Numerical Methods for the Quantification of Uncertainty in Discontinuous Functions of High Dimension

John Davis Jakeman

February 2011

A thesis submitted for the degree of Doctor of Philosophy of the Australian National University
This thesis is dedicated to my family. My mother Jane, my father Tony, and my two brothers Mike and Rowan. My life revolves around you. I love you all so very much.
Declaration

Some of the work in this thesis was carried out in collaboration with Dongbin Xiu, Mike Eldred and Richard Archibald. In particular Chapter 3 is based on results established jointly with Dongbin Xiu and Mike Eldred, and Chapter 5 is based on joint work with Richard Archibald and Dongbin Xiu.

The publication details of the thesis are Jakeman and Roberts [61], Jakeman, Eldred and Xiu [63] and Jakeman, Archibald and Xiu [62]. Often the text of these papers has been closely followed.

Elsewhere in the thesis, unless another source is acknowledged, the work described is my own.

John Jakeman
Acknowledgements

The work presented in this thesis was undertaken at the Department of Mathematics, The Australian National University, with financial assistance from the CSIRO Water for a Healthy Country Flagship.

I would first like to thank the three members of my advisory panel, Stephen Roberts, Barry Croke and Ian Webster for their guidance and valuable suggestions. I would particularly like to thank Stephen Roberts for his unerring support. His suggestions of research topics and avenues for improvement were stimulating and highly appreciated. Special thanks must also go to Dongbin Xiu. His enthusiasm was infectious and his advice and experience invaluable. I would also like to thank Rick Archibald, Mike Eldred Jaideep Ray and again Dongbin Xiu who have contributed to the development of the work contained in this thesis. Collaboration between Mike, Dongbin and myself formed the basis of the framework presented in Chapter 3 and the discussions between Rick, Dongbin and myself led to the development of the work contained in Chapter 5.
Abstract

This thesis deals with efficient numerical methods for quantifying uncertainty in mathematical models with uncertain input data. Input uncertainty in the governing equations may assume two forms: aleatory uncertainty which can be characterised by known probability distributions; and epistemic uncertainty which arises from a lack of knowledge of probabilistic information.

The first contribution of this thesis is to present a numerical framework for quantification of epistemic uncertainty. The proposed methodology does not require any probabilistic information on uncertain input parameters. The method only necessitates an estimate of the range of the uncertain variables that encapsulates the true range of the input variables with overwhelming probability. To quantify the epistemic uncertainty, we solve an encapsulation problem, which is a solution to the original governing equations defined on the estimated range of the input variables. In the case where probability distributions of the epistemic variables become known a posteriori, this information can be used to post-process the solution and evaluate solution statistics. Convergence results are established for such cases, along with strategies for dealing with mixed aleatory and epistemic uncertainty.

Stochastic collocation (SC) methods are one way of constructing the polynomial approximation necessary to solve the encapsulation problem. The second contribution of this thesis is the development of an $h$-adaptive generalised sparse grid ($h$-GSG) collocation method for quantifying input uncertainty in high-dimensional functions with discontinuities. The proposed algorithm combines the strengths of the generalised sparse grid algorithm and hierarchical surplus guided $h$-adaptivity. A high-degree basis is used to obtain a high-order method which, given sufficient smoothness, performs significantly better than the piecewise-linear basis. The underlying generalised sparse grid algorithm greedily selects the dimensions and variable interactions that contribute most to the variability of a function. The hierarchical surplus of points within the sparse grid is used as an
error criterion for $h$-refinement with the aim of concentrating computational effort within rapidly varying or discontinuous regions. This approach limits the number of points that are invested in 'unimportant' dimensions and regions within the high-dimensional domain. We show the utility of the $h$-GSG method for non-smooth functions with as many as 700 variables.

Although the $h$-GSG method is an improvement on existing sparse grid SC methods, it is likely that higher rates of convergence could be obtained by decomposing the input domain into regions of high-regularity on which independent global polynomial methods can be applied. With this goal we present a method for identifying and locating discontinuities in high-dimensional spaces. The proposed discontinuity detection method is a locally adaptive approach which uses polynomial annihilation to determine the existence of discontinuities and to guide refinement. The adaptive procedure is constructed to ensure that refinement occurs only in those dimensions deemed to contribute to the location of the discontinuity. When the dimensionality of the discontinuity resides in a small subset of dimensions of the input space the proposed method is very efficient. In these situations the algorithm becomes "optimal," that is the total number of points required grows linearly with the dimensionality.
Contents

1 Introduction ........................................................................................................ 1
   1.1 Defining Uncertainty .................................................................................. 1
       1.1.1 Nature ..................................................................................... 2
       1.1.2 Level ..................................................................................... 2
       1.1.3 Location ............................................................................... 2
   1.2 Quantifying Uncertainty ............................................................................ 3
       1.2.1 The Need for Increased Efficiency ........................................... 5
       1.2.2 Sensitivity Analysis .................................................................. 8
       1.2.3 Response Surface Methods ..................................................... 9
   1.3 Thesis Objectives and Outline ................................................................. 15

2 Stochastic Collocation ..................................................................................... 19
   2.1 Formulation ......................................................................................... 19
   2.2 Representing the Random Input Data ..................................................... 20
       2.2.1 Parametric Uncertainty ......................................................... 21
       2.2.2 Karhunen–Loève Expansion ................................................ 21
       2.2.3 Alternative Decompositions ............................................... 22
   2.3 Collocation Method ............................................................................... 23
       2.3.1 Specifying the Test Space ....................................................... 24
       2.3.2 Constructing the Trial Space ................................................. 24
   2.4 Conclusions ......................................................................................... 25

3 Quantifying Epistemic Uncertainty ................................................................ 27
   3.1 Problem Setup ................................................................................... 30
   3.2 Methodology ....................................................................................... 31
       3.2.1 Range Estimation ................................................................. 31
       3.2.2 Encapsulation Problem ......................................................... 32
       3.2.3 Solution for the Encapsulation Problem .................................. 33
### CONTENTS

3.2.4 Epistemic Uncertainty Analysis ........................................... 38  
3.2.5 Mixed Aleatory–Epistemic Uncertainty Analysis ............... 40  
3.3 Numerical Examples ............................................................. 42  
3.3.1 Ordinary Differential Equation ........................................... 42  
3.3.2 Random Oscillator ................................................................. 45  
3.3.3 Homogeneous Diffusion Equation ........................................ 49  
3.4 Conclusions ............................................................................. 52

4  
 **h-Adaptive Sparse Grids** .................................................. 55  
4.1 Sub-Space Splitting ................................................................. 56  
4.2 A Local High-Order Basis .......................................................... 59  
4.2.1 Construction .......................................................................... 61  
4.2.2 Efficient Evaluation of the Hierarchical Surplus ............... 63  
4.2.3 Quadrature ............................................................................. 69  
4.3 Adaptivity .................................................................................... 69  
4.3.1 Generalised Sparse Grid (GSG) Algorithm ......................... 71  
4.3.2 Extending the GSG algorithm ............................................. 75  
4.4 Error Analysis .......................................................................... 78  
4.5 Numerical Study ...................................................................... 84  
4.5.1 2D Examples ......................................................................... 85  
4.5.2 Increasing the Degree of the Local Polynomial Basis .......... 86  
4.5.3 Efficient Termination of h-GSG .......................................... 88  
4.5.4 Comparison with other Methods ........................................... 90  
4.5.5 High-Dimensional Interpolation ........................................... 106  
4.6 Multi-dimensional Stochastic Application .................................... 111  
4.6.1 Model Formulation ............................................................... 112  
4.6.2 Results ................................................................................ 115  
4.7 Conclusions .............................................................................. 117

5  
 **Discontinuity Detection** ....................................................... 121  
5.1 Introduction ........................................................................... 121  
5.2 Polynomial Annihilation Edge Detection ............................... 122  
5.3 Edge Tracking .......................................................................... 125  
5.3.1 Edge Tracking Algorithm .................................................... 126  
5.4 High-Dimensional Discontinuity Detection ............................. 130  
5.4.1 Adaptive Sparse Grids ........................................................... 132  
5.4.2 Sparse Grid Discontinuity Detection ................................... 133
## CONTENTS

5.5 High-Dimensional Edge Tracking .................................................. 136
5.6 Post-Processing ............................................................................. 141
  5.6.1 Estimating Distance Functions .............................................. 141
  5.6.2 Domain Classification ............................................................. 143
5.7 Numerical Results .......................................................................... 146
  5.7.1 Benchmark Problems ............................................................... 146
  5.7.2 Multidimensional Stochastic Application ............................... 150
5.8 Conclusions .................................................................................... 151

6 Conclusions ....................................................................................... 153

Bibliography ........................................................................................... 158
Chapter 1

Introduction

1.1 Defining Uncertainty

Knowledge of the sources and effects of uncertainty is needed for effective decision making, policy and planning [84]. In practical applications, the uncertainties identified are typically only the uncertainties that can be quantified, thereby underestimating uncertainty. However, in order to determine the utility of models, it is important to understand all sources of uncertainties associated with the model [90].

Numerous definitions and perceptions of uncertainty exist. Some authors use the term uncertainty to describe the level of doubt in a prediction, whilst other authors include numerical errors, natural stochastic variation and even ignorance in their definition of uncertainty. Walker et al. [116] define uncertainty as 'any departure from the unachievable ideal of complete determinism' and Klauer and Brown [64] refer to uncertainty as the 'degree of confidence a person has about the specific outcome of an event of action.' Chang et al. [21] and Harremoës and Madsen [51] believe uncertainty to be inherent in all physical systems and cannot be eliminated.

No single definition of uncertainty is correct. In this section we adopt the definition of Klauer and Brown [64] and utilise the framework provided by Walker et al. [116] and extended by Warmink et al. [123] to facilitate structured identification and classification of uncertainties, that arise during model development and application.

Adopting the aforementioned framework we classify uncertainty based upon three criteria: the nature of the uncertainty; the level of uncertainty, which is where the uncertainty arises along the span of knowledge between total ignorance
and complete determinism; and the location at which the uncertainty arises in the modelling process. Each of these dimensions of uncertainty can be further decomposed into classes which are discussed below and visualised in Figure 1.1.

1.1.1 Nature

Warmink et al [123] define three natures of uncertainty: natural variability; epistemic uncertainty; and ambiguity. Natural variability arises from the inherent variation associated with the system under consideration and is irreducible. Epistemic uncertainty represents any lack of knowledge or information in any phase or activity of the modelling process [58] and is reducible through the introduction of additional information. Ambiguity refers to the simultaneous presence of multiple equally valid, yet differing, frames of knowledge [116]. The choice of which knowledge framework to use is often contestable and determined through expert opinion. However unlike epistemic uncertainty, in the presence of ambiguity the collection of more knowledge does not necessarily result in convergence to a single truth.

1.1.2 Level

Once the nature of the uncertainty has been identified the uncertainty can be classified into one of four levels: statistical; scenario; qualitative; and ignorance. Statistical uncertainty refers to uncertainty that can be quantified by probabilistic information or other numerical properties such as ranges. Scenario uncertainty arises when a range of outcomes are possible, but the mechanisms leading to those outcomes prevents the assignment of probabilities or weights to each outcome. Qualitative uncertainty represents uncertainty which cannot be expressed in terms of numerically measurable values. Sources of such uncertainty include expert opinion and values of decision makers. Finally, ignorance refers to uncertainty in the mechanisms and relations between the systems and components being studied. In these situations it is impossible to provide any quantitative or qualitative description of the uncertainty. Identification and documentation of ignorance is necessary to improve the transparency of the modelling process.

1.1.3 Location

Uncertainty is introduced through all stages of the modelling process. These uncertainties arise in four locations: context; model structure; input; and param-
1.2. QUANTIFYING UNCERTAINTY

eters.

Context refers to the bounds of a model's intended purpose. During the process of model development numerous choices are made on what physical and/or other processes to include in the model formulation, the existence and nature of any stochastic variability and so on. Models validated in one region of the input space may be a poor representation in other regions. Extrapolation errors are case-dependent and extremely difficult to estimate, but one must be aware that such uncertainty exists and limit model application to its intended and tested use.

Model structure refers to the mathematical expressions chosen to describe the relationships between system variables and components. The same processes can be modelled using vastly different modelling methodologies, such as empirical, statistical, physical or hybrids of these approaches. Once a model structure has been chosen it must then be implemented. For example a finite element method may be used to solve a system of partial differential equations. The numerical method used to implement a given model and the type and level of discretization are additional sources of model structure uncertainty.

Input is defined as all data that are used to describe geometry and time or spatially dependent forces. Uncertainty in input is caused by uncertainties in measurements, and uncertainties in outputs of other models that are used as inputs.

Parameters differ from input in that they refer only to a priori determined values which do not change during model simulation. Parameters are invariant for any particular model context, structure and numerical implementation and may, but do not necessarily, have a physical or other theoretical foundation. Input and parametric uncertainty are closely related. Here and throughout this manuscript we will considered these two locations of uncertainty together. In Figure 1.1 and throughout this thesis we use inputs and parameters interchangeably.

1.2 Quantifying Uncertainty

Once model uncertainties have been identified and classified, the effects of such uncertainty should be traced through the system thoroughly enough to allow one to evaluate their effects on the intended use of the model, that often being the prediction of model outputs. To date, most uncertainty analysis has focused on quantifying statistical input uncertainty arising from natural variability, com-
Figure 1.1: A decision tree to identify the nature, level and location of the uncertainties in any model. This tree is based upon the classification scheme of Walker et al [116] and a modification of the tree found constructed by Warmink et al [123]. The grey boxes depict the model uncertainties considered in this manuscript.
1.2. QUANTIFYING UNCERTAINTY

monly referred to as aleatory uncertainty. Recently some attention has also been
given to quantifying epistemic statistical input uncertainty. Quantification of
other forms (nature, level and location) of uncertainty is much more challenging,
due to their non-numerical character. Consequently we restrict our attention to
Uncertainty Quantification (UQ) studies of functions of the form

\[ u = M(\xi) \]

where \( M \) represents the model under consideration, \( u = (u_1, \ldots, u_m) \) is a \( m \)-
dimensional vector of model outputs, and \( \xi = (\xi_1, \ldots, \xi_d) \) is a \( d \)-dimensional
vector of variables which describe the uncertainty in the model inputs. We will
assume that some statistical information about these variables, such as ranges or
probability densities, is available. The sources of uncertainty considered in this
manuscript are highlighted in grey in Figure 1.1.

1.2.1 The Need for Increased Efficiency

The models \( M \) can be highly complex, non-linear and often discontinuous. In
these situations computational methods and software tools are needed to facilitate
analysis of model uncertainty. The most appropriate method and tool is depen­
dent on the goals of the uncertainty quantification analysis. Four common aims of
UQ include: the construction of probability density estimates such as probability
density functions (PDFs) and cumulative distribution functions (CDFs) of the
model observables; the calculation of moments, such as mean, variance, skewness
and kurtosis of the model observables; the estimation of the likely distributions
of the input variables given some observational data [6, 60, 120]; and determining
model reliability, that is the probability that a response metric is below a given
level [7, 15, 27, 48, 52, 95, 103].

Monte Carlo (MC) simulation is frequently used to estimate moments, densi­
ties and reliability. The method is extremely robust, is easy to implement, and
provides results that are easy to interpret. Furthermore the method does not
require any manipulation of the deterministic model, and unlike many determin­
istic methods has no requirement on the smoothness of the response surface. The
major disadvantage of Monte Carlo methods is that sampling is infeasible when
model simulations require large amounts of computational time. The error in the
approximation decreases slowly with the number \( N \) of samples used. Specifically
the error \( \epsilon \) and the number of samples \( N \) associated with the Monte Carlo method
are related by

$$\epsilon = O(\sigma N^{-1/2}), \quad N = O(\sigma/\epsilon)^2$$

The efficiency of MC can be improved with the use of variance reduction techniques, such as Latin Hypercube sampling (LHS) [81] and importance sampling [44, 50]. These methods attempt to enhance the performance of the MC method by reducing the variance $\sigma^2$ of the samples generated using knowledge of the integrand and/or the distribution of the input variables. The construction of samples using variance reduction techniques requires extra computational time but in most cases the savings due to variance reduction are quite significant. Decreasing $\sigma$ by a factor of $c$ will reduce the number of samples by a factor of $c^2$.

Note that if one wishes to simply approximate statistical moments the Quasi-Monte Carlo (QMC) methods [29, 49, 57, 87, 91, 104, 108] provide an efficient alternative to MC sampling. Unlike traditional MC which employs randomly generated point sets, QMC utilises correlated sequences that are designed to eliminate clumping and to provide a more uniform coverage of the input space and thus faster convergence of quadrature formulas. The resulting convergence rate is $O(N^{-1} \log(N)^d)$ compared to the slower convergence rate of $N^{-1/2}$ using traditional MC sampling [18].

Even with the improvements provided by variance reduction techniques and QMC methods, often the number of model evaluations required by these methods remain infeasible, especially if high levels of accuracy are needed. Sensitivity analysis and response surface methods have been developed to overcome this limitation.

Traditionally UQ methods are applied directly to the model under investigation and consider all inputs subject to uncertainty. Sensitivity analysis attempts to identify important inputs so as to allow future research to focus on increasing knowledge of the behaviour of these inputs to constrain the input variability and hence reduce the output uncertainty. Response surface methods employ surrogate models which can be used in place of the more expensive true model in any UQ analysis. Sensitivity analysis and response surface methods are described in Sections 1.2.2 and 1.2.3, respectively. Figure 1.2 depicts the stages of the uncertainty quantification process at which sensitivity analysis and response surfaces methods are employed.
Figure 1.2: The most frequently employed paths used to quantify measurable uncertainty. The bottom row depicts four common means of expressing the model uncertainty. The grey boxes highlight where this thesis contributes to the existing literature. Specifically, this thesis focuses on efficient and accurate construction of response surfaces for significantly reducing the number of true model runs required by uncertainty quantification analyses. Evaluation of the true model is often much more computationally expensive to evaluate than the response surface constructed. Note that the response surface methods considered in this thesis enable direct approximation of moments without need for additional quadrature methods. Furthermore, the new response surface method proposed in Chapter 4 automatically identifies important variables, thereby removing the need for sensitivity analysis.
1.2.2 Sensitivity Analysis

The efficiency of the vast majority of UQ methods is heavily dependent on the number of variables used to represent the uncertainty. Reducing the number of variables that are considered can drastically decrease the computational cost of any UQ analysis.

Sensitivity analysis (SA) is used to identify the sources of uncertainty and their relative influence [99]. Specifically SA can be used to determine: which input factors contribute most to the output variability, the presence and types of interactions between input factors; and the presence of non-linearities within the model. Identifying important inputs allows future research to focus on increasing knowledge of the behaviour of the inputs in order to constrain the input variability and hence reduce the output uncertainty. Identifying insignificant inputs can also help refine model structure through the removal of parameters that have negligible effect on the behaviour of the model. Some commonly used SA techniques include: local sensitivity methods; variance based techniques; and Regional Sensitivity Analysis.

Local SA methods, such as automatic differentiation [13, 125] and the Morris method [19, 85], characterise sensitivity by partial derivatives or gradients at the local point. These methods are generally very simple and easy to implement and work well for linear models. However, when the model is non-linear, the results obtained at a nominal point are in general not representative of the entire space;

Variance based techniques, such as the Fourier Amplitude Sensitivity Test (FAST) [23, 100, 102] and the Sobol method [109, 107], involve decomposing the output variance into parts attributed to individual variables and interactions between variables.

