynomial surrogate model, gradients of the output at the training points can be approximated, which can be used to construct the gradient covariance matrix of loss $\boldsymbol{C}_{Y_p}$. This can then be used to find the active and inactive subspace matrices $(\boldsymbol{W}_{Y_p}, \boldsymbol{V}_{Y_p})$ according to the procedures detailed in Section 2.2.4.2,

$$\boldsymbol{C}_{Y_p} = [\boldsymbol{W}_{Y_p} \;\; \boldsymbol{V}_{Y_p}] \begin{bmatrix} \boldsymbol{\Lambda}_{Y_p,1} & \\ & \boldsymbol{\Lambda}_{Y_p,2} \end{bmatrix} \begin{bmatrix} \boldsymbol{W}_{Y_p}^\top \\ \boldsymbol{V}_{Y_p}^\top \end{bmatrix}. \tag{5.13}$$

The eigenvalues of this covariance matrix are reported in Figure 5.9a. Based on this decay, it is reasonable to set the active subspace to be spanned by the first column, $\boldsymbol{W}_{Y_p} \in \mathbb{R}^{d\times 1}$ and thus let the remaining 19 columns span the inactive subspace, which can be taken as the invariant subspace. The loss for all 1000 geometries is plotted against the active coordinate corresponding to design $\mathbf{x}$,

$$u_{Y_p} = \boldsymbol{W}_{Y_p}^\top \mathbf{x}, \tag{5.14}$$



(a) Eigenvalues of $\boldsymbol{C}_{Y_p}$.

(b) Sufficient summary plot on test data.

Figure 5.9 Active subspace computation: (a) eigenvalues of the covariance matrix for loss; (b) sufficient summary plot of CFD values of loss on the active coordinate.

in Figure 5.9b. This shows that the active coordinate adequately summarises the variation of loss.

5.4.1.1 Generating invariant geometries

Endowed with the invariant subspace matrix for loss $\boldsymbol{V}_{Y_p} \in \mathbb{R}^{d\times 19}$, one can now generate blade designs that have approximately the same loss as the nominal LS89. The degree of approximation is dictated entirely by the cut-off value used for partitioning the eigenvalues. Here, $H = 5000$ new geometries are generated using the hit-and-run sampling strategy detailed in Section 5.3.2. The matrices obtained for $\boldsymbol{W}_{Y_p}$ and $\boldsymbol{V}_{Y_p}$ are substituted into the bounds in (5.3) with $u = 0$, which corresponds to the active coordinate associated with the nominal blade profile (a scalar since the active subspace is 1-dimensional). The values for $\mathbf{z}$ obtained via hit-and-run sampling are then used to arrive at design vectors $\mathbf{x}$ with (5.2). To verify that these 5000 additional geometries have similar Y_p values compared to the datum, CFD is run on 500 of these designs to evaluate their Y_p values. The results are reported in Figure 5.10 and shown alongside the original training data used to fit the global polynomial. The length of the two-standard-deviation interval for the Y_p values of invariant samples is 0.0011, much smaller compared to that of unconstrained designs which is 0.0185. This offers evidence that the generated designs have comparable loss values to the LS89. Moreover, it also serves to show that the global sparse polynomial approximation offers an acceptable approximation of the loss. One important point to emphasise here is that the only computational overhead associated with generating 5000 geometries is

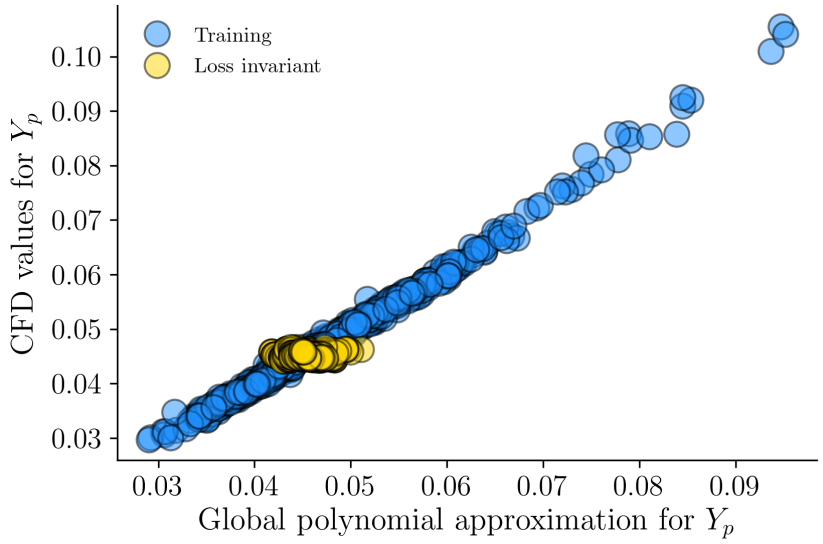


Figure 5.10 Loss invariant designs generated by sampling from the inactive subspace.

the cost of deforming the nominal mesh and generating a new mesh, and thus airfoil coordinates. While a few CFD evaluations can be run to confirm the veracity of the subspaces if allowed within the computational budget, this verification step is not strictly necessary.

After converting the design parameters for invariant designs to geometry coordinates (see Figure 5.5), one can generate the blade envelope for loss; see Figure 5.11. The LE displacement, which is the average displacement over the first 5% of the chord, defined in Section 5.1 is used to colour-code the displacements of invariant profiles to give a visual representation of their typical curvature profiles. Any blade that satisfies the tolerance covariance (visualised by the LE displacement but quantified by the tolerance covariance matrix $\boldsymbol{S}$ (5.4)) will admit loss values comparable to the nominal design.

5.4.1.2 Use or scrap?

Following the flowchart in Figure 5.7, the ensemble mean $\boldsymbol{\mu}$ and tolerance covariance matrix $\boldsymbol{S}$ associated with the different airfoils sampled from the invariant subspace are calculated. The choice of $H = 5000$ was dictated by the convergence of the two aforementioned statistical quantities in the L_2 norm sense up to 1% accuracy. The left plot of Figure 5.12 shows a heatmap of the tolerance covariance matrix $\boldsymbol{S} \in \mathbb{R}^{240\times240}$, where the colour scale represents the value of each entry in the matrix. To correctly

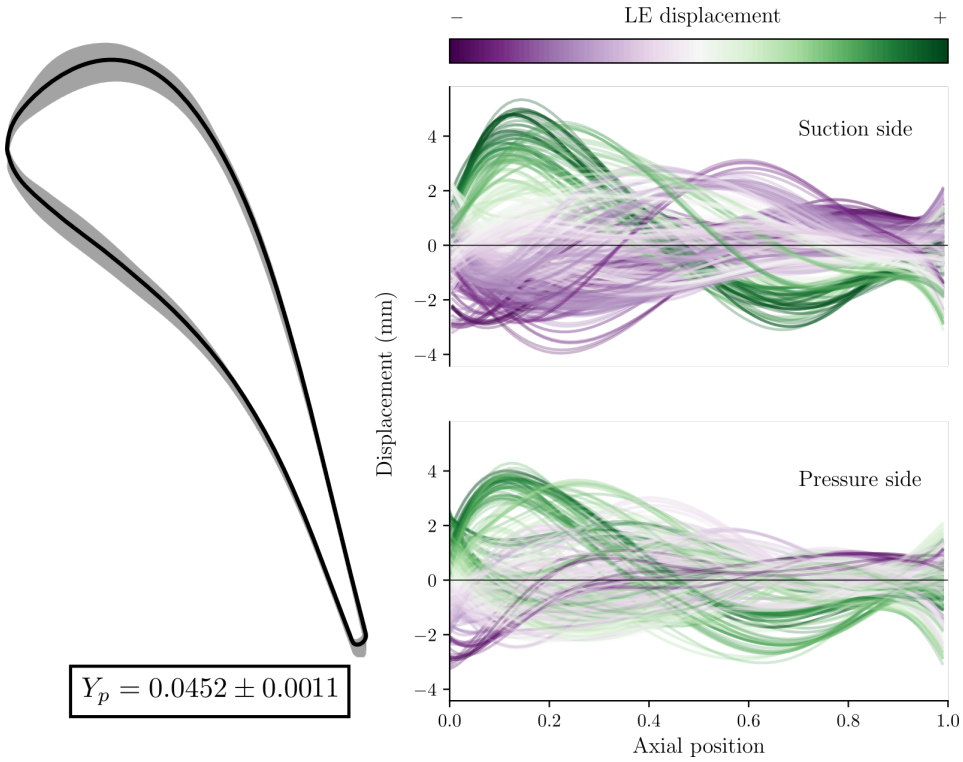


Figure 5.11 The blade envelope for loss.

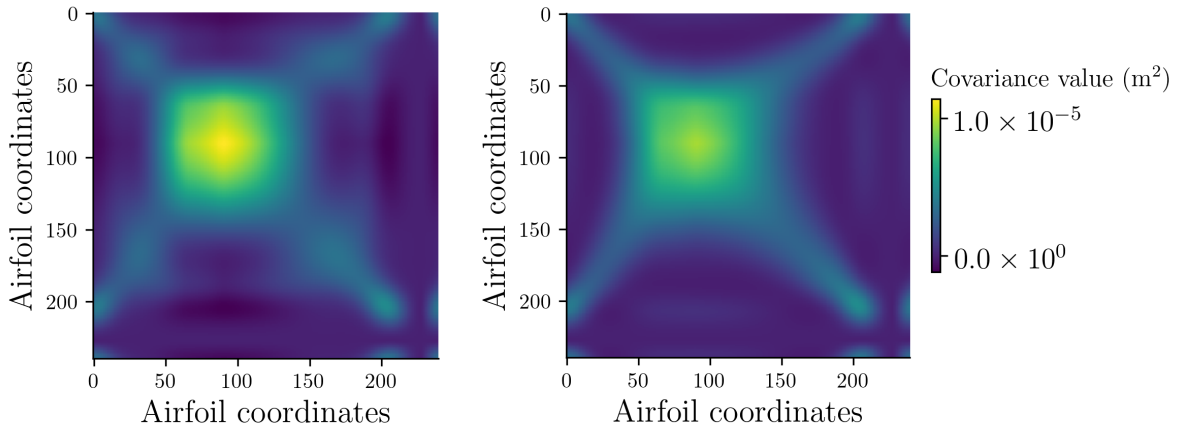
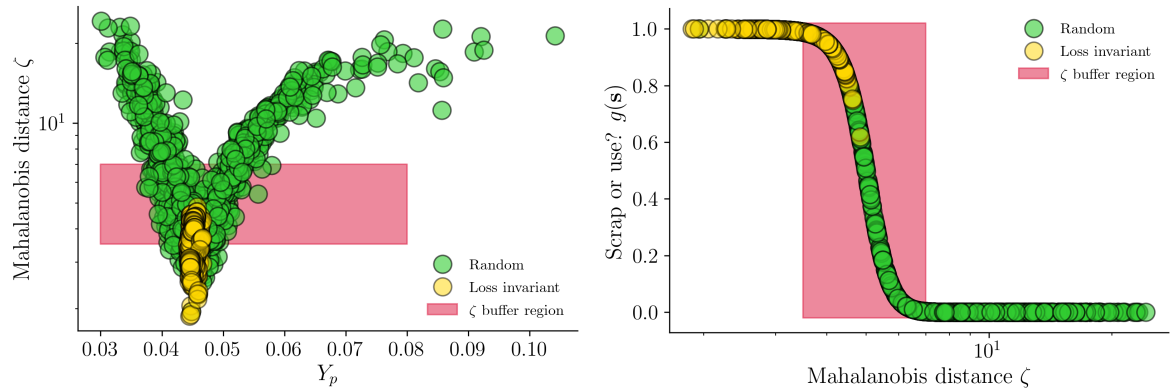


Figure 5.12 The tolerance covariance matrix $\boldsymbol{S}$ generated with $H = 5000$ inactive samples (left) and with random geometries (right). Colours indicate the value of the matrix at corresponding airfoil coordinates.

obtain these quantities, each airfoil profile has been carefully normalised to ensure that the horizontal coordinates across the profiles are the same, and thus only vertical displacements are used in estimating $\boldsymbol{\mu}$ and $\boldsymbol{S}$. On the right, this is repeated for random geometries from the original design space. Comparing the two matrices, it can be seen that extra off-diagonal correlation structures are found among the loss-invariant designs. Geometrically, these structures encapsulate the curvature requirements for a geometry to be loss-invariant.