Regional Sensitivity Analysis (RSA) [59, 132] partitions model realisations into behavioural sets and non-behavioural sets, that is the set of input factors that satisfy the problem constraints and those that do not.

Sensitivity analysis is a large field and the references listed here represent only a very small subset of the existing literature. The book written by Saltelli et al. [101] provides a comprehensive study of SA methods. Also, the reviews by Frey and Patil [33] and Helton et al [56] provide a clear introduction to SA methods.
1.2. QUANTIFYING UNCERTAINTY

1.2.3 Response Surface Methods

The major limitation of many UQ methods are their slow rates of convergence which limit their application to models which require only a small amount of computational effort to run. Response surface methods (RSM) have arisen as a means of reducing the computational effort of UQ analysis by applying UQ methods to surrogate models

\[ \tilde{M}(\xi) \approx M(\xi) \]

The surrogate models can often be constructed using a small number of model evaluations relative to the number of samples needed by the UQ analysis. Once the surrogate is constructed, it is used as a replacement for the original expensive model during the UQ analysis. If these surrogates can be evaluated much faster than the original model, the use of RSM can drastically reduce the computational effort required by the UQ analysis. Sampling the surrogate model does not require any simulation of the original model. Evaluation of the original model is only required when constructing the model approximation and is a one time cost. Hence when applying any UQ method to a surrogate, a much larger number of samples can be used to obtain the various uncertainty measures than if the original model was used. Surrogate models have been used effectively to increase the efficiency of reliability methods [15, 48], for Bayesian inference [79] and variance based sensitivity indices [97], to name only a few applications.

The efficient and accurate construction of response surfaces is the focus of this thesis. Consequently, we provide here a review of the existing literature. Numerous response surface methods exist. The connections between the various response surface methods are depicted in Figure 1.3. Stochastic Galerkin methods and stochastic collocation methods are two of the most popular approaches\(^1\). These methods will be the focus of this discussion, however many other techniques exist. These include: perturbation methods [65, 73, 94, 126, 134], ANOVA- and CUT-HDMR [93]; random sampling HDMR [70, 71]; Gaussian process emulation [92]; state dependent regression [133, 97]; singular value decomposition based techniques [82], neural networks [114]. These alternative methods are lumped together in Figure 1.3 and labeled alternative response surface methods.

\(^1\)The author has published a review of stochastic Galerkin and collocation methods in the ANZIAM Journal [61]
Figure 1.3: Hierarchy of response surface methods. This thesis focus on construction of those methods contained in the grey boxes. Specifically this thesis is dedicated to the development of interpolation based stochastic collocation methods. Interpolation methods ensure that the approximation is exact at the collocation nodes where as projection methods do not. In Chapter 4 an adaptive sparse grid interpolation method is proposed which can be applied to models with hundreds of unknown variables. This and other similar methods can be applied to domains consisting of one element or many disjoint regions via multi-element approximation. Multi-element methods obtain the highest rates of convergence when the domain is decomposed into regions of high-regularity. With this aim we develop a discontinuity detection algorithm in Chapter 5 which can be used to decompose the domain into the minimal number of elements.
1.2. QUANTIFYING UNCERTAINTY

Stochastic Galerkin Methods

Recently stochastic Galerkin [10, 42, 128, 130] methods have emerged as efficient and highly competitive means of constructing accurate surrogate models in high dimensional spaces. Stochastic Galerkin (SG) methods employ generalised polynomial chaos gPC expansions of the form

$$u(\xi) = \sum_{i=0}^{P} c_i \Phi_i(\xi)$$

(1.1)

to represent the solution and inputs to stochastic differential equations [10, 42, 130]. Here \(\{\Phi_i(\xi)\}\) forms a complete orthogonal basis that depends on the distribution of \(\xi\), and \(P\) is dependent on the dimension \(d\) and order \(M\). The type of polynomial basis used depends on the distribution of the random variables \(\xi\). Utilising the orthogonality of the chosen basis, a Galerkin projection is used to minimise the error of the truncated expansion and the resulting set of \(P+1\) coupled equations can be solved to obtain the expansion coefficients.

SG methods are highly suited to dealing with ordinary and partial differential equations and have the ability to deal with steep non-linear dependence of the solution on random model data [67]. Provided sufficient smoothness conditions are met, PC estimates of uncertainty converge exponentially with the order of the expansion and, for low dimensions, come with a small computational cost.

The SG method has four main limitations. (I) Computational complexity increases rapidly with the number of random inputs and order of the expansion. (II) A high expansion order is required when the dependence of the solution on the random input data varies rapidly or if a singularity exists in the random space [117]. (III) Errors in the gPC approximation of transient solutions may become unacceptably large even after only a short time [75]. (IV) SG necessitates the solution of a system of coupled equations requiring efficient, robust solvers and modification of deterministic code, in order to obtain the gPC coefficients.

Several techniques have been proposed to overcome the problems associated with long-term integration and steep dependence in the random space. Wan and Karniadakis [117, 118], Le Maître et al. [68, 69] and Babuska et al. [10] have developed \(k-p\) gPC approximations which combine local \((-k)\) refinement in the stochastic space with increases in the polynomial degree \((-p)\). Wan [117, 118] advocates the use of Multi-Element generalised polynomial chaos (ME-gPC) which adaptively decomposes the random input space into sub-domains or elements. This technique takes advantage of the observation that SG is more efficient for
random variables with standard deviation that is relatively small in comparison to its mean. Although utilising the fast (exponential) convergence of gPC approximation the complexity of this method reduces the observed rate of convergence. Le Maitre [68] advocates the use of an orthogonal projection of uncertain data and solution variables onto a wavelet basis, consisting of compact piecewise-smooth polynomial functions. The use of a localised polynomial basis results in a low-order rate of convergence. Both these localised expansions produce more robust schemes than the global gPC expansion and the adaptive decomposition of random space efficiently controls the error of long-term integration. However, these methods are based upon tensor product spaces in higher dimensions and suffer the curse of dimensionality.

Adaptive $p$-type methods have been developed by Li [72] and Lucor [75]. These methods are based on the observation that some of the terms in the gPC representation of the solution to the governing equations do not contribute significantly to its value. Only the terms which have the greatest contribution to the solution are kept. Schwab and Todor [113] propose a sparse polynomial chaos approximation for elliptic problems with stochastic coefficients, as an alternative to tensor product polynomial approximation bases.

**Stochastic Collocation Methods**

Recently a new technique referred to as stochastic collocation (SC) [9, 80, 112, 129] has arisen to address some limitations of SG. SC endeavours to combine the strengths of non-intrusive sampling and SG. As with Monte Carlo methods, SC requires only the solution of a set of decoupled equations, allowing the model to be treated as a black box and run with existing deterministic solvers. Provided the solutions possess sufficient smoothness in the random space, SC methods maintain the fast convergence of SG [129].

Stochastic collocation seeks to approximate the solution to the governing equations using interpolation

$$u(\xi) = \sum_{k=0}^{Q} u_k(\xi^{(k)}) \Psi_k(\xi)$$

(1.2)

where $u_k(\xi^{(k)})$ is the solution at the collocation points $\{\xi^{(k)}\}_{k=0}^{Q}$ and $\Psi_k$ is a multi-dimensional basis.

Sparse grid interpolation has emerged as the most useful tool to construct the multi-dimensional approximation. Sparse grids have been extensively used
1.2. QUANTIFYING UNCERTAINTY

for high-dimensional interpolation and quadrature [11, 40]. The approach can yield several orders of magnitude reduction in the number of collocation points required to achieve the same level of accuracy as the full tensor grid [89, 129]. Stochastic collocation methods are discussed extensively in Chapter 2.

Standard sparse grids are isotropic, treating all dimensions equally. Although an advance on full tensor product spaces, such approximations can still be improved. Many problems vary rapidly in only some dimensions, remaining much smoother in other dimensions. Consequently it is advantageous to increase the level of accuracy only in certain non-smooth dimensions, resulting in so-called adaptive or anisotropic grids. In some cases the important dimensions can be determined a priori, but in most cases the collocation points must be chosen during the computational procedure. Examples of dimension-adaptive sparse grid collocation schemes are given by Gana [35] and Nobile [88].

Ma and Zabaras [76, 78] developed an adaptive sparse grid collocation method called Adaptive Sparse Grid Collocation (ASGC). This method uses piecewise multi-linear hierarchical basis functions, based upon the wavelet based representations found in the deterministic sparse grid literature [46, 115]. ASGC achieves the benefits of local adaptivity of the ME-gPC and Wiener–Harr expansions whilst remaining non-intrusive and allowing application to higher-dimensional problems. Although implicitly dimension-adaptive the efficiency of this method can be increased by including explicit dimension adaptivity. With this goal, Ma [78] developed ASGC-HDMR which combines ASGC with a dimension adaptive procedure that constructs the important sub-dimensional components of the anchored ANOVA decomposition of a function. Note that aside from producing efficient surrogates, the dimension adaptive nature of ASGC-HDMR facilitates the identification of important parameters and dimensions [31, 78]. The detection of important dimensions is necessary for approximation in very high dimensional spaces but also often removes the need for sensitivity analysis.

Foo et al. [32] propose a Multi-Element Probabilistic Collocation Method (MEPCM) which adaptively discretises the random space into elements and implements a local isotropic sparse grid collocation problem on each element. To avoid an exponential growth in the number of elements only a small subset of the important dimensions is refined. Foo [31] increased the utility of MEPCM for high-dimensional problems by combining this approach with truncated ANOVA functional decomposition. A similar dimension-wise decomposition was proposed by Bieri and Schwab [14] for stochastic collocation of elliptic partial differential equations. The MEPCM and MEPCM-A methods have been shown to be nu-
merically efficient quadrature methods and their utility for interpolation has been identified [32, 31]. Indeed Agarwal [3] has presented an MEPCM approach for interpolation using linear reconstructions on each element.

Projection methods also known as pseudo-spectral methods have also been proposed as a non-intrusive means of constructing a response surface [112, 127]. These approaches involve using an appropriate quadrature rule [98, 127] or regression [1, 66] to project the model onto a pre-defined basis using a set of independent model runs. Typically orthogonal polynomials such as gPC expansions are used [127], but other bases, such as one based upon local piecewise-linear functions [1] have also been employed.

The unknown coefficients can be approximated using non-linear least squares. But, if quadrature is used the coefficients are determined by evaluating an integral equation for each basis coefficient. Sparse grid quadrature is one popular choice used to approximate the high-dimensional integrals needed to determine the coefficients of a polynomial expansion [127]. Other quadrature methods such as Latin-hyper cube sampling [98] and tensor products of one-dimensional Gaussian quadrature rules [28] have also been used. Note that using a pseudo-spectral construction, regardless of its exact construction, introduces an aliasing error that can become quite large in high-dimensional spaces. Unlike interpolation based stochastic collocation the use of pseudo-spectral methods does not guarantee that the approximation will be exact at the collocation nodes.

In Figure 1.3 we depict interpolation methods such as ASGC and pseudo spectral methods as forms of multi-element approximation. Typically interpolation methods, such as ASGC, and projection methods are applied on only one element that covers the entire input space. However ASGC or other interpolation or projection methods could easily be used on a set of elements in place of the isotropic sparse grid scheme used by Foo et al. [32].

Note that, similar to SG methods, errors in the SC approximation of transient solutions may become unacceptably large even after only a short time. This long-time integration issue is not a just a failing of SG and SC methods, but rather it is indicative of all approximation methods when applied to time-variant functions subject to uncertainty [45].
1.3 Thesis Objectives and Outline

This thesis deals with the efficient numerical treatment of uncertainty quantification for high-dimensional problems. The aim is to develop a set of robust stochastic collocation response surface methods that maintain high-rates of convergence even in the presence of discontinuities. The main contributions of this thesis include:

- Presentation of a numerical framework for the quantification of epistemic uncertainty. Traditionally stochastic collocation has only been used to quantify statistical input uncertainty arising from natural variability, known as aleatory uncertainty. The proposed methodology does not require any probabilistic information on uncertain input parameters. It only necessitates an estimate of the range of the uncertain variables that encapsulate the true range of the input variables with overwhelming probability. To quantify the epistemic uncertainty, we solve an encapsulation problem, which is a solution to the original governing equations defined on the estimated range of the input variables. We discuss solution strategies for solving the encapsulation problem and the sufficient conditions under which the numerical solution can serve as a good estimator for capturing the effects of the epistemic uncertainty. In the case where probability distributions of the epistemic variables become known \textit{a posteriori}, we can use the information to post-process the solution and evaluate solution statistics. Convergence results are also established for such cases, along with strategies for dealing with mixed aleatory and epistemic uncertainty. This work has been published in the Journal of Computational Physics [63].

- The development of a sparse grid algorithm for stochastic collocation for high-dimensional discontinuous problems. The proposed method combines the strength of the generalised sparse grid algorithm [41] and \( h \)-refinement guided by the hierarchical surplus [76, 78], to efficiently interpolate discontinuous problems with hundreds of dimensions We coin this approach the \( h \)-Adaptive Generalised Sparse Grid (\( h \)-GSG) method. The method adaptively identifies the effective dimensionality of the function under consideration, whilst limiting the number of collocation nodes invested in 'unimportant' regions and dimensions. The method is used to build a response surface for a model simulating flow through porous media.

- The development a new algorithm for the detection of discontinuities in
functions with a large number of variables. This algorithm can be used to classify the input domain into disjoint smooth regions on which global approximation methods and their high-rates of convergence can be employed. It is demonstrated that in the commonly encountered cases where a discontinuity resides on a small subset of the dimensions, the present method becomes "optimal", in the sense that the total number of points required for function evaluations depends linearly on the dimensionality of the space.

The work contained in this chapter is to appear in the Journal of Computational Physics [62].

The remainder of the thesis is formulated as follows. Chapter 2 introduces the stochastic collocation method as a means for constructing efficient surrogate models. We provide the problem formulation that will be used throughout this thesis, discuss various methods for introducing uncertainty and present the basic construction of stochastic collocation methods.

In Chapter 3 we present a framework for quantifying epistemic uncertainty using the various methods developed throughout the thesis. We propose solving an "encapsulation problem" which generates a solution to the governing equations in a domain that encloses the true (and unknown) probability space. We discuss the construction and solution of the encapsulation problem. In particular we illustrate how stochastic collocation (and Galerkin) methods can be used to construct efficient and accurate approximations of the solution to the encapsulation problem. We also extend the encapsulation approach to models with mixed epistemic and aleatory uncertainty and discuss how to extract and interpret important statistical information.

Solving the encapsulation problem posed in Chapter 3 motivates the need for efficient construction of response surfaces that possess point-wise convergence. In Chapter 4 we present one such construction. We review the generalised sparse grid algorithm and hierarchical surplus guided $h$-adaptivity. We then combine the strengths of these two methods to construct an efficient stochastic collocation method that can be used for problems with hundreds of variables. Error bounds are presented and numerical convergence is studied for a number of examples. The proposed method is then used to construct a surrogate of a model simulating flow through porous media.

Chapter 5 proposes a high-dimensional discontinuity detection method. It is hoped that this method can be eventually used for discretising high-dimensional domains into a minimal number of sub-domains whose interfaces lie exactly along
the discontinuities. We present the one-dimensional polynomial annihilation discontinuity detection method and demonstrate how this method can be combined with adaptive strategies to efficiently characterise discontinuities in high dimensions. Various numerical examples are utilised to demonstrate the efficacy of the method. We also identify an approach to combine the proposed discontinuity detection method with polynomial interpolation to produce a multi-element interpolation method that is constructed using a near minimal number of elements.

Finally, conclusions and possible avenues of future work are provided in Chapter 6.
Chapter 2

Stochastic Collocation

In Chapter 1 we presented a number of methods for quantifying uncertainty in mathematical models arising from uncertainty in model input data. Many other sources of uncertainty exist (refer to Section 1.1), but the non-numerical nature of these uncertainties makes quantifying their effect on the model behaviour difficult.

Stochastic Collocation (SC) methods, first outlined in Section 1.2.3, are frequently used to efficiently quantify uncertainty in mathematical models subject to random statistical input. The SC approach is used extensively throughout this thesis and is the basis of all the new methods presented.

In this chapter we discuss in detail the formulation and properties of the stochastic collocation method. Specifically we present the basic formulation of SC, some commonly used approaches for representing input uncertainty, and outline the how these representations of the input uncertainty can be propagated through a model.

2.1 Formulation

Traditionally stochastic collocation has focused on quantifying the effect of aleatory uncertainty. The sources of uncertainty are typically represented using a probabilistic framework under which the aleatory uncertainty can be represented by a finite number of random variables with some known distribution. The sources of aleatory uncertainty include both uncertainty in model coefficients (parametric uncertainty) and uncertainty in the sequence of possible events (stochastic uncertainty). In this chapter we restrict attention to the aleatory setting, but remark that stochastic collocation can be used to quantify uncertainty even when full distributional information is unavailable. Quantifying this epistemic uncertainty
is the subject of Chapter 3.

To quantify aleatory uncertainty we adopt a probabilistic approach and define a complete probability space \((\Omega, \mathcal{F}, \mathcal{P})\). This space consists of an event space \(\Omega\), comprising possible outcomes \(\omega\), a \(\sigma\)-algebra \(\mathcal{F}\) and a probability measure \(\mathcal{P}: \mathcal{F} \to \mathbb{R}[0,1]\).

Utilizing this framework, the uncertainty in a model can be introduced by representing the model input data as random fields \(X(\omega)\). These random fields \(X: \Omega \to V\) are mappings from the probability space into a function space \(V\). If \(X\) is a random variable then \(V = \mathbb{R}\) and \(X: \Omega \to \mathbb{R}\). Alternatively if \(X\) is a random field or process, such as a Wiener process, \(V\) is a function space over a temporal and/or spatial interval.

Following convention for ease of exposition, consider the general differential equation defined on an \(s\)-dimensional bounded domain \(D \subseteq \mathbb{R}^s\) \((s = 1, 2, 3)\)

\[
\begin{align*}
\mathcal{L}(x, t, \omega; u) &= f(x, t, \omega), \quad x \in D, \ t \in T \\
\mathcal{B}(x, t, \omega; u) &= g(x, t, \omega), \quad x \in \partial D, \ t \in T \\
u(x, 0, \omega) &= h(x, \omega), \quad x \in \bar{D}
\end{align*}
\]

where \(x = (x_1, \ldots, x_s)\) are the coordinates in \(\mathbb{R}^s\), \(s \geq 1\), \(T = (t_0, T)\) are the temporal coordinates in \(\mathbb{R}\), \(\mathcal{L}\) is a linear or non-linear differential operator and \(\mathcal{B}\) is a boundary operator. We are interested in finding the stochastic solution \(u: \Omega \times \bar{D} \times T \to \mathbb{R}\) such that for \(\mathcal{P}\)-almost everywhere \(\omega \in \Omega\), equation (2.1) holds.

The boundary operator can take a number of forms, for example \(\mathcal{B} = n \cdot \nabla\) on Neumann segments and the identity operator on Dirichlet segments. Furthermore the boundary \(\partial D\) can be divided into disjoint subsets \(\partial D_d\) and \(\partial D_n\) on which Dirichlet and Neumann boundary conditions can be applied respectively. The partitioning must satisfy \(\partial D_d \cup \partial D_n = \partial D\) and \(\partial D_d \cap \partial D_n = \emptyset\).

Note the application of stochastic collocation is not restricted to models based upon the differential equation formulation presented here.