With these metrics computed, the Mahalanobis distance between any new scanned geometry and the point cloud associated with the invariant samples can be determined. This distance is computed for 500 out of the 5000 airfoils generated from the invariant subspace and shown in Figure 5.13a. The same is plotted for the unconstrained 1000-point training-testing database, for which most of the airfoils admit different values of Y_p. A clear divergence in the loss values is observed for ζ values greater than 7; all the loss invariant designs have low values of ζ, clearly distinguishing them from unconstrained designs. To propagate a degree of uncertainty in our assessment, designs with ζ values between 3.5 and 7 are deemed to lie within a ζ *buffer region*, also demarcated in the figure.

These designs are then used to tune a scrap-or-use logistic function $g(\mathbf{s})$ as in (5.8); the resultant outputs are illustrated in Figure 5.13b. Here the horizontal axis plots ζ, while the vertical axis represents the binary outcome: a value of 1 implies that the design can be used, while a value of 0 implies that the design should be scrapped.



(a) Mahalanobis distances ζ.

(b) Binary scrap-or-use decision.

Figure 5.13 Mahalanobis distances and converted use-or-scrap confidence values for various geometries under the loss blade envelope.

A value in-between—i.e., falling within the buffer region—indicates a geometry that requires further examination before a final scrap-or-use decision.

5.4.1.3 Evaluating designs from another design space

To confirm that the blade envelope is independent of the design space parameterisation, 1000 new airfoil profiles are generated from a different design space, one comprising $d = 30$ FFD design variables, as shown in Figure 5.14. The losses of these new profiles are evaluated by running them through the SU2 suite. Then, for each profile, the ζ value is computed using the $\boldsymbol{\mu}$ and $\boldsymbol{S}$ for the loss blade envelope obtained from the 20-D design space. These distances are then fed through the aforementioned logistic model and plotted along the vertical axis in Figure 5.15. Here the colours of each marker denote the deviation of their CFD-computed loss $Y_p(\mathbf{s})$ from the nominal loss μ_{Y_p},

$$\delta(\mathbf{s}) = |Y_p(\mathbf{s}) - \mu_{Y_p}|. \tag{5.15}$$

These results attest to the notion that a blade envelope's utility in distinguishing between tolerance-abiding airfoils is not restricted to a particular parameterisation, since most of the blades with comparable loss values to the LS89 fall either below or within the ζ buffer region.

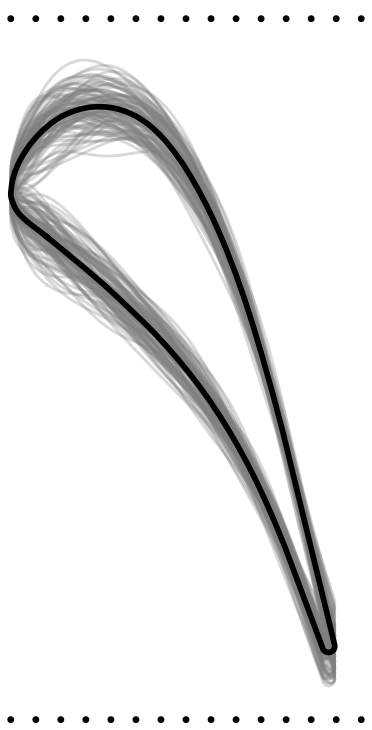


Figure 5.14 Sample designs from the larger design space of 30 FFD design variables.

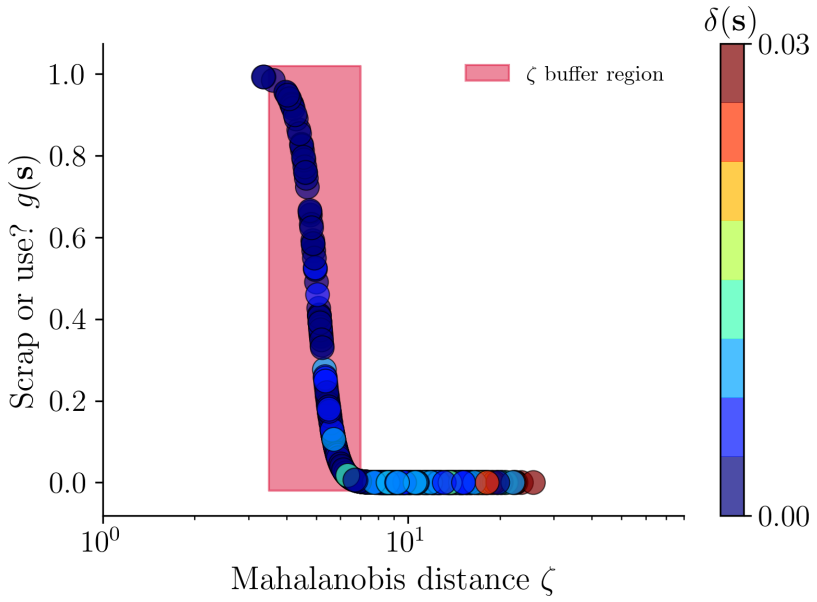


Figure 5.15 Application of the logistic model on profiles generated from the larger design space with $d = 30$ design variables. The samples are coloured according to their deviation from the nominal loss, calculated with (5.15).

5.4.2 Blade envelopes for multiple objectives

Next, the blade envelope developed so far is augmented by placing further constraints on additional objectives. During the shape design of highly-loaded turbine stages, the minimisation of stage losses is often accompanied by constraints on the flow capacity [110] or exit flow angle [149] to avoid trivial solutions where the blade is simply unloaded. Here, we consider the exit mass flow function, which is defined as follows,

$$f_m = \frac{\dot{m}\sqrt{T_{01}}}{p_{01}} \times 10^4, \tag{5.16}$$

where $\dot{m}$ is the mass flow rate and a scale factor of 10^4 is included for convenience. The units for f_m are $\mathrm{s} \cdot \mathrm{m} \cdot \mathrm{K}^{-1}$. The objective is to arrive at a blade envelope that contains designs invariant to both the loss coefficient and mass flow function.

Following a similar approach to the calculation of the invariant subspace for loss, a surrogate model for the mass flow function is constructed using an orthogonal polynomial approximation. The same DoE in the 20-D FFD design space containing $M = 800$ training samples and 200 validation samples as in the previous section is used. In this case, it is found that a linear polynomial ($p = 1$) suffices to yield an R^2 value of 0.993 on the validation data (see Figure 5.16). Since the model is linear, the gradient is constant across the input domain. This implies that one can read off the 1-D active subspace as the linear polynomial coefficients to the design parameters, i.e.,

$$f_m \approx u_{f_m} = \boldsymbol{W}_{f_m}^{\top}\mathbf{x}, \tag{5.17}$$

where $\boldsymbol{W}_{f_m} \in \mathbb{R}^{20\times 1}$ specifies the active subspace to be taken as the dimension-reducing subspace, and u_{f_m} is the active coordinate for f_m.

Equipped with the dimension-reducing subspaces for both loss ($\boldsymbol{W}_{Y_p} \in \mathbb{R}^{20\times 1}$) and mass flow ($\boldsymbol{W}_{f_m} \in \mathbb{R}^{20\times 1}$), one can now find the intersection of their invariant subspaces via the method described in Section 4.1. The dimension-reducing subspaces are linearly independent and the intersection $\mathcal{V}_{int}$ is $20 - (1 + 1) = 18$-dimensional. Then, using the hit-and-run sampling strategy (see Section 5.3.2), $H = 5000$ samples are generated from $\mathcal{V}_{int}$. The invariance in both objectives is verified by picking 500 designs and running them through the CFD solver to obtain their true loss and mass flow function values, which are plotted against the corresponding values for the training data-set in Figure 5.17. It can be seen that the variation in both loss and mass flow is much smaller for invariant designs compared to random ones. The length of

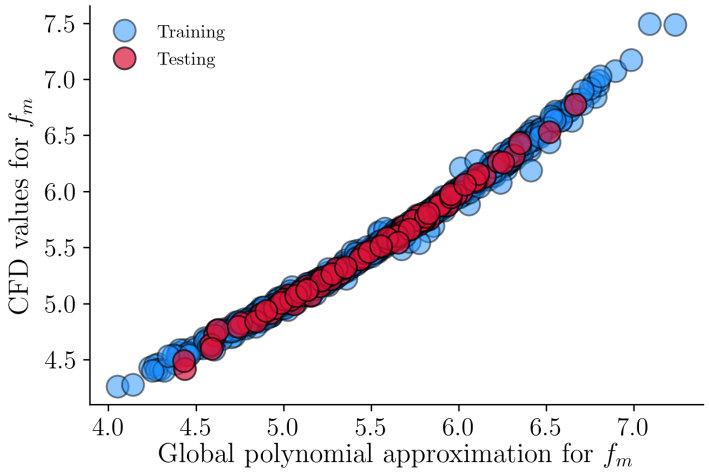


Figure 5.16 Validation of the polynomial model for the mass flow function. This coincides with the sufficient summary plot because the global polynomial approximation is a linear model.

the two-standard-deviation interval for Y_p has been reduced from 0.0185 to 0.0011, while that for f_m from 0.847 to 0.008.

The blade envelope for invariance in loss and mass flow is shown in Figure 5.18. Compared to the blade envelope for loss in Figure 5.11, the pointwise tolerance is stricter from mid-chord to the trailing edge because an extra performance constraint on the mass flow function is imposed. The typical curvature of the sample profiles is also milder. Using the sample profiles, the ensemble mean and covariance matrix can be calculated and the Mahalanobis distance can be computed for test profiles. This is then input to an appropriately parameterised logistic function to calculate a scrap-or-use confidence value as before.

Figure 5.19 shows the application of the trained logistic function on profiles from two different design spaces: the 20-D design space from which the training data is drawn, and the 30-D design space described in Section 5.4.1.3. The samples from the 30-D space are coloured according to how close their loss and mass flow values are to nominal values calculated according to the following formula

$$\delta(\mathbf{s}) = \sqrt{\left(\mathbf{y}(\mathbf{s}) - \boldsymbol{\mu}_y\right)^\top \boldsymbol{S}_y^{-1} \left(\mathbf{y}(\mathbf{s}) - \boldsymbol{\mu}_y\right)}, \tag{5.18}$$

where $\mathbf{y}(\mathbf{s}) = [Y_p(\mathbf{s}),\ f_m(\mathbf{s})]^\top$ are the output values for the geometry $\mathbf{s}$, and $\boldsymbol{\mu}_y$ and $\boldsymbol{S}_y$ are the ensemble mean and covariance matrix of $\mathbf{y}$ over all training designs respectively. Here, $\delta(\mathbf{s})$ is another Mahalanobis distance, this time defined over the two output objectives. This is analogous to that defined over geometric profiles. Here, similar

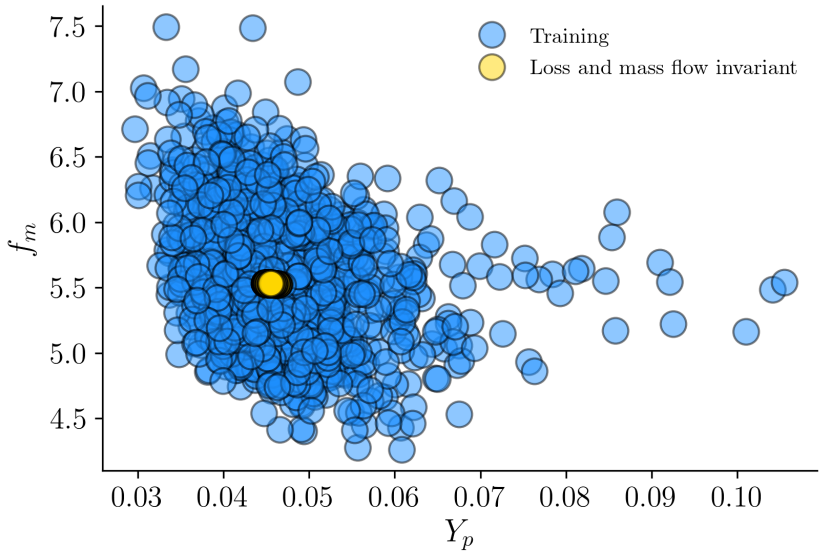


Figure 5.17 Loss and mass flow function for random training designs (blue) and invariant designs within the intersection of the inactive subspaces (yellow).

results as those obtained for a single objective shown in Figure 5.10 can be seen: geometries with low Mahalanobis distance and high scrap-to-use value have low deviation $\delta(\mathbf{s})$ from nominal loss and mass flow. This shows that the Mahalanobis distance defined over geometries is an effective metric for identifying profiles invariant in multiple objectives.

5.4.2.1 Key manufacturing modes for loss and mass flow

Suppose that the geometric variation is distributed according to the designs that are generated under the 20-D FFD design space with a uniform distribution on the FFD amplitudes. Following the steps described in Section 5.3.4, the gradient covariance matrix $\boldsymbol{C}_{geom}$ is computed and key manufacturing modes are inferred via eigendecomposition. Figure 5.20 reports the largest 30 out of 240 eigenvalues of $\boldsymbol{C}_{geom}$ (defined over the space of airfoil coordinates), which indicate that two eigenvectors can summarise a large portion of the eigenvalue spectrum. The component magnitudes of these two eigenvectors are displayed in Figure 5.21, overlaid on the nominal geometry. This figure shows that geometric variations near the midspan on the suction side as well as the trailing edge are the most important factors leading to performance deviation. This agrees with studies performed on the shape design of the LS89 profile. For example, Torreguitart et al. [149] highlights the importance of the trailing edge shape in determining the base pressure, which has a significant impact on profile

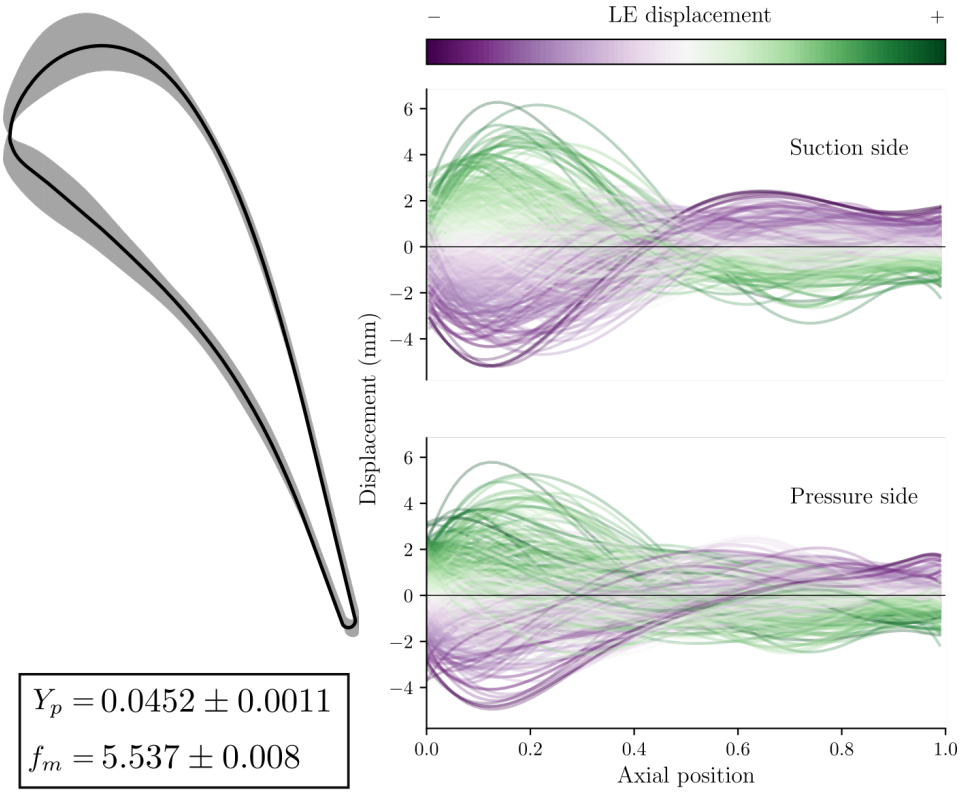
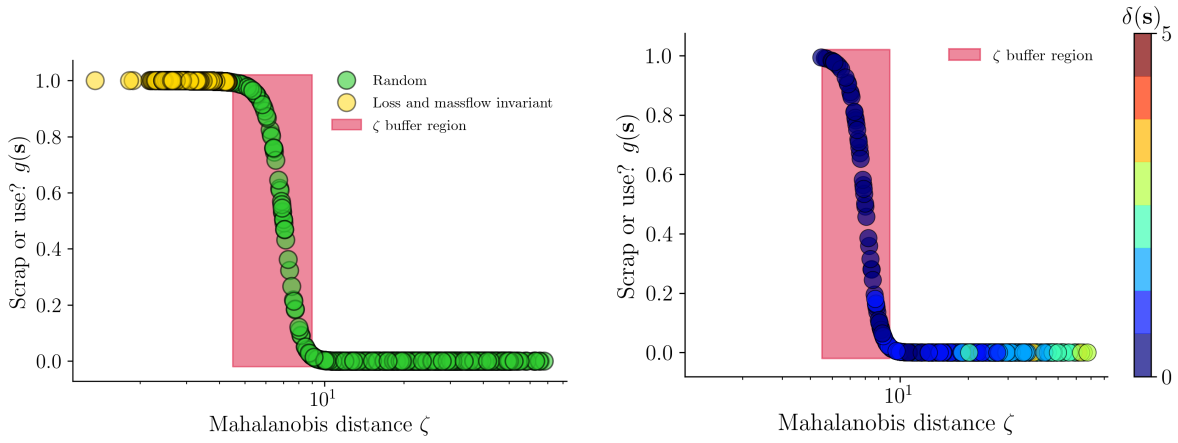


Figure 5.18 The blade envelope for loss and mass flow.



(a) 20-D design space.

(b) 30-D design space.

Figure 5.19 Trained logistic function for applying a binary scrap-or-use decision on profiles, requiring invariance in both the loss and mass flow function; (a) shows samples from the 20-D design space and (b) shows samples from the 30-D design space described in Section 5.4.1.3. In (b), the samples are coloured according to their deviation from nominal loss and mass flow, quantified via (5.18).

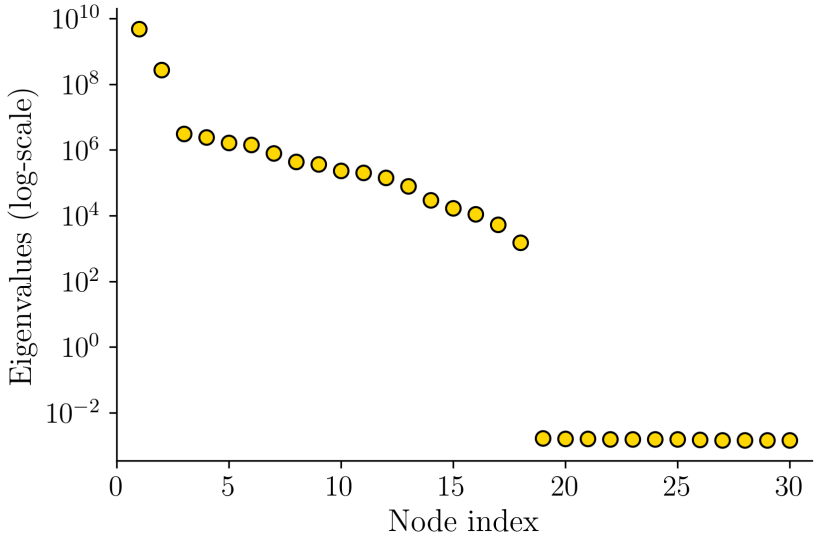


Figure 5.20 The largest 30 eigenvalues of $\boldsymbol{C}_{geom}$ for constraining loss and mass flow.

losses. The results shown in Figure 5.21 can be useful to inform engineers on where one should focus on when scanning manufactured components, reducing scanning times by avoiding areas where large variations can be acceptable.

Compared to a similar analysis performed on the FFD design space (e.g. by computing the gradient covariance matrix of the loss and mass flow as functions of the design space parameters), the results obtained here are more readily interpretable, highlighting regions on the blade itself that require attention. Moreover, multiple performance criteria are naturally incorporated with the present method.

5.4.3 Conditional tuning of flow properties for inverse design

The identification of flow features that are critical to the performance of bladed components has played an important role in the shape design of blades. The work by Goodhand [62] highlights the importance of the pressure distribution near the leading edge of compressor blades. For turbine blades, Clark [20] establishes the relation between aerodynamic features, defined by characteristics of the isentropic Mach number distribution over different regions on the blade surface, and aerodynamic performance. The isentropic Mach number is given by

$$M(s) = \sqrt{\frac{2}{\gamma - 1}\left(\left(\frac{p_{01}}{p(s)}\right)^{\frac{\gamma-1}{\gamma}} - 1\right)}, \tag{5.19}$$

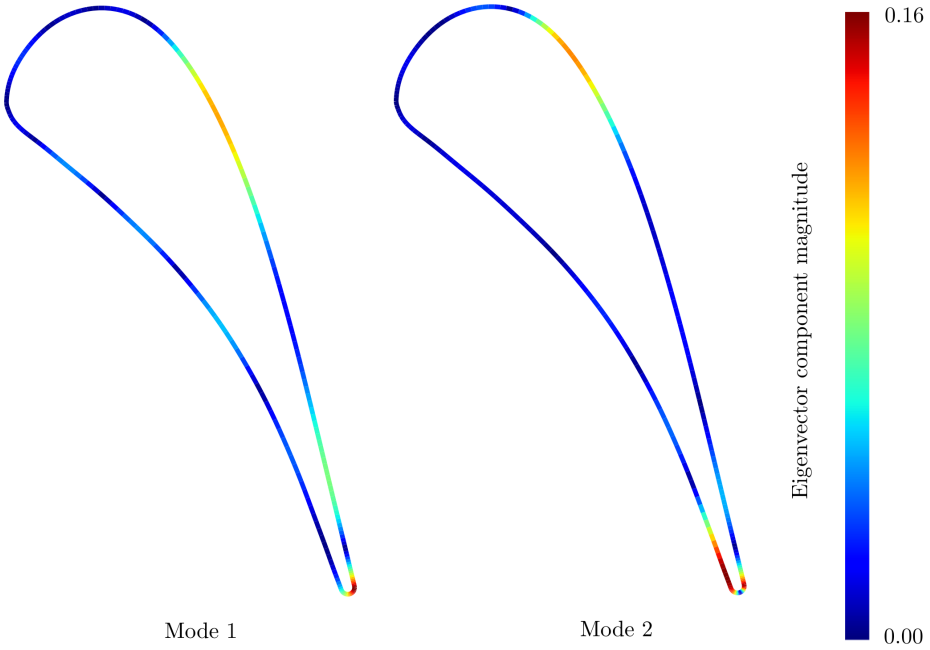


Figure 5.21 First two key manufacturing modes for loss and mass flow.

where $p(s)$ is the local static pressure at location s on the surface and other quantities are defined in Table 5.1. This profile QoI can be discretised into $N = 240$ measurements at $(s_1, s_2, ..., s_N)$, implying that the isentropic Mach number distribution can be defined by a vector-valued function,

$$\begin{aligned} \mathbf{M} &= [M(s_1),\ M(s_2),\ ...,\ M(s_N)]^\top \\ &= [M_1,\ M_2,\ ...,\ M_N]^\top. \end{aligned} \tag{5.20}$$

Each component of this distribution can be written as a function of the shape perturbation parameters. In the following, the surface isentropic Mach number is included in the blade envelope as a vector-valued function of shape perturbations to achieve control over key flow features.

Using the methodology described in Section 4.1.2, invariant subspaces of the isentropic Mach number distribution can be found. As explained in that section, since the number of components N considered here is much larger than the underlying design space dimension d, treating the distribution as a vector-valued objective is more appropriate than the intersection approach. The weight matrix $\boldsymbol{R}$ defined in (4.15) can be used to adjust the degrees of invariance required at different locations on the surface. This offers the flexibility to control only certain parts of the flow while allowing other parts to vary, thus resulting in a range of designs that satisfy the required constraints.

Note that this is similar to the objective of *inverse design*, where a target distribution is specified and the baseline geometry is modified iteratively to achieve that distribution. However, instead of a single final design, the present approach identifies a distribution of designs.