### 2.2 Representing the Random Input Data

To solve equation (2.1) numerically we must invoke the 'finite noise assumption' [9] to reduce the infinite-dimensional probability space \((\Omega, \mathcal{F}, \mathcal{P})\) to a finite-dimensional space. Employing any truncated spectral expansion of the stochastic process in the probability space we can characterize the random inputs by a set of \(d\) random variables \(\xi = (\xi_1(\omega), \ldots, \xi_d(\omega))\). By the Doob–Dynkin Lemma [96],
2.2. **REPRESENTING THE RANDOM INPUT DATA**

the solution can then be described by the same set of random variables, that is \( u(x, t, \omega) = u(x, t, \xi) \).

It is generally assumed that \( \xi_n(\omega) \) are mutually independent random variables with images \( \Gamma_n \) and probability density functions \( \rho_n : \Gamma_n \to \mathbb{R} \). Under this assumption the joint density of \( \xi \) is

\[
\rho(\xi) = \prod_{n=1}^{d} \rho_n(\xi_n), \quad \xi \in \Gamma
\]

defined over the support

\[
\Gamma = \prod_{n=1}^{d} \Gamma_n \subseteq \mathbb{R}^d
\]

Furthermore we assume without loss of generality that the random variables have identical probability density functions. In theory, random variables with different PDFs can be transformed to the same PDF.

Under the finite noise assumption we can now write the stochastic problem (2.1) as a deterministic problem: find \( u : \Gamma \times \bar{D} \to \mathbb{R} \) such that for \( \mathcal{P} \)-almost everywhere \( \xi \in \Gamma \) the following equation holds.

\[
\begin{align*}
\mathcal{L}(x, t, \xi; u) &= f(x, t, \xi), \quad x \in D, \ t \in T \\
B(x, t, \xi; u) &= g(x, t, \xi), \quad x \in \partial D, \ t \in T \\
u(x, 0, \xi) &= h(x, \xi), \quad x \in \bar{D}
\end{align*}
\]

### 2.2.1 Parametric Uncertainty

Parametric uncertainty of problem (2.2), often used in model sensitivity and calibration exercises, can be investigated by representing the parameters of interest as a \( d \)-variate random vector \( \xi \) of mutually independent random variables \( \xi_n, \ n = 1, \ldots, d \). Additional parameters may be present in (2.2) but these can either be written as functions of existing parameters or are not of interest.

When solely considering parametric uncertainty the decomposition of the random space is trivial because the random inputs already take the form of random variables. Refer to Xiu [127] for a detailed discussion of efficient stochastic collocation approaches for parametric uncertainty analysis.

### 2.2.2 Karhunen–Loève Expansion

If the input data of equation (2.2) are spatially and/or temporally dependent random fields, an alternative decomposition of the stochastic processes must be
used. A popular choice for second order stochastic processes is the finite-term Karhunen-Loève (KLE) expansion [74].

Consider a random field \( X(x, \omega) \) generated by the covariance function \( C(x_1, x_2) \) where \( x_1 \) and \( x_2 \) are spatial or temporal coordinates. \( X : \Omega \rightarrow L^2(\Omega) \) is a second order random field in the Hilbert space \( L^2(\Omega) \) equipped with the inner product norm

\[
\langle f(\omega), g(\omega) \rangle = \int_{\omega \in \Omega} g(\omega)f(\omega)dP(\omega)
\]

and

\[
\|X(\omega)\|^2 = \langle X(\omega), X(\omega) \rangle < \infty
\]

Under these conditions the finite-term KLE of such a stochastic process is

\[
X(x, \omega) = \bar{X}(x) + \sum_{n=1}^{d} \sqrt{\lambda_n} \phi_n(x) \xi_n(\omega), \quad x \in D, \omega \in \Omega
\]  
(2.3)

where \( \bar{X} \) is the mean of the random process, \( \xi_n(\omega) \) are a set of uncorrelated random variables determined by

\[
\xi_n(\omega) = \int_D X(x, \omega) \phi_n(\omega)dx, n = 1, \ldots, N
\]

The set of eigenvalues \( \lambda_n \) and a corresponding set of orthogonal eigenfunctions \( \phi_n(x) \) of \( C(x, \omega) \) form a complete basis spanning the function space to which \( C(x, \omega) \) belongs. Specifically \( \psi_n \) are eigenfunctions of the integral operator

\[
\int_D C(x_1, x_2) \psi_n(x_2) dx_2 = \lambda_n \psi_n(x_1)
\]

corresponding to \( C \).

### 2.2.3 Alternative Decompositions

Principle component analysis (PCA) [2, 34], random walks [43], and Brownian bridge [86] constructions can also be used to decompose a random field into a finite set of random variables. These methods are often used to decompose temporal random fields \( W(t, \omega) \) arising from Brownian motion [47]. Although varying in their construction these methods all generate a \( d \)-dimensional vector \( W = (W_1, \ldots, W_d) \) whose joint distribution coincides (at the discrete time points \( t_i = i\Delta t, i = 1, \ldots, d \) and \( \Delta t = T/d \)) with the joint distribution of the continuous time Brownian motion process \( W(t, \omega) \) up until time \( T \). All random vectors satisfy

\[ W = Ay \]
where \( y = (y_1, \ldots, y_d) \) is a vector of \( d \) independent and normally distributed random numbers and \( \mathbf{A} \) satisfies \( \mathbf{A}^T \mathbf{A} = \mathbf{C} \), where \( \mathbf{C} \in \mathbb{R}^{d \times d} \) is the covariance matrix of the Brownian motion with entries \( c_{i,j} = \text{Cov}(W(t_i, t_j)) = \min(t_i, t_j) \).

The Brownian bridge, random walk and PCA constructions only vary in the choice of matrix \( \mathbf{A} \).

Principle component analysis \([2, 34]\) is based upon an eigenvalue decomposition of the covariance matrix \( \mathbf{C} = \mathbf{V}^T \mathbf{A} \mathbf{V} \). It maximizes the concentration of the total variance of the random field in the first few dimensions. Here \( \mathbf{A} \) satisfies

\[
\mathbf{A} = \sqrt{\mathbf{A}} \mathbf{V}
\]

The random walk construction \([43]\) iteratively calculates the components of \( \mathbf{W} \)

\[
W_i = W_{i-1} + \sqrt{\Delta t} y_i, \quad i = 1, \ldots, d
\]

with \( W_0 = 0 \). This is equivalent to setting \( \mathbf{A} \) as the Cholesky matrix of \( \mathbf{C} \). In contrast the Brownian bridge construction \([86]\) determines the components \( W_i \) hierarchically so that the lower index terms are more important than the higher index terms. The explicit form of \( \mathbf{A} \) can be found in \([121]\).

### 2.3 Collocation Method

Representing the input by a finite number of random variables transforms the original stochastic problem (2.1) into the fully deterministic problem (2.2). For any random vector \( \boldsymbol{\xi} \), the exact solution to equation (2.2) will always be the same everytime the model is run using that particular realisation of \( \boldsymbol{\xi} \).

For ease of discussion, and without loss of generality, consider (2.2) with the spatial and temporal dependence removed. Moreover, let us restrict attention to real valued functions with finite second moments. To solve (2.2) we must consider the variational formulation: find \( u(\boldsymbol{\xi}) \in L^2_{\rho}(\Gamma) \) such that

\[
\int_\Gamma \mathcal{L}(\boldsymbol{\xi}, u) v(\boldsymbol{\xi}) \rho(\boldsymbol{\xi}) d\boldsymbol{\xi} = \int_\Gamma f(\boldsymbol{\xi}) v(\boldsymbol{\xi}) \rho(\boldsymbol{\xi}) d\boldsymbol{\xi}, \quad \forall v(\boldsymbol{\xi}) \in \hat{\mathcal{V}}
\]

where \( \mathcal{V} \) is referred to as the trial space and \( \hat{\mathcal{V}} \), the test space. Choosing \( \mathcal{V} = \hat{\mathcal{V}} \subset L^2_{\rho}(\Gamma) \) results in the stochastic Galerkin method (see Section 1.2.3).
CHAPTER 2. STOCHASTIC COLLOCATION

2.3.1 Specifying the Test Space

The stochastic collocation method is obtained by choosing a test space of the form

\[ \hat{V} = \left\{ \hat{v} : \Gamma \rightarrow \mathbb{R} : \hat{v} \in \text{span}\{\delta(\xi - \xi_k)\}_{k=0}^{Q} \right\} \]

With this choice of the test space the variational formulation (2.4) is satisfied exactly at \( Q \) distinct points \( \{\xi_k\}_{k=1}^{Q} \). The number of points \( Q \) is dependent on the construction of the trial space \( V \). The stochastic collocation method is equivalent to solving \( Q \) deterministic problems at each nodal point \( \xi_k \). That is

\[ \mathcal{L}(\xi_k; u(\xi_k)) = f(\xi_k) \tag{2.5} \]

The problem (2.5) is decoupled and existing deterministic solvers can be applied to find each realisation of the solution.

2.3.2 Constructing the Trial Space

Numerous constructions can be used for the trial space in (2.4). The simplest construction of stochastic collocation methods is based upon the span of tensor product polynomials with a fixed maximum degree \( p = (p_1, \ldots, p_d) \). Denoting

\[ V_i = \left\{ v : \Gamma_i \rightarrow \mathbb{R} : v \in \text{span}\{\Psi_{j}^{p_i}\}_{j=0}^{p_i} \right\}, \quad i = 1, \ldots, d \]

the one-dimensional subspaces of \( V \) we obtain

\[ V(\Gamma) \approx V_p = \bigotimes_{i=1}^{d} V_i(\Gamma_i) \]

Typically the Lagrange basis \( \{l_j\}_{j=1}^{p_i+1} \) is used to decompose the one dimensional spaces \( V_i \) [9, 129]. The Lagrange basis is given by

\[ l_j(\xi) = \prod_{k=1}^{p_i+1} \frac{\xi - \xi_k}{\xi_j - \xi_k}, \quad j = 1, \ldots, p_i + 1 \]

and satisfies the orthogonality condition \( l_j(\xi_k) = \delta_{j,k} \).

The choice of the collocation nodes upon which the Lagrange basis is built depends on the distribution of the random variables \( \xi \). For example the Gauss-Hermite abscissa are used for Gaussian distributions, Gauss-Jacobi for beta distributions, Gauss-Legendre for uniform variables, and so on.
2.4. CONCLUSIONS

The size of the finite dimensional space $V_p$ is $\prod_{i=1}^{d} (p_i + 1)$ grows exponentially with dimension $d$ and thus suffers from the curse of dimensionality. Consequently alternative methods are required to discretize the input space $\Gamma$. Sparse grid based approaches, discussed in Chapter 4, are one such alternative.

Here we mention pseudo-spectral methods which use spectral expansions of orthogonal polynomials to decompose $V$. Least squares or quadrature methods are used to determine the coefficients of the expansion. However unlike interpolation based collocation methods, such an approach does not guarantee the approximation will pass through the collocation nodes and can suffer aliasing error.

2.4 Conclusions

In this chapter we presented the necessary information to implement a stochastic collocation method. The stochastic collocation method is the basis of all the new methods proposed in this thesis. In Chapter 3 we present a framework, based upon stochastic collocation, for quantifying input uncertainty. Then in Chapter 4 we present a new method for constructing the trial space of the variational formulation of the stochastic collocation method. Finally Chapter 5 proposes a new method for discontinuity detection which can be used to decompose the parametric space into regions of high-regularity. It is hoped that this method can be eventually used to construct a multi-element collocation method which uses the discontinuity information provided to determine the optimal decomposition of the input space.
Chapter 3

Quantifying Epistemic Uncertainty

Recall from Chapter 1 that the nature of quantifiable input uncertainty can either be: aleatory or epistemic. Aleatory uncertainty arises from the inherent variation associated with the system under consideration and is irreducible. Epistemic uncertainty represents any lack of knowledge or information in any phase or activity of the modeling process [58] and is reducible through the introduction of additional information.

The stochastic collocation method, presented in Chapter 2, is frequently used to quantify the uncertainty in model states and observables arising from aleatory uncertainty in the inputs [3, 32, 76, 129]. The sources of aleatory uncertainty are typically represented using a probabilistic framework under which the aleatory uncertainty can be represented by a finite number of random variables with some known distribution.

Frequently, strong statistical information such as probability distribution functions or high order statistical moments is not available. Experimental data needed to construct this information is often expensive and consequently no data, or only a small collection of data points, may be obtainable. In these cases "expert opinion" is used in conjunction with the available data to produce weak inferential estimates of parametric characteristics, often in the form of lower and upper bounds. Other sources of epistemic uncertainty include limited understanding or misrepresentation of the modeled process, known commonly as "model structure" uncertainty. Inclusion of "enough" additional information about either the model parameters or structure can lead to a reduction in the predicted uncertainty of a model output. Consequently, we can consider epistemic uncertainty as provid-
CHAPTER 3. QUANTIFYING EPISTEMIC UNCERTAINTY

ing (conservative) bounds on an underlying aleatory uncertainty, where reduction and convergence to the true aleatory uncertainty (or, in some cases, a constant value) can be obtained given sufficient additional information.

Until recently, most uncertainty analysis has focused on aleatory uncertainty. Numerous methods have been developed that provide accurate and efficient estimates of this form of uncertainty. In comparison to the quantification of aleatory uncertainty, the analysis of epistemic uncertainty has proved more challenging. Probabilistic representations of epistemic uncertainty are inappropriate, since the characterization of input epistemic uncertainty through well-defined probability distributions imposes a large amount of unjustified structure on the influence of the inputs on the model predictions. This can result in stronger inferences than are justified by the available data. Evidence theory \[56\], possibility theory \[24\] and interval analysis \[55, 110\] have been proposed as more appropriate alternatives, where they are listed in descending order based on the amount of imposed input structure.

Of the aforementioned methods, evidence theory is the most closely related to probability theory. Evidence theory starts from basic probability assignments on the inputs, propagates these descriptions through a model using standard sampling techniques, and produces estimates of the lowest and highest probabilities of the model observables. These estimates define cumulative belief and cumulative plausibility functions that represent the uncertainty in the output metrics, where belief provides a measure of the amount of information that supports an event being true and plausibility measures the absence of information that supports the event being false. The evidence theory representation of uncertainty approaches the probabilistic representation as the amount of information about the input data increases \[56\].

Possibility theory is closely related to fuzzy set theory and, similar to evidence theory, utilizes two descriptions of likelihood, necessity and possibility. These two measures are based upon the properties of individual elements of the universal set of events, unlike plausibility and belief which are derived from the properties of subsets of the universal set. For more details, see \[54\].

Evidence and possibility theory require aggregation of data from multiple sources into a format consistent with the chosen technique. In practice, this can be difficult and time consuming. Interval analysis \[83\], on the other hand, only requires upper and lower bounds on the uncertain input data. Sampling and/or optimization \[25, 110\] is then used to generate upper and lower bounds (intervals) on the model outputs from predefined intervals on the input data.
The application of evidence theory, possibility theory and interval analysis to non-linear and complex problems often requires a prohibitively large number of samples and typically underestimates the output extrema. Global surrogate models have been used in an attempt to alleviate this problem [54]; however, the performance of these approaches is highly dependent on the accuracy of the surrogate model and construction costs can be high when global accuracy is required and convergence rates are not exponentially fast. In more recent work, surrogates with adaptive refinement strategies have been combined with stochastic collocation methods [25, 26, 110, 111] in order to segregate aleatory quantification with stochastic expansions from epistemic quantification using optimization-based interval estimation.

The choice of the aforementioned methods depends on the amount of available information which can be utilized to characterize the input uncertainty. Consequently this choice is highly problem dependent. Here we propose a new and more general framework to numerically quantify epistemic uncertainty. This proposed method can deal with varying amounts of information on the input data from simple bounds to full probabilistic descriptions, and thus can seamlessly handle the problems with both epistemic and aleatory uncertainties. Furthermore the proposed approach utilizes classical approximation theory in multidimensional space and achieves higher efficiency than the methods currently available.

Unlike many existing numerical methods for quantifying epistemic uncertainty, the proposed method requires only an approximation of the ranges of the input data that encapsulates the “true” bounds of the input data.

We then propose solving an “encapsulation problem” which generates a solution to the governing equations in a domain that encloses the true (and unknown) probability space. Here a multidimensional polynomial expansion can be employed to approximate the solution on the larger encapsulation space. We show that if such a representation converges in the encapsulation space then this method will also converge in the true probability space. Furthermore, convergence is maintained even in the presence of dependencies between input data. We also demonstrate numerically that, if the distributions of the input data are found a posteriori, the polynomial approximation of the solution statistics will converge.

In Section 3.1, we present the necessary mathematical framework for quantifying epistemic uncertainty and in Section 3.2 we discuss the construction and solution of the encapsulation problem. In particular we focus on polynomial based collocation methods and illustrate how these methods can be used to construct
efficient and accurate approximations of the solution to the encapsulation problem. We also extend the encapsulation approach to models with mixed epistemic and aleatory uncertainty and discuss how to extract and interpret important statistical information. Numerical examples are presented in Section 3.3 and we conclude the chapter in Section 3.4. This chapter follows the author's published work in the Journal of Computational Physics [63].

3.1 Problem Setup

As in Chapter 2, let \( D \subset \mathbb{R}^s, \ s = 1, 2, 3, \) be a physical domain with coordinates \( x = (x_1, \ldots, x_s) \) and let \( T > 0 \) be a real number. We consider the following general stochastic partial differential equation

\[
\begin{align*}
  v_t(x, t, \xi) &= \mathcal{L}(v), \quad D \times (0, T] \times I_\xi, \\
  B(v) &= 0, \quad \partial D \times [0, T] \times I_\xi, \\
  v &= v_0, \quad D \times \{t = 0\} \times I_\xi,
\end{align*}
\]

(3.1)

where \( \mathcal{L} \) is a (nonlinear) differential operator, \( B \) is the boundary condition operator, \( v_0 \) is the initial condition, and \( \xi = (\xi_1, \ldots, \xi_d) \in I_\xi \subset \mathbb{R}^d, d \geq 1, \) are a set of random variables characterizing the random inputs to the governing equation. The solution is therefore a stochastic quantity,

\[
v(x, t, \xi) : D \times [0, T] \times I_\xi \to \mathbb{R}^{n_v}.
\]

(3.2)

Without loss of generality, hereafter we assume (3.1) is a scalar system with \( n_v = 1. \) We also make a fundamental assumption that the problem (3.1) is well posed in \( I_\xi. \)

Most of the existing studies adopt a probabilistic formulation to quantify \textit{aleatory} uncertainty. That is, it is typically assumed that the distribution of the random variables \( \xi \) is known, with the most widely adopted approach assuming the marginal distributions of \( \xi_i \) are known and all \( \xi_i \) are independent from each other. In this chapter, however, we consider the case where the uncertainty is \textit{epistemic}. That is, the distribution functions of \( \xi_i \) are not known, primarily due to our lack of understanding and characterization of the physical system governed by the system of equations (3.1).

The focus is on the dependence of the solution on the uncertain inputs \( \xi; \) therefore, we present solutions for fixed location \( x \) and time \( t \) and omit these variables whenever possible.

Note the approach proposed here is not restricted to models based upon the differential equation formulation presented here.
3.2 Methodology

We now present a method for solving system (3.1) subject to epistemic uncertain inputs. The proposed methodology is a three-step procedure which involves identifying the ranges of the uncertain inputs, generating an accurate polynomial approximation of the solution to (3.1) within estimated ranges and post-processing the results. Note that no probability distribution information will be utilized in the solution procedure.