While the vector-valued invariant subspaces can yield designs that have constrained isentropic Mach numbers at specified regions, one can also use the corresponding dimension-reducing subspaces to conditionally adjust local flow properties while keeping other QoIs constant. To make this clear, assume that one intends to keep O objectives $(f_1, f_2, ..., f_O)$ invariant, where they may be scalar-valued or vector-valued. Provided that $O \ll d$, it is reasonable to further assume that their corresponding dimension-reducing subspaces spanned by $(\boldsymbol{W}_1, \boldsymbol{W}_2, ..., \boldsymbol{W}_O)$ are distinct and $\text{colspan}(\boldsymbol{W}_i) \nsubseteq \text{colspan}(\boldsymbol{W}_j)$ for any i, j. Following the intersection method, one can obtain a decomposition of the design space characterised by a matrix with independent columns

$$\boldsymbol{D} = \begin{bmatrix} \boldsymbol{W}_1 & \boldsymbol{W}_2 & ... & \boldsymbol{W}_O & \boldsymbol{V}_{int} \end{bmatrix}, \tag{5.21}$$

where $\boldsymbol{V}_{int}$ is the multi-objective invariant subspace matrix. Corresponding to $\boldsymbol{D}$, a coordinate transformation on the design $\mathbf{x}$ yields the following

$$\begin{aligned} \boldsymbol{D}^\top \mathbf{x} &= \begin{bmatrix} \left(\boldsymbol{W}_1^\top \mathbf{x}\right)^\top & \left(\boldsymbol{W}_2^\top \mathbf{x}\right)^\top & ... & \left(\boldsymbol{W}_O^\top \mathbf{x}\right)^\top & \left(\boldsymbol{V}_{int}^\top \mathbf{x}\right)^\top \end{bmatrix}^\top \\ &= \begin{bmatrix} \underbrace{\mathbf{u}_1^\top & \mathbf{u}_2^\top & ... & \mathbf{u}_O^\top}_{n_1+n_2+...+n_O} & \mathbf{z}^\top \end{bmatrix}^\top. \end{aligned} \tag{5.22}$$

The first components of the new coordinates $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_O$ correspond to the active coordinates of the objectives $f_1, f_2, ..., f_O$; the final coordinates $\mathbf{z}$ correspond to the inactive coordinates. Any perturbation in the column space of $\boldsymbol{V}_{int}$ has its first $n_1 + n_2 + ... + n_O$ coordinates equal to zero, implying that it causes no change to the active coordinates for all objectives. Thus, this perturbation does not cause significant change to any objective. On the other hand, the active coordinates can also be treated as tunable parameters that directly control each objective separately. Adjusting the values of $\mathbf{u}_1$ while keeping every other active coordinate constant results in changes in f_1, while maintaining the values of other objectives. Note that the inactive coordinates $\mathbf{z}$ can be allowed to float, since those coordinates do not affect any objective.

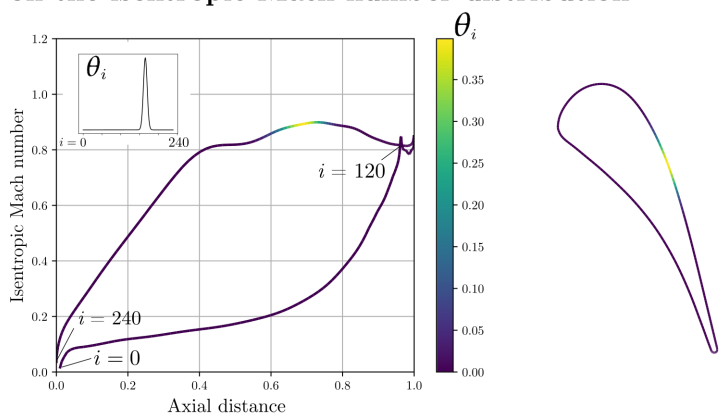
Below, we combine the dimension-reducing subspaces of the loss coefficient, mass flow function and the isentropic Mach number distribution and demonstrate how one

can find the range of compatible geometries obeying invariance in loss and mass flow, as well as exert control over the flow profile at specified regions.

5.4.3.1 Controlling the peak isentropic Mach number

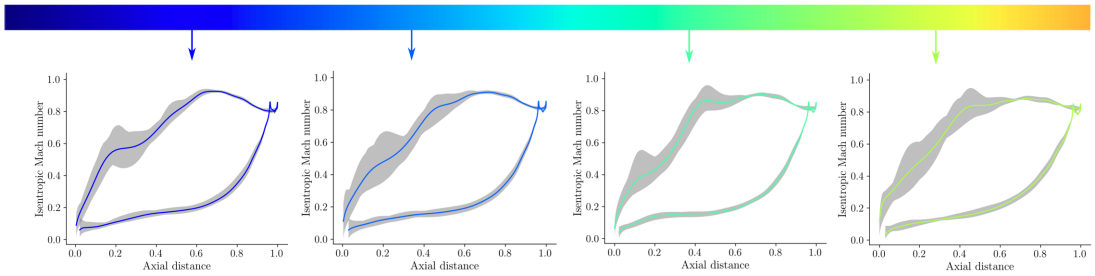
First, we aim to control the variation of the peak of the surface isentropic Mach number distribution by extracting a small number of active coordinates using vector-valued dimension reduction. Following the methodology in Section 4.1.2, the gradient covariance matrix for the isentropic Mach number distribution $\boldsymbol{H}_{peak}$ is required, where the weight matrix $\boldsymbol{R}_{peak}$ is set such that diagonal elements corresponding to the peak are large. This is illustrated in Step I in Figure 5.22, where the colour indicates the magnitude of the corresponding weights. To compute the necessary gradients with respect to each nodal isentropic Mach number, quadratic models are trained at each node using uniformly sampled designs from the full design space. When evaluated on independent validation data, the models on nodes with non-zero weights all achieved an R^2 value above 0.99. Note that non-zero weights are set on multiple nodes in the vicinity of the peak. As the height of the peak changes, it is observed that the chord-wise location of the peak shifts slightly. This effect is accommodated by smoothing out the weights to a small region near the peak.

I. Set weights on the isentropic Mach number distribution



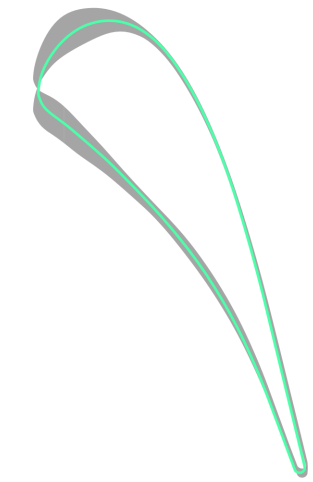
II. Tune desired peak isentropic Mach number

Increasing $u_{peak,1}$ ⟶



$u_{peak,1}$	−0.19	−0.13	0.00	0.06
$\bar{M}_{peak}$	0.93	0.91	0.90	0.88
$(s/c)_{peak}$	0.68	0.72	0.73	0.73

III. Sample within the inactive subspace to get a blade envelope



$$Y_p = 0.0452 \pm 0.0011$$
$$f_m = 5.537 \pm 0.008$$
$$M_{peak} = 0.901 \pm 0.014$$

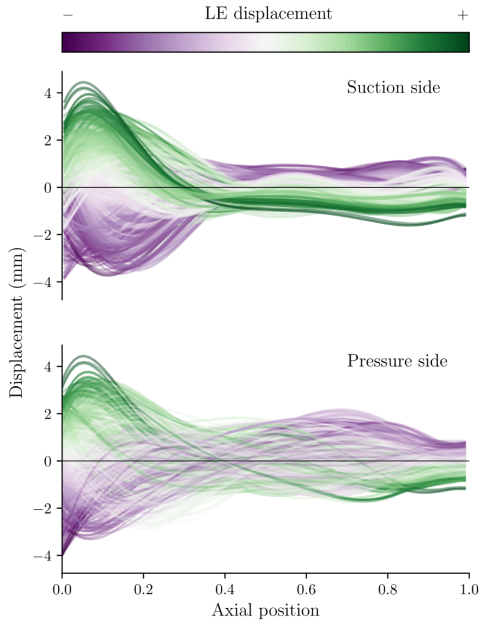


Figure 5.22 Conditional tuning of the peak isentropic Mach number.

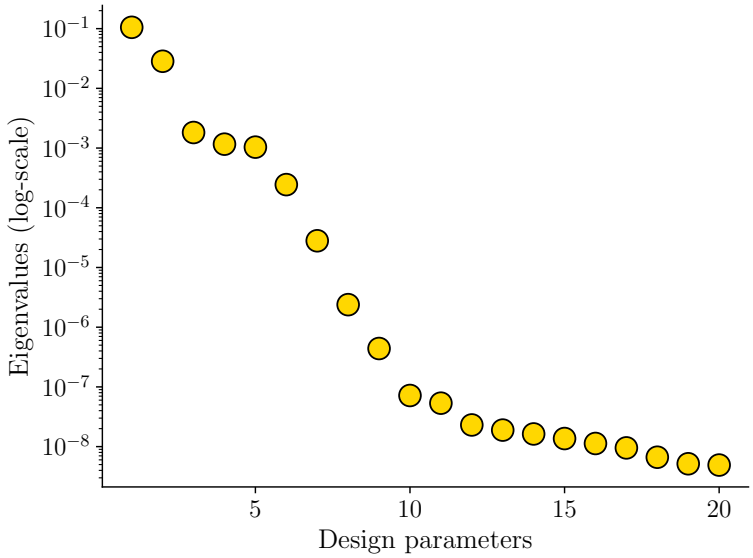


Figure 5.23 Eigenvalues of $\boldsymbol{H}_{peak}$, weighted for the peak isentropic Mach number.

Carrying out the eigendecomposition of $\boldsymbol{H}_{peak}$,

$$\boldsymbol{H}_{peak} = [\boldsymbol{W}_{peak} \;\; \boldsymbol{V}_{peak}] \begin{bmatrix} \boldsymbol{\Lambda}_{peak,1} & \\ & \boldsymbol{\Lambda}_{peak,2} \end{bmatrix} \begin{bmatrix} \boldsymbol{W}_{peak}^{\top} \\ \boldsymbol{V}_{peak}^{\top} \end{bmatrix}. \tag{5.23}$$

The eigenvalues of $\boldsymbol{H}_{peak}$ are shown in Figure 5.23. The active coordinates can be defined using the first two eigenvectors for approximate control over the peak isentropic Mach number, namely,

$$\begin{aligned} \mathbf{u}_{peak} &= \begin{bmatrix} u_{peak,1} & u_{peak,2} \end{bmatrix}^{\top} \\ &= \boldsymbol{W}_{peak}^{\top} \mathbf{x}. \end{aligned} \tag{5.24}$$

This set of active coordinates can be combined with those corresponding to the loss coefficient and mass flow to form the following vector

$$\mathbf{u} = \begin{bmatrix} u_{Y_p} & u_{f_m} & u_{peak,1} & u_{peak,2} \end{bmatrix}^{\top}. \tag{5.25}$$

To control the peak isentropic Mach number while keeping the loss and mass flow constant, the final two coordinates of $\mathbf{u}$ can be modified while keeping the first two unchanged. In this example, we only tune the first active coordinate $u_{peak,1}$ for illustra-

tive purposes. The effect of changing this variable is shown in Step II in Figure 5.22. Increasing $u_{peak,1}$ results in the flattening of the *mean* isentropic Mach number distribution (averaged over variations in the invariant subspace). The location of the peak of the mean isentropic Mach number distribution moves gradually towards the trailing edge. If the coordinate is increased further, the mean peak shifts to the junction at the start of the plateau near mid-chord. The uncertainty bands around the distributions indicate the variation in the isentropic Mach number distribution for geometries with the same values of $\mathbf{u}$. It should be noted that these uncertainty bands imply that some designs may have an earlier peak than predicted by the mean. Tighter control on the shape and location of the peak can be achieved by increasing the number of active coordinates, or using a more diffuse set of weights on the $\boldsymbol{R}$ matrix.