3.2.1 Range Estimation

Unlike traditional aleatory uncertainty quantification, the proposed method only requires an estimate of the ranges of the random input.

The goal is to identify a range, or bound, that is sufficiently large such that the “true”, and yet unknown, range of the input uncertainty is mostly incorporated. We now illustrate the idea more precisely.

For each random variable $\xi_i$, $i = 1, \ldots, d$, let

$$I_{\xi_i} = [\alpha_i, \beta_i], \quad \alpha_i < \beta_i,$$

be its (unknown) support. We consider the following two cases.

- Bounded. That is, $I_{\xi_i}$ is a bounded interval with

$$-\infty < \alpha_i < \beta_i < \infty$$

- Unbounded. That is, either

$$\alpha_i = -\infty \quad \text{and/or} \quad \beta_i = \infty$$

This implies that the distribution of $\xi_i$ has tail(s) near infinity.

The goal of range estimation is to identify a bounded interval

$$I_{X_i} = [a_i, b_i], \quad -\infty < a_i < b_i < \infty,$$

such that $I_{X_i}$ and $I_{\xi_i}$ overlap each other with sufficiently large probability. Let us consider the symmetric difference between $I_{X_i}$ and $I_{\xi_i}$, i.e.,

$$I_i^- = I_{\xi_i} \triangle I_{X_i} = (I_{\xi_i} \cup I_{X_i}) \setminus (I_{\xi_i} \cap I_{X_i}).$$
We then require that the range $I_{X_i}$ is defined in (3.4) such that, for a small real number $\delta_i \geq 0$,

$$P(\xi_i \in I_i) \leq \delta_i. \quad (3.6)$$

Intuitively speaking, this requires the tail probability of $\xi_i$, if there is any, outside the bounded interval $I_{X_i}$ to be sufficiently small. It can always be achieved by choosing the $I_{X_i}$ sufficiently wide. If $I_{\xi}$ is bounded then we can always choose $I_{X_i}$ to overlap $I_{\xi}$ completely. In such a case, $I_{\xi} \subseteq I_{X}$, and $P(\xi \in I^-) = 0$.

Note that the proposed method removes the need for accurate estimates of the input ranges. The estimate of the range must ensure that the range be sufficiently wide such that it encapsulates the “true” input range either completely if the “true” range is bounded, or with overwhelming probability if the “true” range is unbounded. The specific techniques for range estimation is not the focus of this chapter.

In addition to ensuring the encapsulation condition (3.6) is satisfied one must also ensure that the governing equations (3.1) are well-posed in the region of $I_{X_i}$. That is, the “over-estimation” part of the $I_{X_i}$, $I_{X_i} \cap I^-$, does not pose any problem for the solution of (3.1). Consequently the properties of the system (3.1) must be used to guide the range estimation procedure.

### 3.2.2 Encapsulation Problem

Let $I_{\xi}$ be the range of the random variables $\xi \in \mathbb{R}^d$. Naturally,

$$I_{\xi} \subseteq \times_{i=1}^d I_{\xi_i}. \quad (3.7)$$

We also define an encapsulation set

$$I_X = \times_{i=1}^d I_{X_i} = \times_{i=1}^d [a_i, b_i]. \quad (3.8)$$

which is the Cartesian product of (3.4). Now let

$$I^+ = I_{\xi} \cup I_X, \quad I^0 = I_{\xi} \cap I_X, \quad (3.9)$$

and

$$I^- = I_{\xi} \triangle I_X = I^+ \setminus I^0, \quad (3.10)$$

be the symmetric difference of $I_{\xi}$ and $I_X$. By following the construction of the range estimate in (3.5) and (3.6), it is easy to see that

$$P(\xi \in I^-) \leq \delta, \quad \delta = 1 - (1 - \max_i \delta_i)^d. \quad (3.11)$$
3.2. METHODOLOGY

Therefore, the encapsulation set $I_X$ encapsulates $I_\xi$, the "true" and unknown support of $\xi$, with probability at least $1 - \delta$, where $\delta \geq 0$ can be made small by enlarging the size of $I_X$. The parameter $\delta$ can be zero, i.e., $I_X$ encapsulates $I_\xi$ with probability one, when $I_\xi$ is a bounded domain.

We now define the following encapsulation problem

\[
\begin{cases}
    u_t(x,t,X) = \mathcal{L}(u), & D \times (0,T] \times I_X, \\
    B(u) = 0, & \partial D \times [0,T] \times I_X, \\
    u = u_0, & D \times \{t = 0\} \times I_X,
\end{cases}
(3.12)
\]

where $I_X$ is the bounded hypercube defined in (3.8). This is effectively the same problem (3.1) defined now on the encapsulation set $I_X$ that covers the original random parameter set $I_\xi$ with probability at least $1 - \delta$. The new problem is well-defined in $I_X$ because we have assumed that the estimated range of each $I_{x_i}$ stays in the range of well-posedness allowed by the governing equation. Since problem (3.1) and (3.12) are exactly the same in the common domain $I^o$, we have the following trivial result

\[
u(\cdot,\xi) = v(\cdot,\xi), \quad \forall \xi \in I^o.
(3.13)
\]

We remark that for the encapsulation problem (3.12) we do not assign any probability information to variables $X$.

3.2.3 Solution for the Encapsulation Problem

For solution of the encapsulation problem (3.12), we again focus only on the dependence on the variables $X$, which now resides in the hypercube $I_X \subseteq \mathbb{R}^d$. For any fixed location $x$ and time $t$,

\[
u(X) : I_X \to \mathbb{R}.
(3.14)
\]

A critical requirement for the proposed methodology is the need for the numerical approximation of (3.12) to converge pointwise. Let $u_n(X)$ be a numerical solution, where the index $n$ is associated with discretization parameters used in the approximation. We then require

\[
\epsilon_n \Delta \|u - u_n\|_{L^\infty(I_X)} \to 0, \quad n \to \infty,
(3.15)
\]

where $\| \cdot \|_{\infty}$ is the standard $L^\infty$ norm defined as

\[
\|u\|_{L^\infty(I_X)} = \sup_{X \in I_X} |u(X)|.
\]
When $u$ is sufficiently smooth, such kind of pointwise convergent approximation can be obtained, at least in principle.

While there are choices for solving the encapsulation problem (3.12), here we will focus on polynomial approximation $u_n$, where the index $n$ is typically associated with the highest degree of polynomials used in the approximation. This approach is a direct extension of polynomial approximation theory and the generalized polynomial chaos (gPC) and stochastic collocation (SC) methods used primarily for aleatory uncertainty analysis. Again we emphasize that the key to choosing a particular method is that, despite its computational efficiency, it should provide an accurate approximation in the $L^\infty$ norm of (3.15).

Without loss of generality and merely for notational convenience, hereafter we assume the encapsulation set $I_X$ is a hypercube

$$I_X = [-1, 1]^d, \quad d \geq 1. \quad (3.16)$$

Note this can always be accomplished by properly translating and scaling of the variables $X$ in (3.12).

**Collocation Approach**

The stochastic collocation method, presented in Chapter 2 can be used to decouple the solution $u(X)$, to the encapsulation problem (3.12), in the parameter space $I_X$. Subsequently we can solve (3.12) at a set of discrete nodes and then construct a polynomial approximation of $u$ that interpolates the solution at each node. This collocation approach has been used extensively to quantify aleatory uncertainty [9, 129, 127]

Let $\Theta_n = \{X_j\}_{j=1}^m \subset I_X$ be a set of (prescribed) nodes, where $m \geq 1$ is the number of nodes. By adopting the collocation methodology, we enforce (3.12) at the node $X_j, j = 1, \ldots, m$, and solve

$$\begin{cases}
  u_t(x, t, X_j) = \mathcal{L}(u), & D \times (0, T], \\
  B(u) = 0, & \partial D \times [0, T], \\
  u = u_0, & D \times \{t = 0}\end{cases} \quad (3.17)$$

It is easy to see that for each $j$, (3.17) is a deterministic problem with fixed values of $X$. Therefore, solving the system poses no difficulty provided one has a well established deterministic algorithm.

Let $u_j = u(\cdot, X_j), j = 1, \ldots, m$, be the solution of the above problem and $\{u_j\}_{j=1}$ be the ensemble of solutions obtained by solving (3.17). Through use of
3.2. METHODOLOGY

the solution ensemble, we then seek to construct \( u_n(X) \in \Pi(X) \), where \( \Pi(X) \) is a proper polynomial space, so that the convergence property (3.15) can be achieved.

While the general strategy is straightforward, the options for practical implementation are limited. Multivariate approximation is a challenging area with many open issues. Here we describe a more established method based on sparse grid interpolation [11], which has been used extensively in quantifying aleatory uncertainty following the work of [129]. Sparse grids are presented in detail in Chapter 4.

Sparse grid interpolation, is based upon a combination of one-dimensional interpolation formulae. Let \( \Theta_i^1 = X_{i,1}, \ldots, X_{i,m_i} \) be a set of distinct nodes in the direction \( X_i \) and \( \{u(X_{i,j})\}_{j=1}^{m_i} \) the numerical solution at these nodes. We can approximate the one dimensional component of the solution \( u \) over the range of \( X_i \) using the following interpolation formula

\[
U_i[u] = \sum_{j=1}^{m_i} u(X_{i,j}) \cdot \Psi_{i,j}(X_i)
\]  

(3.18)

where \( m_i \) is the number of collocation nodes and \( \Psi_{i,j} \) is the interpolating basis which satisfies the discrete orthogonality property \( \langle \Psi_{i,j}, \Psi_{i,k} \rangle = \delta_{jk} \). Lagrange polynomials and piecewise-linear functions are two commonly used bases.

In the multivariate case \( d > 1 \) we can approximate \( u \) by the \( N \)th-level Smolyak formula ([105])

\[
U_N = \sum_{N-d+1 \leq ||i|| \leq N} (-1)^{N-||i||} \cdot \binom{d-1}{N-||i||} \cdot (U_{i_1} \otimes \cdots \otimes U_{i_d})
\]  

(3.19)

where \( i = (i_1, \ldots, i_d) \) and \( ||i|| = i_1 + \cdots + i_d \). See [124] for detailed derivation of the formula. To compute the interpolating solution

\[ u_N(X) = U_N[u], \]

one only needs to evaluate the function \( u \) on the sparse grid,

\[ H_{N,d} = \bigcup_{N-d+1 \leq ||i|| \leq N} (\Theta_{i_1}^1 \times \cdots \times \Theta_{i_d}^1). \]  

(3.20)

To achieve better efficiency, the nodal sets should be nested so that \( \Theta_i \subset \Theta_{i+1} \) and \( H_{N,d} \subset H_{N+1,d} \). This means that to increase the level of interpolation from \( N \) to \( N+1 \) we need only solve (3.12) on the new set of points \( H_{N+1,d} \setminus H_{N,d} \).
Unlike a full tensor product construction, which suffers from the curse of dimensionality in that the number of nodes grows exponentially with the dimension $d$, the number of nodes required by the Smolyak formula only grows logarithmically with $d$.

Different sparse grid interpolations can be constructed based on the choice of one-dimensional interpolation (3.18). One popular choice is Clenshaw-Curtis interpolation, which utilizes the Lagrange polynomial basis defined on the extrema of the Chebyshev polynomials. For any choice of $m_i$, $1 \leq i \leq d$, these nodes are given by

$$X_{i,j} = -\cos\left(\frac{\pi(j - 1)}{m_i - 1}\right), \quad j = 1, \ldots, m_i$$

To ensure the nodal sets are nested, we choose

$$m_1 = 1 \quad \text{and} \quad m_i = 2^{i-1} + 1, \quad \text{for} \ i > 1$$

There have been extensive studies on the approximation properties of sparse grids, particularly those based on Clenshaw-Curtis abscissas. Here we cite one of the early results from [11]. For functions in space

$$F_d^\ell = \{ f : [-1, 1]^d \to \mathbb{R} | \partial^{[\ell]} f \text{ continuous, } i_j \leq \ell, \forall j \},$$

with norm

$$\|f\| = \max\{\|D^\alpha f\|_\infty, \alpha \in \mathbb{N}_0^d, \alpha_i \leq \ell\},$$

the interpolation error follows

$$\|I - U_N\| \leq C_d,\ell m^{-\ell}(\log m)^{(\ell + 2)(d-1)} + 1,$$ (3.22)

where $m$ is the total number of nodes required by the sparse grid interpolation $H_{N,d}$. (Note there is in general no explicit formula for $m$.)

When quantifying aleatory uncertainty the Clenshaw-Curtis sparse grids are only appropriate when the underlying random variables possess uniform distributions. However when quantifying epistemic uncertainty this requirement can be removed. The Clenshaw-Curtis grid may not be optimal for the “true” unknown distribution, however the resulting approximation will still exhibit the required point-wise convergence, albeit at a slower rate. Subsequently Clenshaw-Curtis sparse grid interpolation, or any sparse grid interpolation for that matter, is certainly not the only choice for the collocation approach. In practice, any valid interpolation approach can be employed, so long as one can establish its convergence in the pointwise sense of (3.15).
3.2. METHODOLOGY

Galerkin Approach

We briefly remark that (3.12) can also be solved by the Galerkin approach. In the Galerkin approach, we seek a numerical solution \( u_n(X) \) in a polynomial space such that the residual of (3.12) is orthogonal to the polynomial space. While most of the convergence of the Galerkin solution is in the weighted \( L^p \) norm on \( I_X \), it is possible to have the solution converge pointwise uniformly, which is what we require. This usually imposes stronger smoothness conditions on the true solution \( u \). For example, for the stochastic diffusion equation with linear form random diffusivity, it has shown that the solution is analytic in terms of the random inputs [9], and a numerical solution converging in pointwise sense can be used for sampling non-Gaussian processes [119]. Since it is not possible to discuss the convergence of the Galerkin approach without specifying the form of (3.12), we will not engage in more discussions on this.

We also remark that there exists a pseudo-spectral collocation method [127], also known as the non-intrusive gPC method. Though this method is of collocation type, its convergence is usually in \( L^p \) norm, similar to Galerkin. Therefore it is not possible to discuss its \( L_\infty \) error without specifying the underlying governing equations and we will not focus on this method either.

Piecewise Smooth Case

In the discussions above we have required the solution to converge in \( L_\infty \)-norm in the entire domain \( I_X \). This requires sufficient global smoothness of \( u \), which is rather strong in many practical problems. In fact, the discussions can be generalized to piecewise smooth functions of \( u \). That is, there exists a finite decomposition of \( I_X \), \( k = 1, \ldots, m \), such that

\[
\bigcup_{k=1}^m I_k = I_X, \quad I_k \cap I_j = \emptyset, \quad i \neq j,
\]

and in each sub-domain \( I_k \), \( k = 1, \ldots, m \), \( u(X) \) is smooth.

In this case, a suitable numerical approach in the global sense (e.g., the sparse grid interpolation) can be applied to each sub-domain separately and obtain a convergent solution (in the \( L_\infty \)-norm) \( u_k^e(X) \) in each sub-domain \( I_k^e \). A globally convergent solution can then be constructed by “patching” the sub-domain solutions together, viz.

\[
u_n(X) = \sum_{k=1}^m u_k^e(X) 1_{I_k^e}(X), \quad (3.23)
\]
where \( I_A(s) = 1 \) if \( s \in A \); \( I_A(s) = 0 \) otherwise, is the indicator function. It is easy to see that this solution will converge to \( u \) in the entire domain of \( I_X \) in the \( L_\infty \)-norm. Note that due to the nature of the problem (3.12), there is no continuity requirement of the solution across the sub-domain interfaces. Therefore, at least on the conceptual level, solving the sub-domain problem is straightforward. From the practical point of view, the multi-element, or piecewise, approximation techniques developed for aleatory uncertainty can be borrowed. These include the work of [10, 32, 68, 76, 117]. Hereafter we will restrict ourselves to globally smooth problems to emphasize the new conceptual ideas related to epistemic uncertainty quantification.

### 3.2.4 Epistemic Uncertainty Analysis

When \( u_n(X) \), the polynomial approximation of the true solution \( u(X) \), is obtained for (3.12) and converges in the \( \infty \)-norm (3.15), it can serve as an accurate and pointwise model. We can then apply various operations on \( u_n \) instead of \( u \). Note the operations on \( u_n \) do not require us to solve the governing equations anymore—they can be treated as post-processing steps.

Assuming information about the distribution of the random inputs \( \xi \) is known \textit{a posteriori}, then we can evaluate the solution statistics by using the \( u_n \). This can be achieved by evaluating the statistics of \( u_n \) using the probability of \( \xi \) in the domain \( I^o \) defined in (3.9). Let \( \rho_\xi(s) = dF_\xi(s) \), \( s \in I_\xi \), be the probability density function of the epistemic uncertain input \( \xi \), which was not known prior to the computations but is now known after the computations. Then, for example, the mean of the true solution \( v(\xi) \),

\[
\mu \triangleq \mathbb{E}[(v)(\xi)] = \int_{I_\xi} v(s)\rho_\xi(s)ds, \tag{3.24}
\]

can be approximated by

\[
\mu_n \triangleq \int_{I^o} u_n(s)\rho_\xi(s)ds. \tag{3.25}
\]

The following result can be established

\textbf{Theorem 3.1.} Assume the solution of (3.1), \( v(\xi) \), is bounded and let \( C_v = \|v\|_{L_\infty(I_\xi)} \). Let \( u_n(X) \) be an approximation to the solution \( u(X) \) of (3.12) and converge in the form of (3.15) and denote

\[
\epsilon_n = \|u_n(X) - u(X)\|_{L_\infty(I_X)}. \tag{3.26}
\]
3.2. METHODOLOGY

Then the mean of \( v \) in (3.24) and the mean of \( u_n \) in (3.25) satisfy

\[
|\mu - \mu_n| \leq \epsilon_n + C_v \delta. \tag{3.27}
\]

**Proof.** We first extend the domain of definition of \( v, \rho, \) and \( u_n \) to \( I^+ \), following the definitions of the domains in (3.9), and define

\[
\rho^+(s) = \mathbb{I}_{I^+}(s)\rho(s), \quad v^+(s) = \mathbb{I}_{I^+}(s)v(s), \quad s \in I^+,
\]

and

\[
u_n^+(s) = \mathbb{I}_{I^+}(s)u_n(s), \quad s \in I^+.
\]

Naturally, \( \rho^+ \) is a probability density function on \( I^+ \). Then (3.24) can be expressed as

\[
\mu = \int_{I^+} v(s)\rho(s)ds = \int_{I^+} v^+(s)\rho^+(s)ds,
\]

which can be split into two parts

\[
\mu = \int_{I^+} v^+(s)\rho^+(s)ds + \int_{I^+} v^+(s)\rho^+(s)ds
\]

\[
= \int_{I^+} v(s)\rho(s)ds + \int_{I^+} v^+(s)\rho^+(s)ds. \tag{3.28}
\]

By using (3.25), we have

\[
\mu - \mu_n = \int_{I^+} (v(s) - u_n(s))\rho(s)ds + \int_{I^+} v(s)\rho^+(s)ds
\]

\[
= \int_{I^+} (u(s) - u_n(s))\rho(s)ds + \int_{I^+} v(s)\rho^+(s)ds, \tag{3.29}
\]

where the property (3.13) has been used. Utilizing the condition (3.11), the main result (3.27) is established. \( \square \)

The second term in (3.29), \( C_v \delta \), is a result of truncating the “tails” of the true probability distribution of \( \xi \) by using a bounded hypercube \( I_X \) to encapsulate a possibly unbounded domain \( I_\xi \). For unbounded domains, for which \( I^- \cap I_\xi \neq \emptyset \), the term \( C_v \delta \) can be made arbitrarily small by choosing a sufficiently large hypercube for \( I_X \). If \( \xi \) is in a bounded domain and the domain \( I_X \) can completely encapsulate \( I_\xi \) then the term \( C_v \delta \) disappears. In this case, the error in the mean would converge to zero as long as \( u_n \) converges in the form of (3.15).

We remark that the encapsulation problem can also be used to obtain upper and lower bounds on the model observables. Consequently solving the encapsulation problem can be used as an alternative to techniques such as interval analysis [83]. The accuracy of these estimates is beyond the scope of this chapter and is left for future work.
3.2.5 Mixed Aleatory–Epistemic Uncertainty Analysis

In practice, situations may arise for which the distributions of some of the random variables characterizing the input are known and some are not. The encapsulation methodology proposed here can easily be extended to such cases possessing mixed aleatory and epistemic uncertainty. Let us consider stochastic differential equations with the form

\[
\begin{align*}
\mathcal{L}(v) &= D \times (0, T) \times I_Y \times I_\xi, \\
B(v) &= 0, \\
v &= v_0,
\end{align*}
\]

where \( Y \) is a set of random variables with known probability distribution \( F_Y(y) = P(Y \leq y), y \in I_Y \subseteq \mathbb{R}^r, r \geq 1, \) and \( \xi \in \mathbb{R}^s \) are a set of random variables with unknown distribution.

As in the purely epistemic case we first begin by defining and solving an encapsulation problem. With this aim we again define the encapsulation set \( I_X \) according to (3.8) which encapsulates \( I_\xi \), the “true” and unknown support of \( \xi \), with probability at least \( 1 - \delta \). The encapsulation problem is then

\[
\begin{align*}
\mathcal{L}(u) &= D \times (0, T) \times I_Y \times I_X, \\
B(u) &= 0, \\
u &= u_0,
\end{align*}
\]

Unlike in the purely epistemic case, the encapsulation problem is now defined in terms of the epistemic and aleatory variables. This encapsulation problem can be solved in two ways depending on whether one wants to solve the epistemic and aleatory problems separately or simultaneously:

- Separate construction. Different methods can be employed to quantify the epistemic \( u(\cdot, X) \) and aleatory uncertainty \( u(\cdot, Y) \). Let \( \hat{u}_m(X) \) be an approximation to \( u(\cdot, X) \) after fixing all variables other than \( X \), and let \( \tilde{u}_k(Y) \) be an approximation to \( u(\cdot, Y) \), where the indices \( m \) and \( k \) denoting the level of approximations. Then \( u(X, Y) \) can be approximated by a tensor product of \( \hat{u}_m(X) \) and \( \tilde{u}_k(Y) \). That is,

\[
u_n(X, Y) = \hat{u}_m(X) \otimes \tilde{u}_k(Y),\]

where the index \( n \) depends on \( m \) and \( k \). (In the case of polynomial approximation, \( n \) can be either the highest polynomial order in \( \hat{u}_m \) and \( \tilde{u}_k \)
or the total order of the mixed polynomials of \( \hat{u}_m \) and \( \hat{u}_k \).) The construction allows us to use different methods for \( X \) and \( Y \). For example, we can mix an accurate collocation solution \( \hat{u}_m(X) \) converging in \( L^\infty \) norm in the epistemic variable \( X \) with an accurate stochastic Galerkin solution \( \hat{u}_k(Y) \) converging in \( L^2_{\rho_Y} \) norm in the aleatory variable \( Y \).

- Simultaneous construction. Instead of treating the epistemic and aleatory variables separately, we can consider the aleatory variables as epistemic and solve the epistemic encapsulation problem (3.12), with \( I_X \) defined in such a way that it encapsulates both \( I_\xi \) and \( I_Y \). For \( r \) aleatory variables \( Y \) and \( s \) epistemic variables \( \xi \), we define

\[
I_X = (\times_{i=1}^r I_{Y_i}) \times (\times_{i=1}^s I_{X_i}),
\]

where \( I_{X_i} \) are bounded intervals that encapsulate \( I_{\xi_i} \) with overwhelming probability. The same methods for the epistemic encapsulation problem can be used to generate an approximation in \( I_X \). The probabilistic information associated with the aleatory variables \( Y \) can be introduced in post processing. Additional probabilistic information for \( \xi \) can be processed \textit{a posteriori} when known. We remark that this approach requires pointwise accuracy in the entire space (3.32). This may be too strong because, in general, accuracy in the mean square sense in the aleatory variables \( Y \) is sufficient. Pointwise accuracy is particularly hard to achieve when the aleatory variables are unbounded. In this case we may need to truncate the domain of the aleatory variables and this leads to additional "truncation" error. Therefore, the simultaneous approach is more appropriate when all the variables are bounded.

After taking into account the probability distribution of the aleatory random variables \( Y \), the solution becomes a function of the epistemic variable \( \xi \). For example, the mean solution is

\[
\mu(s) = \mathbb{E}_Y[v(Y, \xi)] = \int v(y, s) dF_Y(y), \quad s \in I_\xi.
\]

This can be approximated by

\[
\mu_n(s) = \mathbb{E}_Y[u_n(Y, X)] = \int u_n(y, s) dF_Y(y), \quad s \in I_X.
\]

Note that no probability information is assigned to the variables \( X \) and \( \xi \) prior to any computations.
3.3 Numerical Examples

In this section we provide several numerical tests to illustrate the implementation and convergence of the proposed methodology. In all examples, we first seek polynomial approximation to the solutions in terms of the epistemic variables. We then, in post-processing steps, assume certain probability distribution information is known \textit{a posteriori}, and evaluate the solution statistics of the true solution and the numerical approximations in order to examine the accuracy of the methods. In all examples, we utilize global polynomial approximations.

3.3.1 Ordinary Differential Equation

Consider

\[
\frac{dv}{dt}(t) = -\xi_1 v, \quad v(0) = \xi_2, \tag{3.33}
\]

where the parameters \(\xi_1\) and \(\xi_2\) are random variables representing the input uncertainty. The exact solution is

\[
v(t, \xi) = \xi_2 \exp(-\xi_1 t). \tag{3.34}
\]

Let us assume that the distributions (and dependence) of \(\xi_1\) and \(\xi_2\) are unknown, except that the bounds of the parameters can be estimated with a range that is sufficiently wide.

The encapsulation problem is

\[
\dot{u}(t, X) = -X_1 u, \quad u(0) = X_2, \tag{3.35}
\]

where \(X = (X_1, X_2) \in [-1, 1]^2\) after scaling.

Here we use the Galerkin method based on Legendre polynomials to solve (3.35). This implies the numerical solution will converge in the \(L^2\) norm. However, since the solution is analytic, pointwise convergence can also be achieved. For comparison, we also present results using a sparse grid approximation of \(u\) based upon the tensor product of Lagrange polynomials defined at the Clenshaw-Curtis abscissas. Both methods provide fast converging polynomial approximations of the solution. We illustrate the convergence of the mean and variance of these approximations when the marginal and joint probability distributions are found \textit{a posteriori} below.
3.3. NUMERICAL EXAMPLES

Independent case

Let us assume the "true" (and yet unknown) distributions of the components of $\xi$ are $\xi_1 \sim \beta(0,1, \alpha_1, \beta_1)$ and $\xi_2 \sim \beta(0,1, \alpha_2, \beta_2)$, where $\xi_1$ and $\xi_2$ are independent and $\beta(a,b,\alpha,\beta)$ is the beta distribution on the interval $[a,b]$ with distribution parameters $\alpha$ and $\beta$. Analytical expressions for the moments of $v$ exist and can be used to test convergence of our numerical solutions of the encapsulation problem in $X$. Note in this case $I_{\xi} = [0,1] \times [0,1]$ and is completely encapsulated by $I_X = [-1,1]^2$.

The moments of the numerical approximations were obtained by using multidimensional tensor product Gauss-Jacobi quadrature. Specifically an appropriate high-order one-dimensional quadrature rule, determined by the now known distribution of $\xi$, was chosen for each independent variable. Then a tensor product of these rules was used to construct a set of multi-dimensional quadrature nodes and associated weights. The order of the quadrature rule was chosen to match the order of the approximating polynomial. Sampling the polynomial expansion at the quadrature nodes is a post-processing process and only requires the evaluation of algebraic expressions, and is thus inexpensive compared to the cost of evaluating the true model.

In Figure 3.1 the relative error in the first two moments are shown for varying values of $\alpha_i$ and $\beta_i$. Here and throughout the remainder of this chapter relative error is defined to be the absolute difference between the approximate and true value normalized by the true value. As the order of the Legendre-Galerkin polynomial expansion and the approximation level of the collocation sparse grid increases, the errors converge exponentially fast before reaching saturation levels.

Dependent case

Let us assume the "true" (and yet unknown) distributions of the components of $\xi$ are $\xi_1 \sim \beta(0,1, \alpha_1, \beta_1)$, $\xi_2$ is dependent on $\xi_1$. Specifically let us assume that $\xi_2 = \xi_1$. This implies that $I_{\xi} = [0,1]$ and can be entirely encapsulated by $I_X = [-1,1]^2$. Again analytical expressions for the moments of $v$ exist and can be used to test convergence of our numerical approximations, whose moments were obtained by selecting an appropriate high-order one-dimensional quadrature rule, determined by the now known distribution of $\xi$, for the variables $\xi_1$. This one-dimensional quadrature rule was then used to evaluate the moments of the approximations along the line $\xi_1 = \xi_2$.

Figure 3.2 plots the error in the first two moments for varying values of $\alpha_i$.
CHAPTER 3. QUANTIFYING EPISTEMIC UNCERTAINTY

Figure 3.1: Convergence of the relative error in the mean and variance for the linear ODE with two dimensional ($d = 2$) independent input with varying a posteriori distributions, $\xi_1, \xi_2 \in \text{beta}(0, 1, 1, 1)$ and $\xi_1 \in \text{beta}(0, 1, 2, 5)$ and $\xi_2 \in \text{beta}(0, 1, 1, 1)$ . (a) Convergence with Galerkin polynomial order. (b) Convergence with collocation sparse grid level.

and $\beta_i$. As the approximation level of the collocation sparse grid increases the errors converge exponentially fast before reaching saturation levels.

Choice of polynomial basis

In sections 3.3.1 and 3.3.1, multi-dimensional Lagrange and Legendre polynomials were used to produce an approximation to the solution of the encapsulation problem. However any type of polynomial that satisfies (3.15) can be used.

It was shown in [130] that, in the context of aleatory uncertainty quantification, if the polynomial basis used to approximate a stochastic solution is chosen according to the distribution of the underlying random variables, better approximation accuracy can be achieved. If the optimal basis is not chosen, the rate of convergence will deteriorate. Here we investigate the effect of the choice of the approximating polynomial on the convergence of the mean and variance of solutions subject to epistemic uncertainty.

Let us assume the “true” (and yet unknown) distributions of $\xi_i$ are independent. Figure 3.3 shows the rate of convergence in the estimates of variance for various types of polynomial approximations of the encapsulation problem (3.35). When the optimal polynomial basis is used, estimates of the variance are obtained directly from the basis coefficients. The variance of the (non-optimal) Legendre approximation was calculated using a high-order two-dimensional tensor-product Gauss-Jacobi quadrature rule.
Figure 3.2: Convergence of the relative error in the mean and variance for the linear ODE with two dimensional \(d = 2\) independent input with varying \textit{a posteriori} distributions, \(\xi_1, \in \text{beta}(0, 1, 1, 1)\) and \(\xi_1, \in \text{beta}(0, 1, 2, 5)\). In all cases \(\xi_2 = \xi_1\). (a) Convergence with Galerkin polynomial order. (b) Convergence with collocation sparse grid level.

When the type of polynomial expansion is chosen to match the distribution of the input variables, a faster rate of convergence is obtained than if another type was chosen. The nature of epistemic uncertainty means that an optimal basis cannot be chosen \textit{a priori} and some accuracy penalty may have to be paid due to the lack of full probabilistic information at the time of computing the expansion. If, by coincidence, the basis chosen to approximate the encapsulation problem matches the weighting functions of the underlying random variables, then the optimal convergence rate will be achieved. In most cases, however, we must select a basis that provides a reasonable compromise given the information available; e.g., if only bounds are provided and there is no justification to weight errors unequally within these bounds, then a Legendre basis is the natural choice.

### 3.3.2 Random Oscillator

This section investigates the performance of stochastic collocation to quantify epistemic uncertainty in a damped linear oscillator subject to external forcing with six unknown parameters. That is,

\[
\frac{d^2x}{dt^2}(t, \xi) + \gamma \frac{dx}{dt} + kx = f \cos(\omega t),
\]

subject to the initial conditions

\[
x(0) = x_0, \quad \dot{x}(0) = x_1,
\]
CHAPTER 3. QUANTIFYING EPISTEMIC UNCERTAINTY

Figure 3.3: Convergence of the relative error in the variance for the linear ODE with two independent input variables. Convergence is shown with respect to the order of the gPC expansion or two choices of stochastic polynomial expansions and for two different input distributions; $\xi_1, \xi_2 \in \text{beta}(-1,1,1,1)$ (solid lines) and $\xi_1, \xi_2 \in \text{beta}(-1,1,2,2)$ (dashed lines).

where we assume the damping coefficient $\gamma$, spring constant $k$, forcing amplitude $f$ and frequency $\omega$, and the initial conditions $x_0$ and $x_1$ are all uncertain, and let

$$\xi = (\gamma, k, f, \omega, x_0, x_1) \in \mathbb{R}^6$$

be the epistemic variables.

The encapsulation problem is then

$$\frac{d^2 x}{dt^2} (t, X) + X_1 \frac{dx}{dt} + X_3 x = X_3 \cos(X_4 t) \quad \text{(3.38)}$$

$$x(0) = X_5, \quad \dot{x}(0) = X_6, \quad \text{(3.39)}$$

where $X = (X_1, \ldots, X_6) \in [-1, 1]^6$ (upon scaling) are the encapsulation variables. We employ sparse grid Lagrange interpolation at the Clenshaw-Curtis abscissas to solve the encapsulation problem.

Epistemic uncertainty with dependent inputs

To again illustrate the convergence of moments, let us assume the "true" (and yet unknown) distribution of $\xi$ is $\xi_i \in \text{beta}(a_i, b_i, \alpha_i, \beta_i)$, $i = 1, 3, 5$ and $\xi_2$ is
dependent on $\xi_1$, $\xi_4$ is dependent on $\xi_3$ and $\xi_6$ is dependent on $\xi_5$. For example, let us assume $\xi_1 \sim \text{beta}(0.08, 0.12, 3, 2)$ and $\xi_2 = \frac{\xi_1^2}{4} + 0.01$, $\xi_3 \sim \text{beta}(0.08, 0.12, 1, 1)$ and $\xi_4 = 10\xi_3$, and $\xi_5 \sim \text{uniform}(0.45, 0.55)$ and $\xi_6 = (\xi_5 - 0.5)$.

Moments can be evaluated by collapsing the expansion in $X$ by substitution based upon the known functional dependence and then applying a lower-dimensional quadrature rule. Here we employed a three-dimensional quadrature rule based upon the tensor product of one-dimensional rules for $\xi_1$, $\xi_3$ and $\xi_5$. Values for the remaining variables were selected based upon the functional dependence specified above.

Figure 3.4 plots the error in the first two moments at $t = 20$. As the order of the approximation level of the sparse grid increases the errors converge exponentially fast before reaching saturation levels.

![Figure 3.4: Convergence of the relative error in the mean and variance for the damped harmonic oscillator with six dependent inputs. Convergence is shown with respect to the approximation level of the SC sparse grid.](image)

**Epistemic uncertainty with known covariance**

In practice one may often encounter uncertainty arising from a set of random variables with normally distributed marginal distributions and known covariance. Consider $X = (X_1, \ldots, X_6) \sim N(0, C)$ where the covariance matrix is a tri-diagonal matrix with non-zero entries $\sigma_{11} = 0.03$, $\sigma_{22} = 0.0009$, $\sigma_{33} =$
0.0003, $\sigma_{44} = 0.01$, $\sigma_{55} = 0.001$, $\sigma_{66} = 0.0025$, $\sigma_{12} = 0.05\sigma_{11}$, $\sigma_{21} = \sigma_{22}$, $\sigma_{34} = 0.02\sigma_{44}$, $\sigma_{43} = 0.2\sigma_{44}$, and $\sigma_{56} = \sigma_{65} = 0.1\sigma_{55}$.

Unlike the previous examples the epistemic variables are now unbounded. Consequently we must construct an approximation to the encapsulation problem which captures the true input range with overwhelming probability. Here we investigate the choice of size of the bounding hyper-region on the accuracy of the solution moments. Figure 3.5 plots the error in the first two moments at $t = 20$. As the approximation level of the collocation sparse grid increases the errors converge exponentially fast before reaching saturation levels. However the accuracy at which saturation occurs is dependent on how “well” the input space is encapsulated. As the encapsulation probability increases, that is as $\delta$ decreases, the best possible accuracy that can be obtained by solving the encapsulation problem increases. It must be noted that the convergence rate slows with decreasing $\delta$, because it increases the size of the encapsulation domain. In general interpolation of a larger domain requires more evaluations of the governing equations to achieve a comparable accuracy.

The exact moments of the solution were obtained by applying high order six-dimensional Gauss-Hermite sparse grid quadrature to the governing equations. Moments of the SC approximation were obtained by applying the same quadrature rule to the numerical solution of the encapsulation problem. The Gauss-Hermite sparse grid quadrature assumes independent Gaussian variables. A Cholesky decomposition of the covariance matrix was used to generate a set of dependent realizations of $\xi$.

**Mixed aleatory-epistemic uncertainty**

Now let us consider the uncertainty of the solution to (3.36) where the distributions of some of the variables are known and the distributions of other variables are unknown.

A simple two-step iterative procedure can be used to generate such an ensemble of statistics. In this case we wish to generate an ensemble of CDFs of the solution to the governing equations at time $t = 20$. In the first step of each iteration a particular value of each epistemic variable is chosen from within their assumed ranges. Fixing these values we then randomly sample from the aleatory variables in a standard probabilistic manner. These samples are then used to evaluate the polynomial approximation of the encapsulation problem. Following this heuristic, each set of epistemic variables generates a full distributional de-
3.3. NUMERICAL EXAMPLES

3.3.3 Homogeneous Diffusion Equation

In this section we consider the homogeneous diffusion equation in one-spatial dimension subject to epistemic uncertainty in the diffusivity coefficient. Attention
is restricted to the one-dimensional physical space to avoid unnecessary complexity. The procedure described here can easily be extended to higher physical dimensions. Consider the following problem with \( d \geq 1 \) random dimensions:

\[
\frac{d}{dx} \left[ a(x, \xi) \frac{du}{dx}(x, \xi) \right] = 0, \quad (x, \xi) \in (0, 1) \times I_\xi
\]  

subject to the physical boundary conditions

\[
u(0) = 0, \quad u(1) = 0
\]

Furthermore assume that the random diffusivity satisfies

\[
a(x, \xi) = 1 + \sigma \sum_{k=1}^{d} \frac{1}{k^2 \pi^2} \cos(2\pi k x) \xi_k
\]

where \( \xi_k \in [-1, 1] \), \( k = 1, \ldots, d \) are independent and uniformly distributed random variables. The form of (3.42) is similar to that obtained from a Karhunen-Loève expansion and satisfies the auxiliary properties

\[
\mathbb{E}[a(x, \xi)] = 1 \quad \text{and} \quad 1 - \frac{\sigma}{6} < a(x, \xi) < 1 + \frac{\sigma}{6}.
\]
3.3. NUMERICAL EXAMPLES

This is the same test case used in [129].