After setting appropriate values for the active coordinates, the hit-and-run sampling algorithm can be used to sample designs with these coordinates. A blade envelope can be formed from these samples, following the same procedure as the previous sections. Step III in Figure 5.22 shows the blade envelope with $\mathbf{u} = \mathbf{0}$. Compared with the previous blade envelopes, the tolerance on the profile from mid-chord to the trailing edge is reduced, especially on the suction side. Using the sample profiles, the tolerance covariance matrix and ensemble mean can be calculated to arrive at an automatic decision criterion for the scrapping of manufactured blades, taking into account all three objectives.

5.4.3.2 Controlling the leading edge isentropic Mach number

The procedure can be repeated for a different setting of the weights on the isentropic Mach number distribution. Setting the weights as in Step I of Figure 5.24 with a distribution centered around 20% of the chord, the focus is now placed on the isentropic Mach number characteristics when the flow is accelerating before the peak on the suction side. Although this region of acceleration can span a significant portion of the chord including that near the leading edge, we call it the LE isentropic Mach number for brevity. Carrying out the eigendecomposition of the vector gradient covariance matrix $\boldsymbol{H}_{LE}$,

$$\boldsymbol{H}_{LE} = [\boldsymbol{W}_{LE} \;\; \boldsymbol{V}_{LE}] \begin{bmatrix} \boldsymbol{\Lambda}_{LE,1} & \\ & \boldsymbol{\Lambda}_{LE,2} \end{bmatrix} \begin{bmatrix} \boldsymbol{W}_{LE}^{\top} \\ \boldsymbol{V}_{LE}^{\top} \end{bmatrix}, \tag{5.26}$$

the spectrum shown in Figure 5.25 is obtained. Choosing the first four eigenvectors as the basis for the dimension-reducing subspace $\boldsymbol{W}_{LE}$, the active coordinates are

specified as

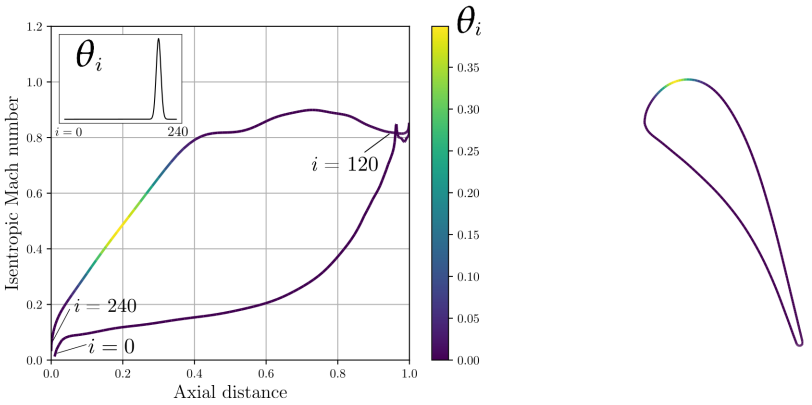
$$\begin{aligned}\mathbf{u}_{LE} &= \begin{bmatrix} u_{LE,1} & u_{LE,2} & u_{LE,3} & u_{LE,4} \end{bmatrix}^\top \\ &= \boldsymbol{W}_{LE}^\top \mathbf{x}.\end{aligned} \tag{5.27}$$

Concatenating this set of coordinates with the active coordinates of the loss and mass flow, we get

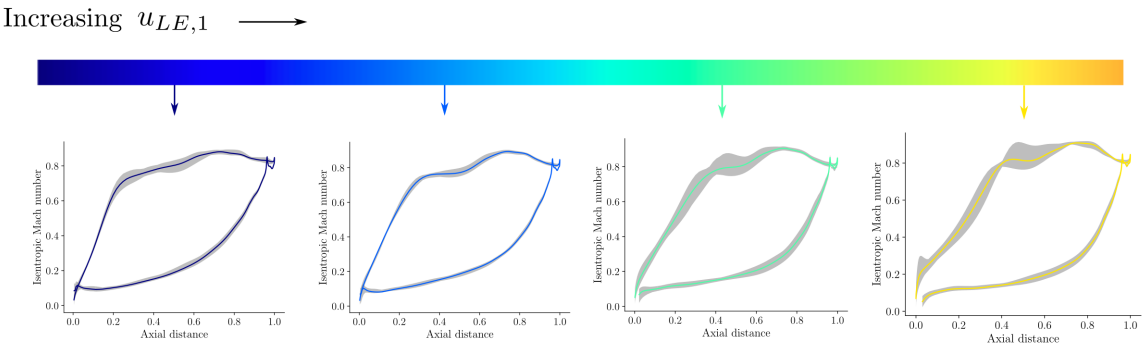
$$\mathbf{u} = \begin{bmatrix} u_{Y_p} & u_{f_m} & u_{LE,1} & u_{LE,2} & u_{LE,3} & u_{LE,4} \end{bmatrix}^\top. \tag{5.28}$$

The effect of tuning the first active coordinate of the leading edge isentropic Mach number $u_{LE,1}$ is shown in Step II of Figure 5.24. As the coordinate is increased, the isentropic Mach number is reduced towards the leading edge, and the acceleration is milder and more distributed towards mid-chord. Step III shows the blade envelope obtained with samples setting $\mathbf{u} = \mathbf{0}$. The geometric tolerance near the leading edge is tightened, especially on the suction side.

I. Set weights on the isentropic Mach number distribution

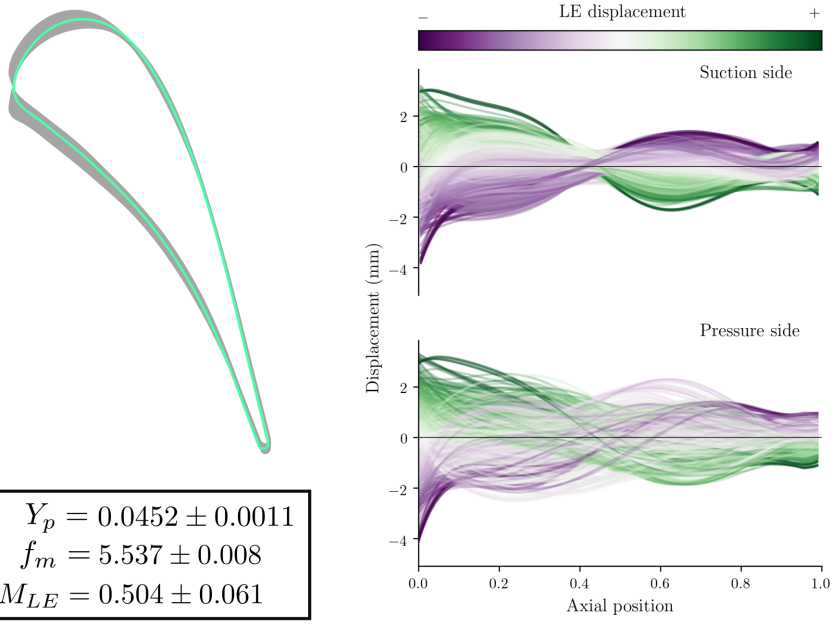


II. Tune desired leading edge isentropic Mach number



$u_{LE,1}$	-1.00	-0.50	0.00	0.50
$\bar{M}_{LE}$	0.63	0.57	0.50	0.44

III. Sample within the inactive subspace to get a blade envelope



$$Y_p = 0.0452 \pm 0.0011$$
$$f_m = 5.537 \pm 0.008$$
$$M_{LE} = 0.504 \pm 0.061$$

Figure 5.24 Conditional tuning of the leading edge isentropic Mach number.

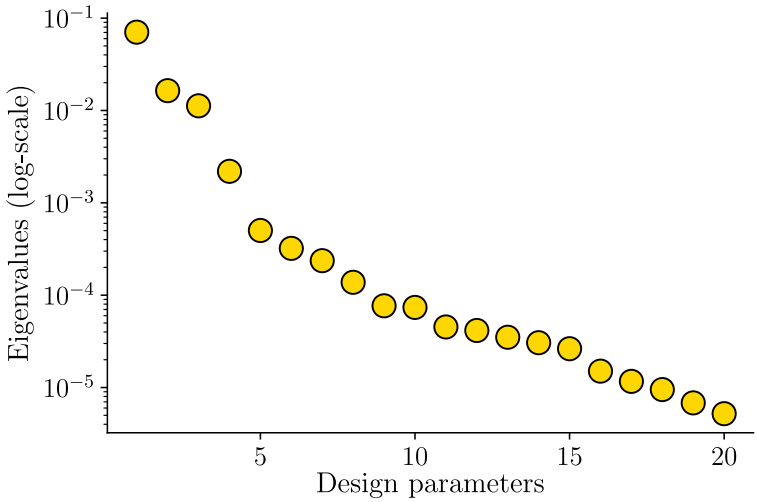


Figure 5.25 Eigenvalues of $\boldsymbol{H}_{LE}$, weighted for the leading edge isentropic Mach number.

5.5 Conclusions for chapter

In this chapter, blade envelopes are introduced, detailing the statistical machinery that forms the backbone of the framework. For demonstration, blade envelopes for the LS89 turbine profile are calculated taking different performance objectives into consideration. As a set of computational techniques, the blade envelopes furnish distributions that can be leveraged to generate geometries that are constrained in multiple objectives, including surface flow characteristics at specified locations on the profile. In addition, the distribution can be used to distinguish between manufactured geometries that can or cannot be used in a quantitative manner by calculating the scrap-or-use value based on the Mahalanobis distance, without additional CFD solves.

Although the bulk of the blade envelopes framework is best interpreted computationally, blade envelopes can also be visualised to aid discussions on tolerance and trade-offs during the design stage of bladed components. Figure 5.26 shows a 3-D render of the blade envelope for loss and mass flow, produced using the open-source rendering software Blender[2]. The purple and green airfoils correspond to profiles with negative and positive leading edge displacements respectively, and the black profile shows the nominal geometry. Example invariant geometries can be highlighted for inspection, such as the one shown in blue here. These renders can be straightforwardly adapted to computer aided design workflows; they can also be 3-D printed or projected onto real life surfaces via augmented reality, facilitating discussions between design

[2]https://www.blender.org

Figure 5.26 3-D rendering of the loss and mass flow blade envelope (Figure 5.18).

engineers, manufacturing engineers, engine maintenance inspectors and more. A few examples of design insights that can be revealed by blade envelopes are captured below.

1. Guiding design trade-offs, such as the feasibility of increasing trailing edge thickness without loss of performance.
2. Informing engineers where to focus scanning efforts via key manufacturing modes.
3. Comparing the tolerance implications across multiple nominal designs with different corresponding blade envelopes.

Thus, it can be envisioned that blade envelopes can be a valuable asset to the robust design of bladed components, bringing forth a step-change in the way tolerances are drafted.

Chapter 6

Conclusions

The aim of this thesis is to extend and build upon fundamental ideas within orthogonal polynomial approximations and model-based dimension reduction to construct novel computational methods for different applications in computational engineering. The main contributions presented are summarised below.

- A new method of *sensitivity analysis through polynomial ridge approximations* is described. Through comparisons with existing polynomial-based techniques, it is shown that polynomial ridge approximations can accurately evaluate sensitivity indices of computational models, especially those with low-dimensional dependence structures.
- A novel set of sensitivity indices, the *extremum Sobol' indices*, can be used to find sensitivities of model parameters when the output is near extrema. Empirical comparisons are made between these indices and skewness-based indices, revealing qualitative similarities.
- Ridge approximation methods are extended to consider multiple objectives. Two application areas based on multi-objective dimension reduction are studied:
  - In the context of simulating scalar fields, *embedded ridge approximations* are proposed where ridges are fitted at each node of the field. Savings in both computational time and storage space can be achieved by exploiting smoothness properties of the underlying physics.
  - By considering invariant subspaces, one or multiple QoIs can be kept approximately invariant. Based on this idea, the novel framework of *blade envelopes* is formulated as a guideline for the tolerance design of bladed com-

ponents, providing quick performance-based sentencing of manufactured geometries.

Although the first concepts of polynomial approximations and dimension reduction date back centuries, the ever-growing prevalence of computational modelling in tackling a wide range of real-world applications continues to encourage research for more accurate and tailored surrogate models in high dimensions given a limited computational budget. For this reason, potential directions for further research are plentiful within this area, some of which are listed below.

- **Bayesian formulation of polynomials and ridges**: A growing trend in the numerical analysis community is the probabilistic modelling of classical computational procedures [22], such as numerical quadrature [116, 117] and conjugate gradients [21]. At the same time, there is growing interest in quantifying and propagating uncertainties associated with surrogate modelling. Bayesian methods allow the expression of uncertainty in model parameters and predictions through posterior distributions. One avenue of research would be to examine the interaction of Bayesian methods with orthogonal polynomials and ridge approximations (such as [90]).