Again we construct an appropriate encapsulation problem and solve it using the Legendre-Galerkin method. Specifically we employ the efficient spectral Galerkin iterative solver discussed in [131]. A high spatial resolution is used to ensure that the associated errors can be neglected in the following analysis.

Whereas previous discussions have focused on singular parameter dependence, here we investigate the performance of the proposed method for multi-parameter dependence. Let us assume the "true" (and yet unknown) distribution of $\xi$ is $\xi_1 \sim \text{beta}(0, 1, 3, 2)$, and $\xi_3 \sim \text{beta}(-1, 0, 1, 1)$ and $\xi_5 \sim \text{beta}(-0.5, 0.5, 0, 0)$, $\xi_2 = \xi_1 \xi_5$, $\xi_4 = (\xi_1^2 + 1)\xi_3$, and $\xi_6 = -\xi_5$ and $\sigma = 4$. The convergence of the mean and variance of the encapsulation problem at $x = 0.848925247397$ are shown in Figure 3.7. Despite the non-linear dependence between the random variables the fast rate of convergence is still maintained. No analytical solution is available so convergence is analyzed against a high order Legendre-Galerkin approximation of the solution.

![Figure 3.7: Convergence of the relative error in the mean and variance for the homogeneous diffusion equation with six dependent inputs. Convergence is with respect to the order of the $\text{gPC}$ expansion, at the spatial location $x = 0.848925247397$.](image-url)
3.4 Conclusions

In this chapter we proposed a framework for quantifying epistemic uncertainty. The methodology presented is a generalization of traditional aleatory uncertainty quantification that allows one to seamlessly switch between epistemic, aleatory and mixed epistemic-aleatory uncertainty analysis. The validity and effectiveness of our approaches is illustrated through several examples.

The approach is based on solution of an "encapsulation problem" which generates a solution to the governing equations in a domain that encloses the true probability space with overwhelming probability. No distributional information about any of the variables needs be assumed, only estimates of the ranges of the variables are needed. Once the bounds have been specified, a polynomial approximation can be constructed in the encapsulation domain. The polynomial approximation to the encapsulation problem is chosen to converge pointwise throughout the input space. Thus it is also accurate on a subset of this space. As long as this pointwise convergence is obtained, the polynomial solution of the encapsulation problem can be used as an effective model for the true solution on the true domain.

Once the polynomial approximation within the encapsulation domain has been computed, it can be employed within epistemic analyses such as interval analysis or evidence theory, within sensitivity analysis studies to explore the importance of epistemic parameters and allocate experimental resources, and ultimately within a posteriori aleatory analyses following collection of additional experimental data. In this chapter, we focus on the a posteriori evaluation of solution statistics and demonstrate convergence of statistics following the introduction of additional information on functional dependence or correlation, distributional form, or both.

The encapsulation approach can readily handle dependencies between input variables. If the functional dependence becomes known a posteriori, then the relationships can be used to collapse the dimensionality of the polynomial approximation and quadrature methods can then be applied on this lower-dimensional space to obtain estimates of moments. We show that exponential convergence can be obtained for correlated normals by utilizing a Cholesky decomposition to map a set of independent variables (needed to construct the polynomial approximation) to a set of dependent variables. Although not presented, such a procedure could be extended to correlated non-normals provided a variable transformation exists, for example the Nataf transformation.

If the distributional form of the epistemic variables becomes known a posteriori-
ori, then solution statistics can be evaluated as a post-processing step. While the polynomial basis selected \textit{a priori} will not in general be optimal for this \textit{a posteriori} post-processing, this suboptimal weighting of polynomial approximation errors is an algorithmic cost that must be paid for having imperfect characterization of uncertainties at the time of approximation construction. The modeler can minimize this penalty by tailoring the basis to the available information to the extent possible, but in the case of a pure interval-based epistemic uncertainty description, there is no justification to weight errors unequally within the interval and a Legendre basis is the logical choice. It is demonstrated that exponentially-fast convergence rates can nonetheless be obtained for the \textit{a posteriori} solution statistics despite the lack of complete information at expansion construction time.
Chapter 4

$h$-Adaptive Sparse Grids

In Chapter 3 we identified sparse grids as a possible means of solving the encapsulation problem, presented in Section 3.2.2, which is used to quantify epistemic uncertainty. Sparse grids exhibit the required point-wise convergence needed to solve the encapsulation problem and can be applied to models with many unknown variables. In this chapter we introduce sparse grids in more detail and propose a new sparse grid method which can interpolate smooth and discontinuous functions in hundreds of dimensions.

Sparse grids have arisen as an efficient means for high-dimensional interpolation and quadrature. In recent years much attention has been given to the use of sparse grids for stochastic collocation [3, 32, 35, 76, 129]. Numerous constructions have been proposed from isotropic sparse grids, which treat all dimensions and regions equally [130], to dimension-adaptive [35] and $h$-adaptive approaches [32, 76] and combinations thereof [31, 78].

The $h$-adaptive method developed by Ma and Zabaras [76, 78] has proven to be an efficient means of approximating high-dimensional functions. Their self-titled Adaptive Sparse Grid Collocation (ASGC) method uses multivariate piecewise linear basis functions to interpolate high-dimensional functions. The refinement of the sparse grid is guided by the magnitude of the so-called hierarchical surplus, which is the difference between the true function and the approximation at a new grid point before that point is used in the interpolation. The resulting method automatically concentrates function evaluations in rapidly varying or discontinuous regions. This method is implicitly dimension adaptive, but often points are constructed unnecessarily in 'unimportant' dimensions. In comparison, the generalised sparse grid algorithm [41] provides an efficient means of identifying the effective dimensionality of a problem and restricting function evaluations to that
sub-dimensional space. This method performs extremely well when the solution is smooth. However the efficiency of the generalised sparse grid method can be significantly improved when only small regions of the input space contribute to the model’s variability.

In this chapter we combine $h$-refinement with the dimension-adaptive algorithm of the generalised sparse grid method to construct an efficient high-dimensional interpolation method. Furthermore we utilise the localised polynomial basis proposed by Bungartz [16] to create a higher-order method which achieves fast rates of convergence in smooth regions and accuracy, comparable to linear methods, around discontinuities. We coin this approach the $h$-Adaptive Generalised Sparse Grid (h-GSG) method. The convergence of the proposed method is analysed, with respect to the order of the local polynomial basis and the dimensionality of the input space. We also compare the $h$-GSG method to the generalised sparse grid method [41], and the dimension-adaptive variant of ASGC known as ASGC-HDMR [78] (first introduced in Section 1.2.3). The chapter is concluded by using $h$-GSG to construct a high-dimensional surrogate of a model simulating flow through porous media.

4.1 Sub-Space Splitting

Numerous constructions can be used for the trial space in the stochastic collocation variational problem (2.4). Sparse grids are one such construction. The exact form of the trial space is determined by the choice of the bases $\Psi$ and the associated abscissa $\xi$.

Before proceeding, let us first introduce some notation. Let $i \in \mathbb{N}_0^d$ denote a $d$-dimensional index with the two norms

$$|i|_\infty = \max_{0 \leq n \leq d} i_n \quad \text{and} \quad |i|_1 = \sum_{n=1}^d i_n$$

These indices satisfy the following component-wise relations

$$i \cdot j = (i_1 j_1, \ldots, i_d j_d), \quad \alpha \cdot i = (\alpha i_1, \ldots, \alpha i_d), \quad \alpha^i = (\alpha^{i_1}, \ldots, \alpha^{i_d})$$

for some $\alpha \in \mathbb{R}$ and the operators

$$i \leq j \iff i_n \leq j_n \forall 1 \leq n \leq d, \quad i < j \iff i_n < j_n \forall n \text{ and } \exists i_n < j_n$$

We also introduce the special multi-indices $0 = (0, \ldots, 0)$ and $1 = (1, \ldots, 1)$. 
The performance of these sparse grid spaces is dependent on the nature of the solution \( u(\xi) \). Letting

\[
D^1 u = \frac{\partial |\xi|^l u(\xi)}{\partial \xi_1 \cdots \partial \xi_d}
\]

we consider functions \( u \) belonging to the Sobolev spaces \( X^{p,q} = \{ u : \Gamma \to \mathbb{R} : \| u \|_{X^{p,q}} < \infty \} \), \( q \in \{2, \infty\} \)
defined by the norms

\[
\| u \|_{X^{p,q}} = \begin{cases} 
\left( \sum_{0 \leq l \leq p} |D^l u|^q \right)^{\frac{1}{q}} & 1 \leq q < \infty \\
\sum_{0 \leq l \leq p} |D^l u| & q = \infty
\end{cases}
\]

and the associated semi norms

\[
|u|_{l,q} = \| D^l u \|_q
\]

Attention is often restricted to functions \( u \in X_0^{p,q} \subset X^{p,q} \) which vanish on the boundary. In practice, however, such grids are generally not appropriate for uncertainty quantification where the behaviour of the solution \( u \) at the boundary is often important. Here we will discuss methods for approximating functions \( u \in X^{p,q} \) with heterogeneous boundary conditions.

Sparse grids are a direct sum of anisotropic grids \( \Omega_i \) on the domain \( \Gamma \) where \( i = (i_1, \ldots, i_d) \in \mathbb{N}^d \) is a multi-index denoting the level of refinement of the grid in each dimension \( d \). Each grid \( \Omega_i \) is a tensor product of one-dimensional grids

\[
\Omega_i = (\xi_{i_1}, \ldots, \xi_{i_d})
\]

where \( m_i \) is odd and represents the number of points in the \( i \)th level one-dimensional grid. Specifically

\[
\Omega_i = \bigotimes_{n=1}^d \Omega_{i_n}
\]

which consists of the points \( \xi_{i,j} = (\xi_{i_1,j_1}, \ldots, \xi_{i_d,j_d}) \) and where \( i \) indicates the level of refinement and \( j \) denotes the location of a given grid point. The exact coordinates of each point and the total number of points \( m_{i_1} \times \cdots \times m_{i_d} \) is dependent on the type of one-dimensional grids used.

Without loss of generality let us consider the \( d \)-dimensional unit hypercube \( \Gamma = [0, 1]^d \). The simplest construction of the trial space is based on the family of \( d \)-dimensional grids \( \Omega_i \) with mesh size

\[
h_i = (h_{i_1}, \ldots, h_{i_d}) = 2^{-i}, \quad 0 \leq i
\]
These grids are equidistant with respect to each individual coordinate direction but typically have different mesh sizes in each direction. The grid points within each grid $\Omega_i$ are

$$\xi_{i,j} = (\xi_{i,j_1}, \ldots, \xi_{i,j_d}) = j \cdot h_i, \quad 0 \leq j \leq m_i - 1 = 2^i \quad (4.2)$$

Each grid $\Omega_i$ is associated with a discrete approximation space and a set of basis functions that span the discrete space. The types of basis functions that can be used are dependent on the type of one-dimensional grids employed. The most frequently used [17, 46, 76] and simplest choice are the multi-linear piecewise basis functions, based upon the one-dimensional formula

$$\Psi_{i,j}(\xi) = \begin{cases} 
1 - (m_i - 1)|\xi - \xi_{i,j}| & \text{if } |\xi - \xi_{i,j}| < h_i \\
0 & \text{otherwise}
\end{cases}$$

These 1D basis functions can be used to form a set of $d$-dimensional basis functions

$$\Psi_{i,j}(\xi) = \prod_{n=1}^{d} \Psi_{i,j_n}(\xi_n)$$

which span $V_i$. Specifically

$$V_i = \text{span} \{\Psi_{i,j} | j_n = 1, \ldots, m_{i_n}, n = 1, \ldots, d\} \quad (4.3)$$

The spaces $V_i$ can be used to define hierarchical difference spaces $W_i$

$$W_i = V_i \setminus \bigoplus_{n=1}^{d} V_{i-e_n} \quad (4.4)$$

These spaces consist of all the basis functions $\Psi_{i,j} \in V_i$ with associated points $\xi_{i,j}$ that are not associated with any of the basis functions in spaces smaller than $V_i$. A discrete space $V_k$ is smaller than a space $V_i$ if $k \leq i$. Setting $V_i = 0$ and using (4.3) and (4.4) we obtain

$$W_i = \text{span} \{\Psi_{i,j} | j \in B_i\}$$

where

$$B_i = \{j_n = 1, \ldots, m_{i_n}, j \text{ odd } n = 1, \ldots, d\} \quad (4.5)$$

These hierarchical difference spaces can be used to decompose the trial space $V$ such that

$$V = \bigoplus_{k_1=0}^{\infty} \cdots \bigoplus_{k_d=0}^{\infty} W_k = \bigoplus_{k \in \mathbb{R}^d} W_k$$
4.2. A LOCAL HIGH-ORDER BASIS

With such a decomposition any function $u(\xi) \in V$ can be approximated by

$$u_{l,d}(\xi) = \sum_{i \leq l} \sum_{j \in B_i} w_{ij} \phi_{ij}(\xi)$$  \hspace{1cm} (4.6)

where $w_{ij} \in \mathbb{R}$ are the coefficient values of the hierarchical product basis, also known as the hierarchical surplus.

For numerical purposes we must truncate the number of difference spaces used to construct $V$ to some level $l$. One choice is the finite-dimensional tensor product space given by

$$V_{l,d}^{(\infty)} \bigoplus_{|i|_{\infty} \leq l} W_i$$  \hspace{1cm} (4.7)

However this approach suffers the curse of dimensionality. The number of points $|V_{l,d}^{(\infty)}| = (2^{l+1} + 1)^d = \mathcal{O}(h_l^{-d}) = \mathcal{O}(2^{l+1})$ grows exponentially with the dimensionality $d$.

Sparse grids can be used to delay the curse of dimensionality by only selecting the difference spaces $W_i$ so that decreasing importance is given to the higher-dimensional subspaces. Specifically the finite dimensional sparse grid space is defined by

$$V_{l,d}^{(1)} = \bigoplus_{|i|_1 \leq l} W_i$$  \hspace{1cm} (4.8)

The size of the sparse grid space is $|V_{l,d}^{(1)}| = \mathcal{O}(h_l^{-1} \cdot |\log_2 h_l|^{d-1}) = \mathcal{O}(2^l \cdot l^{d-1})$ which is a significant reduction on the $\mathcal{O}(2^{l+1})$ number of points required by the full tensor product space $V_{l,d}^{(\infty)}$.

Figures 4.1 and 4.2 illustrate the subspace decomposition of the two dimensional tensor product space $V^{(\infty)}$ and the sparse grid space $V^{(1)}$.

4.2 A Local High-Order Basis

Sparse grids are not restricted to piecewise multi-linear basis functions that are constructed on equidistant grids. Various formulations exist. In the following we propose a high order local basis for stochastic collocation. This basis was first proposed by Bungartz [16] for the solution of partial differential equations. Here we extend this work to high-dimensional uncertainty quantification of non-smooth problems. The local nature of the basis functions allows for $h$-adaptivity and restricts the effects of Gibbs type phenomenon experienced by global polynomial approximation whilst still achieving polynomial convergence in smooth regions.
Figure 4.1: The two-dimensional tensor product space $V_2^{(\infty)} \bigoplus_{|i|_{\infty} \leq 2} W_i$ and the assignment of the grid points to the subspaces $W_i$

Figure 4.2: The two-dimensional sparse grid space $V_2^{(1)} = \bigoplus_{|i|_{1} \leq 2} W_i$ and the assignment of the grid points to the subspaces $W_i$
4.2. A LOCAL HIGH-ORDER BASIS

4.2.1 Construction

As with the linear case, we restrict our attention to grids $\Omega_i$ with mesh spacing $h_i$ that is equidistant with respect to each individual dimension but may vary between dimensions. Let $\Psi_{i,j}^{(p)}$ denote a one-dimensional polynomial of degree $p$ defined on the interval $[\xi_{i,j} - h_i, \xi_{i,j} + h_i]$. This localized support is essential for application of the high-order basis to non-smooth problems and the ultimate goal of an adaptive method. Uniquely defining this basis requires $p + 1$ conditions that $\Psi_{i,j}^{(p)}$ must satisfy.

Here we take advantage of the fact that each point $\xi_{i,j}$ has an ancestry. The one-dimensional equidistant points $\xi_{i,j}$ can be considered as a tree-like data structure as shown in Figure 4.3. The coordinate of each point is defined uniquely by the level $i$ and the position $j$. With this observation we can define a local $p$-th order polynomial using the points $\xi_{i,j} - h_i, \xi_{i,j} + h_i$ and the next $p - 2$ closest hierarchical ancestors of $\xi_{i,j}$.

**Definition 4.1.** Given the one-dimensional grid $\Omega_i$ with grid points defined according to (4.2) the $p$-th order basis function $\Psi_{i,j}^{(p)}$ is the hierarchical interpolant of the point $\xi_{i,j} - h_i, \xi_{i,j} + h_i$ and the next $p - 2$ closest hierarchical ancestors of $\xi_{i,j}$ restricted to the local support $[\xi_{i,j} - h_i, \xi_{i,j} + h_i]$. Specifically by renaming all the points except $\xi_{i,j}$ in ascending order as $\xi_0, \ldots, \xi_p$ the piecewise Lagrange basis can be written

$$\Psi_{i,j}^{(p)}(\xi) = \begin{cases} \prod_{k=0}^{p} \frac{\xi - \xi_k}{\xi_{i,j} - \xi_k} & \text{if } |\xi - \xi_{i,j}| < h_i \\ 0 & \text{otherwise} \end{cases}$$

The points $\xi_k$ used to construct the basis function $\Psi_{i,j}^{(p)}$ satisfy the following property. For any $\Psi_{i,j}^{(p)}$ there are two points which satisfy $|\xi_k - \xi_{i,j}| = h_i$ and one
Figure 4.4: Ancestors necessary to calculate piecewise Lagrange basis $\Psi_{i,j}^{(p)}$ for $p = 4$

point belonging to each of the following sets:

\[
\begin{align*}
\{ \xi_k : |\xi_k - \xi_{i,j}| = 3h_i \} \\
\{ \xi_k : |\xi_k - \xi_{i,j}| \in \{5 \cdot h_i, 7 \cdot h_i\} \} \\
\vdots \\
\{ \xi_k : |\xi_k - \xi_{i,j}| \in \{(2^{p-1} + 1) \cdot h_i, \cdots, (2^p - 1) \cdot h_i\} \}
\end{align*}
\]

Figure 4.4 shows the hierarchical ancestors necessary to construct the four possible quartic basis functions. On level 5 each basis function is repeated twice.

In general there are $2^{p-2}$ basis functions of degree $p$. The order $p$ of the basis function is dependent on the hierarchical level $i$ of $\xi_{i,j}$. For $p > 2$, $p + 1$ ancestors are needed to construct the basis $\Psi_{i,j}$. On level one, only one ancestor ($\xi = 0.5$, on level zero) is available and thus only linear basis functions can be used. On level two, only two ancestors exist, and thus only linear or quadratic basis functions can be used. Subsequently basis functions of degree $p$ can only be used when $i \geq p$. This represents a slight modification of the approach employed by Bungartz [16] who allowed basis functions of degree $p$ to be used when $i \geq p - 1$. Bungartz approach was designed for sparse grids with homogeneous boundary conditions.

Figure 4.5 (a) shows the quartic basis functions. Here there are four different
but symmetrical basis functions of degree four. Figure 4.5 (b) shows the ancestor points needed to construct one of these four possible quartic basis functions, the hierarchical interpolant and its restriction to its local support.

![Figure 4.5: (a) The hierarchical basis functions for \( p = 4 \) and (b) the hierarchical interpolant \( \Psi_{5,3}^{(4)} \) and its restriction to the local support](image)

Throughout the remainder of this manuscript we restrict our attention to basis functions with fixed maximum degree \( p_{\text{max}} \). That is, the order of the basis is increased with each level of the sparse grid until the order of the basis is \( p_{\text{max}} - 1 \). On all subsequent levels the order of the basis is kept constant at \( p_{\text{max}} \). The tensor product construction of the multi-dimensional basis means that the degree \( p = (p_1, \ldots, p_d) \) of a \( d \)-dimensional basis function \( \Psi_{1,j} \) must satisfy

\[
0 \leq p_n = \min\{p_{\text{max}}, i_n\}, \quad i_n \geq 0, \quad n = 1, \ldots, d
\]

Here \( p_n = 0 \) represents the constant function centred at the midpoint of \( \Gamma_n \).

Figure 4.6 shows the hierarchical decomposition and the associated basis functions of the sparse grid space \( V_2^{(2)} \). The quadratic basis does not appear until \( l = 2 \). Note that increasing the degree \( p \) indefinitely can result in large increases in the hierarchical surplus. This will be discussed further in Section 4.4.

### 4.2.2 Efficient Evaluation of the Hierarchical Surplus

In the piecewise linear setting the hierarchical surplus is given by

\[
w_{i,j} = u(\xi_{i,j}) - \frac{u(\xi_{i,j} - h) + u(\xi_{i,j} + h)}{2}
\]
Figure 4.6: Hierarchical decomposition and the associated basis functions of the sparse grid space $V_2^{(2)}$. 
4.2. A LOCAL HIGH-ORDER BASIS

This can be rewritten in the following operator form

\[ w_{i,j} = \left[ \begin{array}{cc} -\frac{1}{2} & 1 - \frac{1}{2} \end{array} \right] u_{i,j} \]

Using the tensor product we can calculate the d-dimensions hierarchical surplus using the stencil formulation

\[ w_{i,j} = \left( \prod_{n=1}^{d} \left[ \begin{array}{c} -\frac{1}{2} \end{array} \right] \right) u_{i,j} \]

Figure 4.7 illustrates the function values needed to evaluate the two-dimensional linear hierarchical surplus. This method is extremely efficient as often the dimension of the subspace under consideration is very small (\( \hat{d} \ll d \)) and the number of points required to evaluate the surplus is \( 3^d \). The hierarchical surplus can also be evaluated by simply measuring the difference between the true function \( u \) and the interpolant \( u_{-1,d} \) at \( \xi_{i,j} \) [76]. However this is always more expensive than the aforementioned approach.

Unlike in the linear setting the hierarchical surplus associated with high-order local basis approximation cannot be evaluated using a simple linear sum of the neighbouring function values. However appropriate linear combinations of the linear surplus can be used to evaluate any higher degree surplus.

**Lemma 4.2** (Bungartz [16]). For any \( u \in C^{p+1}([0,1]) \), the hierarchical surplus \( w_{i,j}^{(p)} \) of degree \( p \) can be calculated with the help of \( w_{i,j}^{(p-1)} \) and \( w_{\text{par},\text{par}}^{(p-1)} \), where
\( i_{\text{par}}, j_{\text{par}} \) are the level and position index of the parent \( \xi_{i_{\text{par}}, j_{\text{par}}} \) of \( \xi_{i,j} \). Specifically

\[
u_{i,j}^{(p)} = \nu_{i,j}^{(p-1)} - \alpha^{(p)} \cdot w_{i_{\text{par}}, j_{\text{par}}}^{(p-1)}
\]

where

\[
\alpha^{(p)} = \frac{\xi_m - \xi_n}{\xi_{p+1} - x_n} \prod_{k=0, k \neq m, k \neq n}^{p} \frac{\xi_m - \xi_k}{\xi_n - \xi_k}
\]

depends on the relative position of \( \xi_{i,j} \) ancestors, but not on the interpolated values \( u(\xi_{i,j}) \).

For example the one-dimensional quadratic hierarchical coefficients are given by

\[
u_{i,j}^{(2)} = \nu_{i,j}^{(1)} - \frac{1}{4} w_{i_{\text{par}}, j_{\text{par}}}^{(1)}
\]

Figure 4.8 illustrates the relationship between the quadratic surplus at \( \xi_{i,j} \) and the linear surplus at \( \xi_{i,j} \) and its ancestor \( p(\xi_{i,j}) \). In this figure \( \xi_0 = \xi_{i,j} \) is the point being interrogated, \( \xi_1 \) is its parent, and \( \xi_2 \) is the direct ancestor of \( \xi_1 \). As determined by (4.9) four function values are needed to evaluate the quadratic hierarchical surplus. The thick solid line represents the true function and \( h \) is the mesh size at the finest level of refinement \( i \). The dashed line represents the quadratic interpolant on the coarser mesh \( i - 1 \), which interpolates the ancestors of \( \xi_{i,j} \), that is \( \xi_0, \xi_2 \) and \( \xi_3 \). The dashed and dotted line depicts the piecewise-linear interpolant on the same mesh and the thin solid line represents the linear interpolant on the coarsest mesh shown, which corresponds to the level of refinement \( i - 2 \). The quadratic surplus is a linear combination of the linear surplus \( \nu_{i,j}^{(1)} \) at \( \xi_{i,j} \) and the linear surplus \( \nu_2^{(1)} \) at its parent \( \xi_2 \).

The one-dimensional values of \( \alpha \) can be calculated for arbitrary order according to (4.10). The hierarchical surpluses can then be calculated recursively by using \( p \) linear surpluses. The values of alpha for \( p \leq 5 \) are shown in Figure 4.9. Figure 4.10 illustrates values of \( \alpha \) if \( p_{\text{max}} = 3 \).

The calculation of the multi-dimensional surpluses can be calculated by applying the following stencil recursively:

\[
u_{i,j}^{(p)} = \left( \prod_{n=1}^{d} \left[ 1 - \alpha^{(p)} \right]_{i_n,j_n} \right) \nu^{(p-1)}, \quad p > 1
\]

An example of the points required to calculate the two-dimensional quadratic surplus is shown in Figure 4.11.
\[
\begin{align*}
\xi_0 &= \xi_{i,j} - h \\
\xi_1 &= \xi_{i,j} \\
\xi_2 &= \xi_{i,j} + h \\
\xi_3 &= \xi_{i,j} + 3h
\end{align*}
\]

Figure 4.8: Evaluation of the one-dimensional quadratic hierarchical surplus from the associated linear surpluses.

Figure 4.9: Values of \( \alpha \), calculated using Equation (4.10), for \( p \leq 5 \).
Figure 4.10: Values of $\alpha$, calculated using Equation (4.10), for $p_{\text{max}} = 3$

Figure 4.11: The points required to evaluate the two-dimensional quadratic surplus
4.3. ADAPTIVITY

4.2.3 Quadrature

The extension from interpolation to quadrature is straightforward. We can approximate the integral of a function \( u \)

\[
I[u(x, t)] = \int_{\Gamma} u(x, t, \xi) \, d\mu(\xi)
\]

using the hierarchical sparse grid interpolant (4.6). Utilizing this formulation these integrals can be approximated by

\[
I[w_{t,d}(x, t)] = \int_{\Gamma} \sum_{i \in \mathcal{I}} \sum_{j \in B_i} w_{ij} \cdot \Psi_{ij}(\xi) \, d\mu(\xi)
\]

\[
= \sum_{i \in \mathcal{I}} \sum_{j \in B_i} w_{ij} \cdot v_{ij}
\]

where the weights

\[
v_{ij} = \int_{\Gamma} \Psi_{ij}(\xi) \, d\mu(\xi)
\]

can be calculated easily and with no need for extra function evaluations once the interpolant has been constructed. One simply needs to store the volumes of the high-order basis functions. For \( d\mu = 1 \) these volumes can be calculated analytically and are given in Table 4.1.

4.3 Adaptivity

The classical sparse grids presented in Section 4.1 are based upon the index set

\[
\mathcal{I} = \{ i \in \mathbb{N}^d : ||i||_1 \leq 1 \}
\]

This construction delays the curse of dimensionality by assuming that the importance of any interaction between a subset of a function's variables decreases as the number of variables involved in the interaction (interaction order) increases. Although an advance on full tensor product spaces, such approximations can still be improved. The classical sparse grid construction treats all dimensions equally and all interactions of the same order equally. In practice, often only a small subset of variables and interactions contributes significantly to the variability of the function \( u \). Moreover, frequently only small regions within the input space possess high variability. In some cases the important dimensions, interactions and regions can be determined a priori, but in most cases these properties must be identified during the computational procedure.
Table 4.1: The quadrature weights associated with each basis function when $p \leq 5$

<table>
<thead>
<tr>
<th>Basis</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Psi_{i,j}^{(0)}$, $i = 0$, $j = 1$</td>
<td>1</td>
</tr>
<tr>
<td>$\Psi_{i,j}^{(1)}$, $i = 1$, $\forall j$</td>
<td>$\frac{1}{4}$</td>
</tr>
<tr>
<td>$\Psi_{i,j}^{(4)}$, $i &gt; 1$, $\forall j$</td>
<td>$h$</td>
</tr>
<tr>
<td>$\Psi_{i,j}^{(2)}$, $i \geq 2$, $\forall j$</td>
<td>$\frac{4h}{3}$</td>
</tr>
<tr>
<td>$\Psi_{i,j}^{(3)}$, $i \geq 3$, $\forall j$</td>
<td>$\frac{4h}{3}$</td>
</tr>
<tr>
<td>$\Psi_{i,j}^{(4)}$, $i \geq 4$, $j % 8 = 1$, $j % 8 = 7$</td>
<td>$\frac{424h}{315}$</td>
</tr>
<tr>
<td>$\Psi_{i,j}^{(4)}$, $i \geq 4$, $j % 8 = 3$, $j % 8 = 5$</td>
<td>$\frac{296h}{225}$</td>
</tr>
<tr>
<td>$\Psi_{i,j}^{(5)}$, $i \geq 5$, $j % 16 = 1$, $j % 16 = 15$</td>
<td>$\frac{256h}{189}$</td>
</tr>
<tr>
<td>$\Psi_{i,j}^{(5)}$, $i \geq 5$, $j % 16 = 3$, $j % 16 = 13$</td>
<td>$\frac{256h}{195}$</td>
</tr>
<tr>
<td>$\Psi_{i,j}^{(5)}$, $i \geq 5$, $j % 16 = 5$, $j % 16 = 11$</td>
<td>$\frac{1088h}{825}$</td>
</tr>
<tr>
<td>$\Psi_{i,j}^{(5)}$, $i \geq 5$, $j % 16 = 7$, $j % 16 = 9$</td>
<td>$\frac{3776h}{2235}$</td>
</tr>
</tbody>
</table>
4.3. ADAPTIVITY

The generalised sparse grid method [41] is extremely effective at determining the dimensions and interactions that contribute significantly to the function variability, according to some predefined measure. However, the efficiency of this method deteriorates when a large proportion of the function variability is concentrated in small regions of the input space. In contrast to the generalised sparse grid method, $h$-adaptive methods such as the ASGC method [76] attempt to reduce the number of points in a sparse grid by concentrating refinement only in rapidly varying or discontinuous regions. This method is implicitly dimension adaptive but often points are constructed necessarily in ‘unimportant’ dimensions [78].

In this section we propose a method which combines the strengths of both $h$-adaptive refinement and the generalised sparse grid algorithm. We coin this approach the $h$-Adaptive Generalised Sparse Grid ($h$-GSG) method.

### 4.3.1 Generalised Sparse Grid (GSG) Algorithm

Gerstner [41] generalised the sparse grid construction by considering the index sets based upon the admissibility criterion

$$i - e_j \in \mathcal{I} \text{ for } 1 \leq j \leq d, \ i_j > 1$$

(4.11)

This so-called generalised sparse grid method [41] is extremely effective at determining the hierarchical difference spaces that contribute significantly to the function variability, according to some predefined measure.

The generalised sparse grid method is a greedy algorithm which attempts to find the index set $\mathcal{I}$ such that for a given number of points the approximation error is minimized. Starting with $\mathcal{I} = \{0\}$ the index set is built iteratively by searching the forward neighbourhood of the current index set for new admissible indices. The forward neighbourhood of an index $i$ is the set of $d$ indices $\{i + e_j : 1 \leq j \leq d\}$. Similarly, the backwards neighbourhood is just $\{i - e_j : 1 \leq j \leq d\}$.

Once the forward neighbourhood has been identified, each forward neighbour is checked for admissibility using (4.11). The grid points associated with each admissible index are then evaluated and the error of these spaces calculated. The calculation of these errors will be addressed shortly. The forward neighbour with the largest error is then added to the current index set $\mathcal{I}$ and the set of admissible indices is updated.

To facilitate easy computation of new admissible indices we partition the index set $\mathcal{I}$ into two disjoint sets $\mathcal{O}$ and $\mathcal{A}$, which Gerstner [41] refers to as the old and
active index sets, respectively. The active index set $\mathcal{A}$ contains all the indices in $\mathcal{I}$ that have been constructed but whose forward neighbours have not been considered. The old index set contains all the indices remaining in the current index set $\mathcal{I}$. The algorithm proceeds by searching the forward neighbourhood of the index $i \in \mathcal{A}$ with the largest error for admissible indices. All the new admissible index sets are added to the active index set $\mathcal{A}$ and the index $i$ is then added to the old index set $\mathcal{O}$. This process is repeated until a global error is below a predefined tolerance $\varepsilon$.

The exact error associated with each index $i$ is unknown. Consequently each time an index $i$ is deemed admissible an approximation of the error $r_i$ must be used. Numerous error criteria can be utilised. Here we employ the error measure

$$ r_i = \left| \sum_{j \in B_i} w_{ij} \cdot v_i \right| $$

These index-based error criteria can be used to approximate the global error. We propose the following global error indicator $r$

$$ r = \sum_{i \in \mathcal{A}} r_i $$

When $r < \varepsilon$ the generalised sparse grid algorithm is terminated. The generalised sparse grid algorithm is presented in Algorithm 1.

Three steps of the generalised sparse grid algorithm depicting the construction of the sparse grid index set are shown in Figure 4.12. The top row represents the current index sets. The bottom row depicts the corresponding sparse grid. At each step the forward neighbours of the grid index $i$ with the largest error $r_i$ (striped box) are checked for admissibility. A forward neighbour is admissible if all indices in its backwards neighbourhood are in the old index set (grey boxes). All admissible indices (pointed to by an arrow) are added to the active index set (black and striped boxes).

The striped box $i = (1,1)$ in the first step has two admissible neighbours as the backwards neighbourhoods of both forward neighbours are complete. In comparison the striped box $i = (2,1)$ in the second step only has one admissible index. The index $j_1 = (3,1)$ has two backwards neighbours $j_1 - e_1 = (0,2)$ and $j_1 - e_2 = (1,1)$ in the old index set, and thus is admissible. In contrast the index $j_2 = (2,2)$ has one backwards neighbour in the old index set $j_2 - e_1 = j_2$ and one in the active set $j_2 - e_2 = (2,1)$, and so is not admissible.

Although we wish to use the generalised sparse grid algorithm for interpolation the error criteria we have proposed are based upon an integral formulation.
4.3. ADAPTIVITY

Algorithm 1 Generalised Sparse Grid Interpolation

\( i = (0, \ldots, 0) \)
\( \mathcal{A} := \{i\} \)
\( O := \emptyset \)
\( r := r_1 \)

while \( r > \varepsilon \) do
  select \( i \in \mathcal{A} \) with largest error indicator \( r_1 \)
  \( \mathcal{A} := \mathcal{A} \setminus \{i\} \)
  \( O := I \cup \{i\} \)
  \( r := r - r_1 \)
  for \( k := 1, \ldots, d \) do
    \( j = i + e_k \)
    if \( j - e_n \in O \ \forall n = 1, \ldots, d \) then
      \( \mathcal{A} := \mathcal{A} \cup \{j\} \)
      CreateGrid(j)
      \( r := r + r_j \)
    end if
  end for
end while
Figure 4.12: Three steps of the generalised sparse grid algorithm. The top row represents the current index sets. Active grid indices are in black, indices in the old index set $\mathcal{O}$ are in grey and the active index with the largest error indicator is striped. The bottom row depicts the corresponding sparse grid. Only the points associated with indices in the old index set are shown.
Specifically the error indicator \( r_i \) measures the contribution of the index \( i \) to the global integral approximation. Furthermore the algorithm is terminated when the approximated error \( r \) in the integral is below a predefined threshold \( \epsilon \). This choice was made purposefully.

The magnitude of the hierarchical surplus, which is the size of the difference between the true function and the sparse grid approximation at a grid point, may be more synonymous with interpolation. Simply adding indices with large hierarchical surpluses, however, is inefficient. The magnitude of the hierarchical surpluses decays slowly in regions adjacent to discontinuities. At the site of jump discontinuities the hierarchical surplus will be at best half of the magnitude of the jump, for any finite number of grid points. Thus the algorithm can proceed much further than is necessary. The use of the error criterion \( r_i \) provides a lower bound on the size of the support of the basis functions used by weighting the magnitude of the hierarchical surplus by the probability that an arbitrary point \( \xi \) will fall within its support.

### 4.3.2 Extending the GSG algorithm

**h-Adaptivity**

Each time a grid index \( i \) is added to the active index set, the traditional generalised sparse grid algorithm evaluates all the points in the set \( B_i \). Such an approach is inefficient if a large proportion of the function variability is concentrated in small regions of the input space. When using equidistant grids, the creation of the grid \( i \) requires approximately two times the number of grid points (and thus function evaluations) than those necessary to construct the index \( i - e_k \). Consequently we propose introducing an \( h \)-adaptive procedure to construct the points associated with each grid index.