- **Using polynomial spline models for approximation**: Through an example of approximating the flow field around a 3-D transonic wing, the authors in [131] show that polynomial ridge approximations remain effective in the presence of discontinuous trends, although the polynomial degree needs to be increased. Polynomial spline models are compositions of locally fitted polynomials, which can reduce the polynomial degree required. In upcoming work, the benefits of an orthogonal polynomial approach to spline models are explored.

- **Extending blade envelopes**: The concepts behind blade envelopes can be applied to a wider range of cases beyond 2-D turbomachinery blades, such as 3-D geometries. The definition of the distribution can be placed on other types of (not necessarily geometric) measurements that can be quantified with a tolerance covariance, extending the framework to general manufacturing operations.

- **Tailored DoE points for ridge approximations**: A more exploratory area of research would be the search for efficient points for furnishing ridge approximation models. Does there exist a (possibly adaptive) sampling scheme that can more accurately detect the presence and deduce the appropriate dimension-reducing subspace than Monte Carlo using the input distribution?

Appendix A

Appendix: Definition of some terms

In this appendix, terms underlined in the main text are defined for reference.

- **Hilbert space**: An inner product space is a vector space V over the real or complex numbers, that is equipped with an inner product $\langle\rangle : V \times V \to \mathbb{R}$ or $\mathbb{C}$. This inner product satisfies

  1. $\langle u, v\rangle = \overline{\langle v, u\rangle}$ for all $u, v \in V$.
  2. $\langle \lambda_1 u + \lambda_2 v, w\rangle = \lambda_1\langle u, w\rangle + \lambda_2\langle v, w\rangle$ for all $u, v, w \in V$.
  3. $\langle u, u\rangle \geq 0$ for all u with equality if and only if $u = 0$.

  In this case, the inner product defines a norm

  $$\|u\| = \sqrt{\langle u, u\rangle}. \tag{A.1}$$

  With regards to this norm, a Cauchy sequence is a sequence $u_1, u_2, ...$ such that for every positive number ε, there exists a positive integer N such that

  $$\|u_n - u_m\| < \varepsilon, \tag{A.2}$$

  for all $n, m > N$.

  A *Hilbert space* is then an inner product space that is complete; that is, every Cauchy sequence has a limit that is also in the space.

- **Poincaré inequality**: Consider a Hilbert space $L_2(\omega)$ on a convex domain $\mathcal{D}$. Define a function u on the domain such that it has zero mean on $\mathcal{D}$ with respect to ω, is Lipschitz continuous and differentiable. The *Poincaré inequality* states

that

$$\|u\|_{L_2(\omega)} \leq C \|\nabla u\|_{L_2(\omega)}, \tag{A.3}$$

where C is a constant that depends on the domain $\mathcal{D}$ and the density ω.

- **Matrix Bernstein inequality**: (Adapted from part of Theorem 1.6.2 of [154]) Let $S_1, S_2, ..., S_n$ be independent random matrices with common dimension $d_1 \times d_2$. Assume that

$$\mathbb{E}[S_k] = 0 \text{ and } \|S_k\|_2 \leq L, \tag{A.4}$$

where $\|\cdot\|_2$ denotes the operator 2-norm (the largest singular value). Consider the sum

$$Z = \sum_{k=1}^{n} S_k. \tag{A.5}$$

The matrix variance is defined as

$$v(Z) = \max\left\{\mathbb{E}[ZZ^\top], \mathbb{E}[Z^\top Z]\right\}. \tag{A.6}$$

Then,

$$\mathbb{E}\|Z\|_2 \leq \sqrt{2v(Z)\log(d_1+d_2)} + \frac{1}{3}L\log(d_1+d_2). \tag{A.7}$$

The bound (4.34) is derived from this inequality by following similar steps delineated in Section 1.6.3 of [154].

- **Cauchy-Schwarz inequality**: Given two real-valued vectors $\mathbf{u}, \mathbf{v}$, the *Cauchy-Schwarz inequality* states that

$$(\mathbf{u}^\top \mathbf{v})^2 \leq (\mathbf{u}^\top \mathbf{u})(\mathbf{v}^\top \mathbf{v}). \tag{A.8}$$

- **Hicks-Henne bump functions**: First used by Hicks and Henne [76], the *Hicks-Henne bump functions* are a set of sine functions $\{\phi_1, \phi_2, ..., \phi_d\}$ defined by

$$\phi_i(s) = (\sin(\pi s^{m_i}))^{t_i}, \tag{A.9}$$

where t_i, m_i are parameters that can vary for different i that control the position and width of the bumps around the coordinate s defined over the airfoil surface. In the context of this thesis (Section 4.2.3.2), these parameters are fixed in the SU2 deformation code. The deformed geometry is given by the base geometry

plus a linear combination of the bump functions,

$$z = z_{base} + \sum_{i=1}^{d} x_i \phi_i. \tag{A.10}$$

The bump amplitudes x_i are the design parameters in the airfoil test case in Section 4.2.3.2.

- **B-splines**: (Adapted from [124, Section 5.2]) A *B-spline* is defined by a linear combination of basis curves defined on certain intervals. Given $n+1$ control points c_i where $i = 0, 1, ..., n$, the spline can be expressed as

$$S(x) = \sum_{i=0}^{n} c_i B_i^k(x). \tag{A.11}$$

Here, k denotes the degree of the B-spline basis curves. Each basis curve is defined recursively via

$$B_i^0(x) = \begin{cases} 1 & \text{if } t_i \leq x < t_{i+1}, \\ 0 & \text{otherwise,} \end{cases} \tag{A.12}$$

$$B_i^k(x) = \frac{x - t_i}{t_{i+k} - t_i} B_i^{k-1}(x) + \frac{t_{i+k+1} - x}{t_{i+k+1} - t_{i+1}} B_{i+1}^{k-1}(x), \tag{A.13}$$

where t_i is the i-th knot. The quantities c_i, t_i, k can be determined by supplying the coordinates that need to be interpolated.

References

[1] Adcock, B. and Cardenas, J. M. (2020). Near-Optimal Sampling Strategies for Multivariate Function Approximation on General Domains. *SIAM Journal on Mathematics of Data Science*, 2(3):607–630.

[2] Anderson, J. D. (2010). *Fundamentals of Aerodynamics*. McGraw-Hill Education, New York, NY, USA, 5th edition.

[3] Arras, B., Bachmayr, M., and Cohen, A. (2019). Sequential Sampling for Optimal Weighted Least Squares Approximations in Hierarchical Spaces. *SIAM Journal on Mathematics of Data Science*, 1(1):189–207.

[4] Arts, T. and Lambert de Rouvroit, M. (1992). Aero-Thermal Performance of a Two-Dimensional Highly Loaded Transonic Turbine Nozzle Guide Vane: A Test Case for Inviscid and Viscous Flow Computations. *Journal of Turbomachinery*, 114(1):147–154.

[5] Bebendorf, M. (2003). A Note on the Poincaré Inequality for Convex Domains. *Zeitschrift fur Analysis und ihre Anwendung*, 22(4):751–756.

[6] Ben-Ari, E. N. and Steinberg, D. M. (2007). Modeling Data from Computer Experiments: An Empirical Comparison of Kriging with MARS and Projection Pursuit Regression. *Quality Engineering*, 19(4):327–338.

[7] Berkooz, G., Holmes, P., and Lumley, J. L. (1993). The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows. *Annual Review of Fluid Mechanics*, 25(1):539–575.

[8] Bhatnagar, S., Afshar, Y., Pan, S., Duraisamy, K., and Kaushik, S. (2019). Prediction of aerodynamic flow fields using convolutional neural networks. *Computational Mechanics*, 64(2):525–545.

[9] Blatman, G. and Sudret, B. (2011). Adaptive sparse polynomial chaos expansion based on least angle regression. *Journal of Computational Physics*, 230(6):2345–2367.

[10] Bos, L., De Marchi, S., Sommariva, A., and Vianello, M. (2010). Computing Multivariate Fekete and Leja Points by Numerical Linear Algebra. *SIAM Journal on Numerical Analysis*, 48(5):1984–1999.

[11] Boyd, S. P. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, Cambridge, UK.

[12] Bradshaw, S. and Waitz, I. (2009). Impact of Manufacturing Variability on Combustor Liner Durability. *Journal of Engineering for Gas Turbines and Power*, 131(3):032503.

[13] Bungartz, H.-J. and Griebel, M. (2004). Sparse grids. *Acta Numerica*, 13:147–269.

[14] Bura, E. and Cook, R. D. (2001). Estimating the structural dimension of regressions via parametric inverse regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):393–410.

[15] Campolongo, F., Cariboni, J., and Saltelli, A. (2007). An effective screening design for sensitivity analysis of large models. *Environmental Modelling & Software*, 22(10):1509–1518.

[16] Candès, E. and Tao, T. (2005). Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215.

[17] Candès, E. J., Li, X., Ma, Y., and Wright, J. (2011). Robust principal component analysis? *Journal of the ACM*, 58(3):11:1–11:37.

[18] Candès, E. J. and Wakin, M. B. (2008). An Introduction to Compressive Sensing. *IEEE Signal Processing Magazine*, 25(2):21–30.

[19] Chen, S. S., Donoho, D. L., and Saunders, M. A. (1998). Atomic Decomposition by Basis Pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61.

[20] Clark, C. J. (2019). A Step Towards an Intelligent Aerodynamic Design Process. In *ASME Turbo Expo 2019: Turbomachinery Technical Conference and Exposition*. American Society of Mechanical Engineers Digital Collection.

[21] Cockayne, J., Oates, C. J., Ipsen, I. C. F., and Girolami, M. (2019a). A Bayesian Conjugate Gradient Method (with Discussion). *Bayesian Analysis*, 14(3):937–1012.

[22] Cockayne, J., Oates, C. J., Sullivan, T. J., and Girolami, M. (2019b). Bayesian Probabilistic Numerical Methods. *SIAM Review*, 61(4):756–789.

[23] Cohen, A., Davenport, M. A., and Leviatan, D. (2013). On the Stability and Accuracy of Least Squares Approximations. *Foundations of Computational Mathematics*, 13(5):819–834.

[24] Cohen, A. and Migliorati, G. (2017). Optimal weighted least-squares methods. *The SMAI Journal of Computational Mathematics*, 3:181–203.

[25] Conrad, P. R. and Marzouk, Y. M. (2013). Adaptive Smolyak Pseudospectral Approximations. *SIAM Journal on Scientific Computing*, 35(6):A2643–A2670.

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

[27] Constantine, P. G. (2009). *Spectral Methods for Parameterized Matrix Equations*. PhD Thesis, Stanford University, Stanford, California, USA.

[28] Constantine, P. G. (2015). *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies*. Society for Industrial and Applied Mathematics.

[29] Constantine, P. G. and Diaz, P. (2017). Global sensitivity metrics from active subspaces. *Reliability Engineering & System Safety*, 162:1–13.

[30] Constantine, P. G. and Doostan, A. (2017). Time-dependent global sensitivity analysis with active subspaces for a lithium ion battery model. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 10(5):243–262.

[31] Constantine, P. G., Dow, E., and Wang, Q. (2014). Active subspace methods in theory and practice: applications to kriging surfaces. *SIAM Journal on Scientific Computing*, 36(4):A1500–A1524.

[32] Constantine, P. G., Eftekhari, A., Hokanson, J., and Ward, R. A. (2017). A near-stationary subspace for ridge approximation. *Computer Methods in Applied Mechanics and Engineering*, 326:402–421.

[33] Constantine, P. G., Eftekhari, A., and Wakin, M. B. (2015b). Computing active subspaces efficiently with gradient sketching. In *2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 353–356.

[34] Constantine, P. G., Eldred, M. S., and Phipps, E. T. (2012). Sparse pseudospectral approximation method. *Computer Methods in Applied Mechanics and Engineering*, 229-232:1–12.

[35] Constantine, P. G., Zaharatos, B., and Campanelli, M. (2015c). Discovering an active subspace in a single-diode solar cell model. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 8(5-6):264–273.

[36] Cook, R. D. (1998). *Regression Graphics: Ideas for Studying Regressions Through Graphics*. Wiley-Interscience, New York, 1st edition.

[37] Cook, R. D. and Weisberg, S. (1991). Sliced Inverse Regression for Dimension Reduction: Comment. *Journal of the American Statistical Association*, 86(414):328–332.

[38] Cook, R. D. and Weisberg, S. (1994). *An Introduction to Regression Graphics*. Wiley-Inerscience, New York.

[39] Cukier, R. I., Levine, H. B., and Shuler, K. E. (1978). Nonlinear sensitivity analysis of multiparameter model systems. *Journal of Computational Physics*, 26(1):1–42.

[40] del Rosario, Z., Constantine, P., and Iaccarino, G. (2017). Developing Design Insight Through Active Subspaces. In *19th AIAA Non-Deterministic Approaches Conference*, Grapevine, Texas. American Institute of Aeronautics and Astronautics.

[41] del Rosario, Z., Towne, A., and Iaccarino, G. (2018). Dimension Reduction for Shape Design Insight. In *2018 AIAA Aerospace Sciences Meeting*, Kissimmee, Florida. American Institute of Aeronautics and Astronautics.

[42] Dell'Oca, A., Riva, M., and Guadagnini, A. (2017). Moment-based metrics for global sensitivity analysis of hydrological systems. *Hydrology and Earth System Sciences*, 21:6219–6234.

[43] Donoho, D. L. (2006). Compressed sensing. *IEEE Transactions on Information Theory*, 52:1289–1306.

[44] Donoho, D. L. and Elad, M. (2003). Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1 minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202.

[45] Donoho, D. L. and Elad, M. (2006). On the stability of the basis pursuit in the presence of noise. *Signal Processing*, 86(3):511–532.

[46] Dow, E. and Wang, Q. (2013). Output Based Dimensionality Reduction of Geometric Variability in Compressor Blades. In *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, Grapevine (Dallas/Ft. Worth Region), Texas. American Institute of Aeronautics and Astronautics.

[47] Duffner, J. D. (2008). *The Effects of Manufacturing Variability on Turbine Vane Performance*. Master's thesis, Aerospace Computational Design Laboratory, Dept. of Aeronautics & Astronautics, Massachusetts Institute of Technology.

[48] Economon, T. D., Palacios, F., Copeland, S. R., Lukaczyk, T. W., and Alonso, J. J. (2016). SU2: An Open-Source Suite for Multiphysics Simulation and Design. *AIAA Journal*, 54(3):828–846.

[49] Edelman, A., Arias, T., and Smith, S. (1998). The Geometry of Algorithms with Orthogonality Constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353.

[50] Eftekhari, A., Wakin, M. B., Li, P., Constantine, P. G., and Ward, R. A. (2017). Learning the second-moment matrix of a smooth function from point samples. In *2017 51st Asilomar Conference on Signals, Systems, and Computers*, pages 671–675.

[51] Fathi, A. and Alizadeh, M. (2012). Effects of Blade Manufacturing Deviations on Turbine Performance. In *ASME 2012 Gas Turbine India Conference*, pages 203–211, Mumbai, Maharashtra, India. American Society of Mechanical Engineers.

[52] Friedman, J. H. and Stuetzle, W. (1981). Projection Pursuit Regression. *Journal of the American Statistical Association*, 76(376):817–823.

[53] Fukumizu, K., Bach, F. R., and Jordan, M. I. (2009). Kernel dimension reduction in regression. *The Annals of Statistics*, 37(4):1871–1905.

[54] Garzón, V. E. and Darmofal, D. L. (2003). Impact of Geometric Variability on Axial Compressor Performance. *Journal of Turbomachinery*, 125(4):692–703.

[55] Gautschi, W. (2004). *Orthogonal Polynomials: Computation and Approximation*. Numerical Mathematics and Scientific Computation. Oxford University Press, Oxford, New York.

[56] Geraci, G., Congedo, P. M., Abgrall, R., and Iaccarino, G. (2016). High-order statistics in global sensitivity analysis: Decomposition and model reduction. *Computer Methods in Applied Mechanics and Engineering*, 301:80–115.

[57] Glaws, A. and Constantine, P. G. (2019). Gauss–Christoffel quadrature for inverse regression: applications to computer experiments. *Statistics and Computing*, 29(3):429–447.

[58] Glaws, A., Constantine, P. G., and Cook, R. D. (2020). Inverse regression for ridge recovery: a data-driven approach for parameter reduction in computer experiments. *Statistics and Computing*, 30(2):237–253.

[59] Golub, G. and Pereyra, V. (2003). Separable nonlinear least squares: the variable projection method and its applications. *Inverse Problems*, 19(2):R1–R26.

[60] Golub, G. H. and van Loan, C. F. (2013). *Matrix Computations*. Johns Hopkins University Press, Baltimore, 4th edition.

[61] Golub, G. H. and Welsch, J. H. (1969). Calculation of Gauss Quadrature Rules. *Mathematics of Computation*, 23(106):221–s10.

[62] Goodhand, M. (2011). *Compressor Leading Edges*. PhD Thesis, University of Cambridge, Cambridge, UK.

[63] Goodhand, M. N. and Miller, R. J. (2012). The Impact of Real Geometries on Three-Dimensional Separations in Compressors. *Journal of Turbomachinery*, 134(2):021007.

[64] Goodhand, M. N., Miller, R. J., and Lung, H. W. (2014). The Impact of Geometric Variation on Compressor Two-Dimensional Incidence Range. *Journal of Turbomachinery*, 137(2):021007.

[65] Grey, Z. J. and Constantine, P. G. (2018). Active Subspaces of Airfoil Shape Parameterizations. *AIAA Journal*, 56(5):2003–2017.

[66] Gross, J. C. (2021). *Derivative-Free Methods for High-Dimensional Optimization with Application to Centrifugal Pump Design*. PhD Thesis, University of Cambridge, Cambridge, UK.

[67] Gross, J. C., Seshadri, P., and Parks, G. (2020). Optimisation with Intrinsic Dimension Reduction: A Ridge Informed Trust-Region Method. In *AIAA Scitech 2020 Forum*. American Institute of Aeronautics and Astronautics.

[68] Guo, L., Narayan, A., Yan, L., and Zhou, T. (2018a). Weighted Approximate Fekete Points: Sampling for Least-Squares Polynomial Approximation. *SIAM Journal on Scientific Computing*, 40(1):A366–A387.

[69] Guo, L., Narayan, A., and Zhou, T. (2018b). A gradient enhanced ℓ_1-minimization for sparse approximation of polynomial chaos expansions. *Journal of Computational Physics*, 367:49–64.

[70] Guo, L., Narayan, A., and Zhou, T. (2020). Constructing Least-Squares Polynomial Approximations. *SIAM Review*, 62(2):483–508.

[71] Guo, X., Li, W., and Iorio, F. (2016). Convolutional Neural Networks for Steady Flow Approximation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 481–490, New York, NY, USA. Association for Computing Machinery.

[72] Hadigol, M. and Doostan, A. (2018). Least squares polynomial chaos expansion: A review of sampling strategies. *Computer Methods in Applied Mechanics and Engineering*, 332:382–407.

[73] Hampton, J. and Doostan, A. (2015a). Coherence motivated sampling and convergence analysis of least squares polynomial Chaos regression. *Computer Methods in Applied Mechanics and Engineering*, 290:73–97.

[74] Hampton, J. and Doostan, A. (2015b). Compressive sampling of polynomial chaos expansions: Convergence analysis and sampling strategies. *Journal of Computational Physics*, 280:363–386.

[75] Herman, J. and Usher, W. (2017). SALib: An open-source Python library for Sensitivity Analysis. *The Journal of Open Source Software*, 2(9):97.

[76] Hicks, R. M. and Henne, P. A. (1978). Wing design by numerical optimization. *Journal of Aircraft*, 15(7):407–412.

[77] Hokanson, J. M. and Constantine, P. G. (2018). Data-Driven Polynomial Ridge Approximation Using Variable Projection. *SIAM Journal on Scientific Computing*, 40(3):A1566–A1589.

[78] Homma, T. and Saltelli, A. (1996). Importance measures in global sensitivity analysis of nonlinear models. *Reliability Engineering & System Safety*, 52(1):1–17.

[79] Hooker, G. (2004). Discovering additive structure in black box functions. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD'04, pages 575–580, Seattle, WA, USA. Association for Computing Machinery.

[80] Jakeman, J. D., Franzelin, F., Narayan, A., Eldred, M., and Plfüger, D. (2019). Polynomial chaos expansions for dependent random variables. *Computer Methods in Applied Mechanics and Engineering*, 351:643–666.

[81] Jakeman, J. D. and Narayan, A. (2018). Generation and application of multivariate polynomial quadrature rules. *Computer Methods in Applied Mechanics and Engineering*, 338:134–161.

[82] Jakeman, J. D., Narayan, A., and Zhou, T. (2017). A Generalized Sampling and Preconditioning Scheme for Sparse Approximation of Polynomial Chaos Expansions. *SIAM Journal on Scientific Computing*, 39(3):A1114–A1144.

[83] Ji, W., Wang, J., Zahm, O., Marzouk, Y. M., Yang, B., Ren, Z., and Law, C. K. (2018). Shared low-dimensional subspaces for propagating kinetic uncertainty to multiple outputs. *Combustion and Flame*, 190:146–157.

[84] Jofre, L., del Rosario, Z. R., and Iaccarino, G. (2020). Data-driven dimensional analysis of heat transfer in irradiated particle-laden turbulent flow. *International Journal of Multiphase Flow*, 125:103198.

[85] Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer Series in Statistics. Springer-Verlag, New York, 2nd edition.

[86] Joshi, S. and Boyd, S. (2009). Sensor Selection via Convex Optimization. *IEEE Transactions on Signal Processing*, 57(2):451–462.

[87] Kamenik, J., Voutchkov, I., Toal, D. J. J., Keane, A. J., Högner, L., Meyer, M., and Bates, R. (2018). Robust Turbine Blade Optimization in the Face of Real Geometric Variations. *Journal of Propulsion and Power*, 34(6):1479–1493.

[88] Kenett, R., Zacks, S., and Amberti, D. (2013). *Modern Industrial Statistics: with applications in R, MINITAB and JMP*. John Wiley & Sons.

[89] Koehler, J. R. and Owen, A. B. (1996). Computer experiments. In *Handbook of Statistics*, volume 13 of *Design and Analysis of Experiments*, pages 261–308. Elsevier.

[90] Kontolati, K., Loukrezis, D., Santos, K. R. M. d., Giovanis, D. G., and Shields, M. D. (2021). Manifold learning-based polynomial chaos expansions for high-dimensional surrogate models. *arXiv:2107.09814 [physics]*.

[91] Kucherenko, S. and Sobol', I. M. (2009). Derivative based global sensitivity measures and their link with global sensitivity indices. *Mathematics and Computers in Simulation*, 79(10):3009–3017.

[92] Lange, A., Voigt, M., Vogeler, K., and Johann, E. (2012). Principal component analysis on 3D scanned compressor blades for probabilistic CFD simulation. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, Honolulu, Hawaii. American Institute of Aeronautics and Astronautics.

[93] Lee, M. R. (2017). *Prediction and Dimension Reduction Methods in Computer Experiments*. PhD Thesis, Stanford University, Stanford, California, USA.

[94] Lee, M. R. (2019). Modified Active Subspaces Using the Average of Gradients. *SIAM/ASA Journal on Uncertainty Quantification*, 7(1):53–66.

[95] Li, B. (2018). *Sufficient Dimension Reduction: Methods and Applications with R*. Chapman and Hall/CRC, Boca Raton, 1st edition.

[96] Li, B. and Wang, S. (2007). On Directional Regression for Dimension Reduction. *Journal of the American Statistical Association*, 102(479):997–1008.

[97] Li, B., Zha, H., and Chiaromonte, F. (2005). Contour Regression: A General Approach to Dimension Reduction. *The Annals of Statistics*, 33(4):1580–1616.

[98] Li, G., Rabitz, H., Yelvington, P. E., Oluwole, O. O., Bacon, F., Kolb, C. E., and Schoendorf, J. (2010). Global Sensitivity Analysis for Systems with Independent and/or Correlated Inputs. *The Journal of Physical Chemistry A*, 114(19):6022–6032.

[99] Li, K.-C. (1991). Sliced Inverse Regression for Dimension Reduction. *Journal of the American Statistical Association*, 86(414):316–327.

[100] Li, K.-C. and Duan, N. (1989). Regression Analysis Under Link Violation. *The Annals of Statistics*, 17(3):1009–1052.

[101] Liu, P.-L. and Der Kiureghian, A. (1986). Multivariate distribution models with prescribed marginals and covariances. *Probabilistic Engineering Mechanics*, 1(2):105–112.

[102] Liu, R. and Owen, A. B. (2006). Estimating Mean Dimensionality of Analysis of Variance Decompositions. *Journal of the American Statistical Association*, 101(474):712–721.

[103] Lobato, H. M. P., Gower, I. R., Powell, C. A., and Orchard, N. B. (2014). United States Patent: 8718975 - Surface profile evaluation.

[104] Logan, B. F. and Shepp, L. A. (1975). Optimal reconstruction of a function from its projections. *Duke Mathematical Journal*, 42(4):645–659.

[105] Lu, L., Meng, X., Mao, Z., and Karniadakis, G. E. (2021). DeepXDE: A Deep Learning Library for Solving Differential Equations. *SIAM Review*, 63(1):208–228.

[106] Lukaczyk, T. W., Constantine, P., Palacios, F., and Alonso, J. J. (2014). Active Subspaces for Shape Optimization. In *10th AIAA Multidisciplinary Design Optimization Conference*, National Harbor, Maryland. American Institute of Aeronautics and Astronautics.

[107] Ma, Y. and Zhu, L. (2012). A Semiparametric Approach to Dimension Reduction. *Journal of the American Statistical Association*, 107(497):168–179.

[108] Ma, Y. and Zhu, L. (2013). A Review on Dimension Reduction. *International Statistical Review*, 81(1):134–150.

[109] Masanet, E., Shehabi, A., Lei, N., Smith, S., and Koomey, J. (2020). Recalibrating global data center energy-use estimates. *Science*, 367(6481):984–986.

[110] Montanelli, H., Montagnac, M., and Gallard, F. (2015). Gradient Span Analysis Method: Application to the Multipoint Aerodynamic Shape Optimization of a Turbine Cascade. *Journal of Turbomachinery*, 137(9):091006.

[111] Morris, M. D. (1991). Factorial Sampling Plans for Preliminary Computational Experiments. *Technometrics*, 33(2):161–174.

[112] Morris, M. D., Mitchell, T. J., and Ylvisaker, D. (1993). Bayesian Design and Analysis of Computer Experiments: Use of Derivatives in Surface Prediction. *Technometrics*, 35(3):243–255.

[113] Narayan, A., Jakeman, J. D., and Zhou, T. (2016). A Christoffel function weighted least squares algorithm for collocation approximations. *Mathematics of Computation*, 86(306):1913–1947.

[114] Natarajan, B. K. (1995). Sparse Approximate Solutions to Linear Systems. *SIAM Journal on Computing*, 24(2):227–234.

[115] Navarro, M., Witteveen, J., and Blom, J. (2014). Polynomial Chaos Expansion for general multivariate distributions with correlated variables. *arXiv:1406.5483 [math]*.

[116] O'Hagan, A. (1991). Bayes–Hermite quadrature. *Journal of Statistical Planning and Inference*, 29(3):245–260.

[117] Osborne, M., Garnett, R., Ghahramani, Z., Duvenaud, D. K., Roberts, S. J., and Rasmussen, C. (2012). Active Learning of Model Evidence Using Bayesian Quadrature. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

[118] Owen, A. B., Dick, J., and Chen, S. (2014). Higher order Sobol' indices. *Information and Inference: A Journal of the IMA*, 3(1):59–81.

[119] Park, H.-S. and Jun, C.-H. (2009). A simple and fast algorithm for K-medoids clustering. *Expert Systems with Applications*, 36(2, Part 2):3336–3341.

[120] Peherstorfer, B., Willcox, K., and Gunzburger, M. (2018). Survey of Multifidelity Methods in Uncertainty Propagation, Inference, and Optimization. *SIAM Review*, 60(3):550–591.

[121] Peng, J., Hampton, J., and Doostan, A. (2014). A weighted ℓ_1-minimization approach for sparse polynomial chaos expansions. *Journal of Computational Physics*, 267:92–111.

[122] Peng, J., Hampton, J., and Doostan, A. (2016). On polynomial chaos expansion via gradient-enhanced ℓ_1-minimization. *Journal of Computational Physics*, 310:440–458.

[123] Pinkus, A. (2015). *Ridge Functions*. Cambridge University Press, Cambridge, 1st edition.

[124] Prautzsch, H., Boehm, W., and Paluszny, M. (2002). *Bézier and B-Spline Techniques*. Mathematics and Visualization. Springer, Berlin, Heidelberg.

[125] Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707.

[126] Reagan, M. T., Najm, H. N., Ghanem, R. G., and Knio, O. M. (2003). Uncertainty quantification in reacting-flow simulations through non-intrusive spectral projection. *Combustion and Flame*, 132(3):545–555.

[127] Russi, T. M. (2010). *Uncertainty Quantification with Experimental Data and Complex System Models*. PhD Thesis, University of California, Berkeley, Berkeley, California, USA.

[128] Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., and Tarantola, S. (2008). *Global Sensitivity Analysis: The Primer*. John Wiley & Sons.

[129] Samarov, A. M. (1993). Exploring Regression Structure Using Nonparametric Functional Estimation. *Journal of the American Statistical Association*, 88(423):836–847.

[130] Schölkopf, B., Smola, A., and Müller, K.-R. (1997). Kernel principal component analysis. In Gerstner, W., Germond, A., Hasler, M., and Nicoud, J.-D., editors, *Artificial Neural Networks — ICANN'97*, Lecture Notes in Computer Science, pages 583–588, Berlin, Heidelberg. Springer.

[131] Scillitoe, A., Seshadri, P., Wong, C. Y., and Duncan, A. (2021). Polynomial ridge flowfield estimation. *Physics of Fluids*, 33(12):127110.

[132] Scillitoe, A. D. (2017). *Towards Predictive Eddy Resolving Simulations for Gas Turbine Compressors*. PhD Thesis, University of Cambridge, Cambridge, UK.

[133] Seshadri, P., Iaccarino, G., and Ghisu, T. (2019a). Quadrature Strategies for Constructing Polynomial Approximations. In *Uncertainty Modeling for Engineering Applications*, pages 1–25. Springer.

[134] Seshadri, P., Narayan, A., and Mahadevan, S. (2017). Effectively Subsampled Quadratures for Least Squares Polynomial Approximations. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):1003–1023.

[135] Seshadri, P., Shahpar, S., Constantine, P., Parks, G., and Adams, M. (2018). Turbomachinery Active Subspace Performance Maps. *Journal of Turbomachinery*, 140(4):041003.

[136] Seshadri, P., Shahpar, S., and Parks, G. T. (2014). Robust Compressor Blades for Desensitizing Operational Tip Clearance Variations. In *ASME Turbo Expo 2014: Turbine Technical Conference and Exposition*. American Society of Mechanical Engineers Digital Collection.

[137] Seshadri, P., Yuchi, S., Parks, G., and Shahpar, S. (2020). Supporting multi-point fan design with dimension reduction. *The Aeronautical Journal*, 124(1279):1371–1398.

[138] Seshadri, P., Yuchi, S., and Parks, G. T. (2019b). Dimension Reduction via Gaussian Ridge Functions. *SIAM/ASA Journal on Uncertainty Quantification*, 7(4):1301–1322.

[139] Shi, W., Chen, P., Li, X., Ren, J., and Jiang, H. (2019). Uncertainty Quantification of the Effects of Small Manufacturing Deviations on Film Cooling: A Fan-Shaped Hole. *Aerospace*, 6(4):46.

[140] Sirovich, L. (1987). Turbulence and the Dynamics of Coherent Structures Part I: Coherent Structures. *Quarterly of Applied Mathematics*, 45(3):561–571.

[141] Smith, R. L. (1984). Efficient Monte Carlo Procedures for Generating Points Uniformly Distributed Over Bounded Regions. *Operations Research*, 32(6):1296–1308.

[142] Smith, R. L. (1996). The Hit-and-run Sampler: A Globally Reaching Markov Chain Sampler for Generating Arbitrary Multivariate Distributions. In *Proceedings of the 28th Winter Conference on Simulation*, WSC '96, pages 260–264, Washington, DC, USA. IEEE Computer Society.

[143] Sobol', I. M. (1993). Sensitivity estimates for nonlinear mathematical models. *Mathematical Modelling and Computational Experiments*, 1(4):407–414.

[144] Sobol', I. M. (2001). Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and Computers in Simulation*, 55(1):271–280.

[145] Sudret, B. (2008). Global sensitivity analysis using polynomial chaos expansions. *Reliability Engineering & System Safety*, 93(7):964–979.

[146] Sudret, B., Berveiller, M., and Lemaire, M. (2006). A stochastic finite element procedure for moment and reliability analysis. *European Journal of Computational Mechanics*, 15(7-8):825–866.

[147] Tang, G. and Iaccarino, G. (2014). Subsampled Gauss Quadrature Nodes for Estimating Polynomial Chaos Expansions. *SIAM/ASA Journal on Uncertainty Quantification*, 2(1):423–443.

[148] Thuerey, N., Weißenow, K., Prantl, L., and Hu, X. (2020). Deep Learning Methods for Reynolds-Averaged Navier–Stokes Simulations of Airfoil Flows. *AIAA Journal*, 58(1):25–36.

[149] Torreguitart, S., Verstraete, T., and Mueller, L. (2018). Optimization of the LS89 Axial Turbine Profile Using a CAD and Adjoint Based Approach. *International Journal of Turbomachinery, Propulsion and Power*, 3(3):20.

[150] Trefethen, L. N. (2008). Is Gauss Quadrature Better than Clenshaw–Curtis? *SIAM Review*, 50(1):67–87.

[151] Trefethen, L. N. (2013). *Approximation Theory and Approximation Practice*. Society for Industrial and Applied Mathematics.

[152] Trefethen, L. N. (2017). Cubature, Approximation, and Isotropy in the Hypercube. *SIAM Review*, 59(3):469–491.

[153] Tropp, J. (2004). Greed is good: algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10):2231–2242.

[154] Tropp, J. A. (2015). An introduction to matrix concentration inequalities. *Foundations and Trends in Machine Learning*, 8(1-2):1–230.

[155] Vitale, S., Gori, G., Pini, M., Guardone, A., Economon, T. D., Palacios, F., Alonso, J. J., and Colonna, P. (2015). Extension of the SU2 Open Source CFD Code to the Simulation of Turbulent Flows of Fluids Modelled with Complex Thermophysical Laws. In *22nd AIAA Computational Fluid Dynamics Conference*, Dallas, Texas. American Institute of Aeronautics and Astronautics.

[156] Wang, S. C. (2001). Quantifying passive and driven large-scale evolutionary trends. *Evolution*, 55(5):849–858.

[157] Wang, X. and Zou, Z. (2019). Uncertainty analysis of impact of geometric variations on turbine blade performance. *Energy*, 176:67–80.

[158] Witteveen, J. A. and Bijl, H. (2006). Modeling Arbitrary Uncertainties Using Gram-Schmidt Polynomial Chaos. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada. American Institute of Aeronautics and Astronautics.

[159] Xia, Y., Tong, H., Li, W. K., and Zhu, L.-X. (2002). An adaptive estimation of dimension reduction space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(3):363–410.

[160] Xiu, D. and Karniadakis, G. E. (2002). The Wiener–Askey Polynomial Chaos for Stochastic Differential Equations. *SIAM Journal on Scientific Computing*, 24(2):619–644.

[161] Xiu, D. and Karniadakis, G. E. (2003). A new stochastic approach to transient heat conduction modeling with uncertainty. *International Journal of Heat and Mass Transfer*, 46(24):4681–4693.

[162] Yin, X. and Cook, R. D. (2005). Direction estimation in single-index regressions. *Biometrika*, 92(2):371–384.

[163] Zahm, O., Constantine, P. G., Prieur, C., and Marzouk, Y. M. (2020). Gradient-Based Dimension Reduction of Multivariate Vector-Valued Functions. *SIAM Journal on Scientific Computing*, 42(1):A534–A558.