To integrate \( h \)-adaptivity into the generalised sparse grid algorithm we define the two sets \( A_i \) and \( R_i \) for each grid index \( i \). We refer to these sets respectively as the active point set and redundant point set of the grid index \( i \). The active point set \( A_i \) contains all admissible points associated with the index \( i \) with an error indicator \( \gamma_{ij} \geq \epsilon \). The redundant point set \( R_i \) contains all admissible points with \( \gamma_{ij} < \epsilon \). A point is admissible if one of its \( d \) possible ancestors exists in the grids associated with the backwards neighbourhood of \( i \). If, and only if, a grid point is admissible it is created (the function is evaluated) and the error indicator \( \gamma_{ij} \) calculated. This drastically reduces the number of points generated when a new grid index is created.
To guide $h$-refinement we propose using the error indicator

$$\gamma_{i,j} = |w_{i,j} \cdot v_{i,j}|$$

If $\gamma_{i,j} \geq \varepsilon$ the point $\xi_{i,j}$ is added to the active point set $A_i$, otherwise it is added to the redundant index set $R_i$. The procedure used to implement $h$-adaptivity on each grid index is outlined in Algorithm 2 below. Figure 4.13 shows an example

Algorithm 2 CreateGrid(i)

\[
\begin{array}{l}
\text{for } (n \in \{1, \ldots, d\} ) \text{ do} \\
\quad \text{for } (\xi_{i-e_i,j} \in A_{i-e_i} ) \text{ do} \\
\quad \quad C = \text{FindAxialChildren}(\xi_{i-e_i,j}, n) \\
\quad \quad \text{for } (\xi_{i,k} \in C ) \text{ do} \\
\quad \quad \quad \text{if } ( \gamma_{i,k} \geq \varepsilon ) \text{ then} \\
\quad \quad \quad \quad A_i := A_i \cup \{\xi_{i,k}\} \\
\quad \quad \quad \text{else} \\
\quad \quad \quad \quad R_i := R_i \cup \{\xi_{i,k}\} \\
\quad \quad \text{end if} \\
\quad \text{end for} \\
\text{end for} \\
\text{end for}
\end{array}
\]

of $h$-adaptivity integrated with the generalised sparse grid algorithm. Here the grid index $i = (2,2)$ has been deemed admissible by the generalised sparse grid algorithm. Both the backwards neighbours ($i - e_1 = (1,2)$ and $i - e_1 = (2,1)$) exist in the old index set $O$. The active points in the backwards neighbours are used to determine which points in the active index $i$ should be evaluated. For any point in the active set of the $n$-th backwards neighbour $n = 1, \ldots, d$ the children of that point are created in the $n$-th axial direction. This refinement is carried out for all points in the active point set of the backward neighbour and for all backwards neighbours.

**Efficient Termination**

The generalised sparse grid (GSG) algorithm is a greedy algorithm which efficiently identifies the sparse grid index set that is necessary to interpolate a function up to a level of predefined accuracy. The algorithm can determine the number of variable interactions and the individual importance of each variable [47]. This
4.3. ADAPTIVITY

Figure 4.13: An example of $h$-adaptivity integrated with the generalised sparse grid algorithm. Let us assume that the function only varies significantly in the left half of the domain. The top left and bottom right grids are the backwards neighbours of the grid being created. Circles represent points in the sparse grid, squares are points in the active point sets, and crosses are points in the redundant point sets. The active (square) points in the backwards neighbours are refined to produce the set of new points that must be added. In this example only two new points (squares in grid $i = (2, 2)$) are added.
is achieved by successively adding the grid index with the largest error indica-
tor to the old index set and searching its forward neighbourhood for admissible
indices. Every admissible index is added (and thus created) to the active index
set without regard for the error associated with that grid. The algorithm finally
terminates when $\sum_{i \in A} r_i < \varepsilon$.

The decision to add all indices $i$, regardless of the size of their associated error
indicator $r_i$, typically results in the creation of a large number of grids with $r_i <<
\varepsilon$ and which have little effect on the accuracy of the approximation. To reduce
the number of these unimportant indices we propose only adding admissible indices
with $r_i \geq \varepsilon$ to the active index set $A$. This significantly reduces the number
of grid indices in the final index set $I$ and thus the total number of function
evaluations, with only minor effect on the overall accuracy of the generalised
sparse grid method.

4.4 Error Analysis

In this section we derive a bound on the error of the proposed $h$-GSG method.
The error in the interpolant is dependent on the smoothness of the function being
approximated. We present the necessary smoothness conditions and show that
when met the degree of the local polynomial significantly increases the efficiency
of the proposed method. Finally we show that by introducing $h$-adaptivity we
are able to reduce further the number of points needed to achieve a pre-defined
level of accuracy.

Let us begin by stating some of the analytical properties of hierarchical sparse
grid representations $u_{t,d}$, for functions $u \in X^{p,q}$, first outlined by Bungartz [16].
For ease of discussion let us rewrite (4.6) in the following form

$$u_{t,d}(\xi) = \sum_{i \leq t} u_i(\xi), \quad u_i(\xi) = \sum_{j \in B_i} w_{ij}^{(p)} \cdot \Psi_{ij}^{(p)} \in W_i$$  \hspace{1cm} (4.12)

Bungartz [17] and Griebel obtained an integral representation of the hier-
archical surplus $w_{ij}^{(p)}$ for arbitrary $p$. From this integral representation of the
hierarchical surplus they determined upper bounds on the magnitude of the sur-
plus.

**Lemma 4.3** (Bungartz and Griebel [17]). If $u \in X^{p+1,q}$ then the hierarchical
coefficients satisfy

\[
|w_{ij}^{(p)}| \leq c(p) \cdot \left( \frac{1}{2} \right)^d \cdot 2^{-(i+1)(p+1)/2} \cdot |u|_{p,1,\infty}
\]

(4.13)

\[
|w_{ij}^{(p)}| \leq c(p) \cdot \left( \frac{1}{6} \right)^{d/2} \cdot 2^{-(i+1)(p+1)/2} \cdot 2^{i+1/2} \cdot \|D^{p+1}u|_{\text{supp}(\psi_{ij})}\|_2
\]

where

\[
c(p) = \prod_{n=1}^{d} \frac{2^{p_0(p_n+1)/2}}{(p_n + 1)!}
\]

This result substantiates the claim made in Section 4.2 that the order of the polynomial basis cannot be increased indefinitely. The non-constant term \( p \) increases quickly with dimensionality. Also note that here we are defining the isotropic index set by \( |i|_1 \leq l \) rather than \( d \leq |i|_1 \leq l + d - 1 \). Consequently the bounds, above and in the following, look slightly different to those in Bungatz [16] and Bungartz and Griebel [17].

Based upon the analytical properties of \( \Psi_{ij}^{(p)} \), bounds on the norms of the basis polynomials can be determined.

**Lemma 4.4** (Bungartz [16]). For any \( d \)-dimensional hierarchical piecewise Lagrange basis function \( \Psi_{ij}^{(p)} \) the following hold:

\[
\|\Psi_{ij}^{(p)}\|_q \leq \begin{cases} 
1.117^d, & q = \infty \\
1.117^d \cdot 2^{d/q} \cdot 2^{-|i+1|_1/q}, & 1 \leq q < \infty 
\end{cases}
\]

The previous two lemmas can be used to obtain bounds on the contribution of \( u_i \in W_i \) to the hierarchical representation (4.12) of any \( u \in X^{p,q} \).

**Lemma 4.5** (Bungartz and Griebel [17]). For any function \( u \in X^{p+1,q} \) approximated in the sparse grid space \( V_1^{*} \) using the hierarchical representation (4.12), the following bounds on the components \( u_i \in W_i \) hold:

\[
\|u_i^{(p)}\|_\infty \leq c(p) \cdot 0.5585^d \cdot 2^{-(i+1)(p+1)/2} \cdot |u|_{p,1,\infty}
\]

(4.14)

\[
\|u_i^{(p)}\|_2 \leq c(p) \cdot 1.117^d \cdot \left( \frac{1}{3} \right)^{d/2} \cdot 2^{-(i+1)(p+1)/2} \cdot 2^{i_1/2} \cdot \|D^{p+1}u|_{\text{supp}(\psi_{ij})}\|_2
\]

Finally, using the above lemmas, Bungartz and Griebel [17] determined upper bounds on the interpolation accuracy of the sparse grid approximation \( u_{i,d} \).
Lemma 4.6 (Bungartz and Griebel [17]). If \( u \in X^{p+1,q} \) then the following bounds hold:

\[
\|u - u_{t,d}^{(p)}\|_\infty \leq \left( \frac{0.5585}{2^{p+1}} \right)^d \cdot c(p \cdot 1) \cdot |u|_{(p+1)-1,\infty} \cdot A_{t,d} \cdot h_t^{p+1} + O(h_t^{p+1})
\]

\[
= O(h_t^{p+1} \cdot n_\delta^{-1})
\]

\[
\|u - u_{t,d}^{(p)}\|_2 \leq 3.257 \cdot \left( \frac{1.117}{\sqrt{2} \cdot 2^{p+1}} \right)^d \cdot c(p \cdot 1) \cdot |u|_{(p+1)-2,2} \cdot A_{t,d} \cdot h_t^{p+1} + O(h_t^{p+1})
\]

\[
= O(h_t^{p+1} \cdot n_\delta^{-1})
\]

where

\[
A_{t,d} = \sum_{k=0}^{d-1} \binom{l + d - 1}{k} = \frac{l^{d-1}}{(d-1)!} + O(l^{d-2})
\]

These bounds hold true for sparse grids with and without boundary points. In comparison the \( L_\infty \) and \( L_2 \) error of the full tensor product construction \( V_t^\infty \) in (4.7) are bounded by \( O(h_t^{p+1}) \). The accuracy of the sparse grid approximation is only slightly worse than its full tensor product counterpart, yet the number of points required to construct the sparse grid is significantly less. This allows for the treatment of higher-dimensional problems than is possible if full tensor product grids are used. We have not, however, removed the curse of dimensionality but simply delayed its onset. Adaptive methods such as the one proposed here can be used to further increase the efficiency of sparse grid approximations.

Based upon the properties of the subspaces \( W_i \) we can derive the following error estimate for the \( h \)-adaptive method presented in Section 4.3.

**Theorem 4.7.** For any function \( u \), a tolerance \( \varepsilon > 0 \) and the error indicator \( \gamma_{i,j} = |w_{i,j}| \), the following error bound on the \( h \)-GSG interpolant \( u_{\varepsilon,d} \) holds

\[
\|u - u_{\varepsilon,d}^{(p)}\|_\infty \leq \|u - u_{t,d}^{(p)}\|_\infty + 1.117^d \cdot \varepsilon \cdot N(\varepsilon)
\]

\[
\|u - u_{\varepsilon,d}^{(p)}\|_2 \leq \|u - u_{t,d}^{(p)}\|_2 + 1.117^d \cdot 2^{d/2} \cdot 2^{-\frac{\|j|}{2}} \cdot \varepsilon \cdot N(\varepsilon)^{1/2}
\]

where given the adaptively determined index set \( I \),

\[
l = \max_{i \in I} \left\{ \max_{n=1,...,d} \hat{t}_n \right\}
\]

and \( N(\varepsilon) \) is the number of points found in \( u_{i,d} \) but absent from \( u_{\varepsilon,d} \).

**Proof.** Given the tolerance \( \varepsilon \) and an error indicator \( \gamma_{i,j} = |w_{i,j}| \), denote the \( h \)-GSG approximant by

\[
u_{\varepsilon,d}(\xi) = \sum_{i \in I} u_i(\xi), \quad u_i(\xi) = \sum_{j \in B_i, |w_{i,j}^{(p)}| \geq \varepsilon} w_{i,j}^{(p)} \cdot \psi_{i,j}^{(p)} \in W_i
\]
where $\mathcal{I}$ is the adaptively determined index set. Let

$$l = \max_{i \in \mathcal{I}} \left\{ \max_{n=1, \ldots, d} i_n \right\}$$

The difference between the $h$-GSG and isotropic sparse grid approximations satisfies

$$u_{i,d}^{(p)} - u_{e,d}^{(p)} = \sum_{|i|_1 \leq l} \sum_{j \in B_1} w_{i,j}^{(p)} \cdot \Psi_{i,j}^{(p)} - \sum_{|i|_1 \in \mathcal{I}} \sum_{j \in B_1} w_{i,j}^{(p)} \cdot \Psi_{i,j}^{(p)}$$

$$= \sum_{|i|_1 \leq l} \sum_{j \in B_1} w_{i,j}^{(p)} \cdot \Psi_{i,j}^{(p)}$$

Noting that the supports of all $\Psi_{i,j}^{(p)}$ contributing to $\sum_{j \in B_1} w_{i,j}^{(p)} \cdot \Psi_{i,j}^{(p)}$ are mutually disjoint and since $\|\Psi_{i,j}^{(p)}\|_{\infty} \leq 1.117^d$

$$\|u_{i,d}^{(p)} - u_{e,d}^{(p)}\|_{\infty} \leq \sum_{|i|_1 \leq l} \left\| \sum_{j \in B_1} w_{i,j}^{(p)} \cdot \Psi_{i,j}^{(p)} \right\|_{\infty}$$

$$= \sum_{|i|_1 \leq l} \sum_{j \in B_1} |w_{i,j}^{(p)}| \cdot \|\Psi_{i,j}^{(p)}\|_{\infty}$$

$$\leq 1.117^d \cdot \varepsilon \cdot N(\varepsilon)$$

Analogously, observing $\|\Psi_{i,j}\|_2^2 \leq 1.117^{2d} \cdot 2^d \cdot 2^{-|i+1|}$

$$\|u_{i,d} - u_{e,d}\|_2^2 \leq \sum_{|i|_1 \leq l} \sum_{j \in B_1} |w_{i,j}|^2 \cdot \|\Psi_{i,j}\|_2^2$$

$$\leq 1.117^{2d} \cdot 2^d \cdot 2^{-|i+1|} \cdot \varepsilon^2 \cdot N(\varepsilon)$$

By the triangle inequality we have

$$\|u - u_{e,d}^{(p)}\|_q = \|u - u_{i,d}^{(p)} + u_{i,d}^{(p)} - u_{e,d}^{(p)}\|_q$$

$$\leq \|u - u_{i,d}^{(p)}\|_q + \|u_{i,d}^{(p)} - u_{e,d}^{(p)}\|_q$$

which proves the assertion. \qed
Corollary 4.8. For any function \( u \), a tolerance \( \varepsilon > 0 \) and the error indicator \( \gamma_{ij} = \| w_{ij} \cdot \Psi_{ij} \|_q \), the following error bound on the h-GSG interpolant \( u_{\varepsilon,d} \) holds
\[
\| u - u_{\varepsilon,d}^{(p)} \|_q \leq \| u - u_{i,d}^{(p)} \|_q + \varepsilon \cdot N(\varepsilon)
\]
where \( N(\varepsilon) \) is the number of points found in \( u_{i,d} \) but absent from \( u_{\varepsilon,d} \).

Proof. The proof follows that used to prove Theorem 4.7 with one minor modification. By choosing the error indicator \( \gamma_{ij} = \| w_{ij} \cdot \Psi_{ij} \|_q \), where \( q \) is chosen to match the \( L_q \) norm used to measure the approximation error \( \| u - u_{\varepsilon,d}^{(p)} \|_q \), any point not in the h-GSG interpolant will satisfy \( \gamma_{ij} < \varepsilon \).

The first term in the bound of Theorem 4.7 and the Corollary 4.8 is the interpolation error of the isotropic sparse grid approximation and the second term is the error between the isotropic sparse grid and h-GSG methods. As the tolerance \( \varepsilon \) is reduced so too is the number of missing points \( N(\varepsilon) \) and the adaptive method converges to the isotropic case.

If \( u \in X^{p+1,q} \) then the result of Lemma 4.6 can be used to bound the interpolation error of the isotropic sparse grid approximation. In practice this assumption can be relaxed with the use of adaptive methods such as the one proposed here. Numerical convergence studies of smooth and discontinuous functions are presented in Section 4.5. However if the assumption of bounded mixed derivatives is relaxed the bounds in Lemma 4.6 no longer hold.

The proposed h-GSG algorithm terminates when all points in the sparse grid with an error indicator \( \gamma_{ij} \geq \varepsilon \) have been considered. This truncation of the sparse grid space has an effect on the accuracy of the approximation this effect is quantified by the following theorem.

Theorem 4.9. Let \( u_{\varepsilon,\text{opt}} \) be an interpolation of \( u \) that obtains \( \| u - u_{\varepsilon,\text{opt}} \|_q \leq \varepsilon \) with the least number of function evaluations. Then for any function \( u \) and a given tolerance \( \varepsilon > 0 \) and the error criterion \( \gamma_{ij} = \| w_{ij} \cdot \Psi_{ij} \|_q \), the h-GSG approximation \( u_{\varepsilon,d} \) satisfies
\[
\| u - u_{\varepsilon,d} \|_q \leq \varepsilon (1 + N(\varepsilon))
\]
where \( N(\varepsilon) \) is the number of points in the optimal interpolant but not in the h-GSG interpolant.
4.4. ERROR ANALYSIS

Proof. Let

\[ u_{\varepsilon, \text{opt}} = \sum_{(i,j) \in P_{\varepsilon, \text{opt}}} w_{i,j} \cdot \Psi_{i,j} \]

be an interpolation of \( u \) that obtains \( \| u - u_{\varepsilon, \text{opt}} \|_q \leq \varepsilon \). The points in this optimal approximant are defined by the index set

\[ P_{\varepsilon, \text{opt}} := \{(i,j) : i \in I_{\varepsilon, \text{opt}} \text{ and } j \in B_{1,\varepsilon, \text{opt}} \subseteq B_1\} \]

Similarly denote the \( h \)-GSG interpolant with an error indicator \( \gamma_{ij} \) by

\[ u_{\varepsilon, d} = \sum_{(i,j) \in P_{\varepsilon, d}} w_{i,j} \cdot \Psi_{i,j} \]

where the point indices in the \( h \)-GSG approximant are

\[ P_{\varepsilon, d} := \{(i,j) : i \in I_{\varepsilon, d}, j \in B_1 \text{ and } |w_{i,j}| \geq \varepsilon \} \]

Now denote \( P_{\text{common}} := (P_{\varepsilon, d} \cap P_{\varepsilon, \text{opt}}) \) the set of indices common to both the optimal and \( h \)-GSG approximants and denote \( P_{\text{unique}} := ((P_{\varepsilon, d} \cup P_{\varepsilon, \text{opt}}) \setminus P_{\text{common}}) \) the set of indices that exist only in \( P_{\varepsilon, \text{opt}} \) or \( P_{\varepsilon, d} \).

We can split \( P_{\text{unique}} \) further into \( P_{\text{unique}}^{\varepsilon, \text{opt}} \) and \( P_{\text{unique}}^{\varepsilon, d} \) which are points unique to the optimal approximant and the \( h \)-GSG approximants respectively. Using this splitting and the linearity of the hierarchical interpolants \( u_{\varepsilon, \text{opt}} \) and \( u_{\varepsilon, d} \), yields

\[ u_{\varepsilon, \text{opt}} = u_{\varepsilon, \text{opt}}^{\text{unique}} + u_{\varepsilon, \text{opt}} \text{common} \quad \text{and} \quad u_{\varepsilon, d} = u_{\varepsilon, d}^{\text{unique}} + u_{\varepsilon, d} \text{common} \]

where

\[ u_{\varepsilon, \text{opt}}^{\text{unique}} = \sum_{(i,j) \in P_{\text{unique}}^{\varepsilon, \text{opt}}} w_{i,j} \cdot \Psi_{i,j}, \quad u_{\varepsilon, d}^{\text{unique}} = \sum_{(i,j) \in P_{\text{unique}}^{\varepsilon, d}} w_{i,j} \cdot \Psi_{i,j} \]

and

\[ u_{\varepsilon, d} \text{common} = \sum_{(i,j) \in P_{\text{common}}} w_{i,j} \cdot \Psi_{i,j} \]

Using these definitions we can write

\[ \| u - u_{\varepsilon, d} \|_q = \| u - u_{\varepsilon, \text{opt}} + u_{\varepsilon, \text{opt}} - u_{\varepsilon, d} \|_q \]
\[ = \| u - u_{\varepsilon, \text{opt}} + (u_{\varepsilon, \text{opt}}^{\text{unique}} + u_{\varepsilon, d} \text{common}) - (u_{\varepsilon, d}^{\text{unique}} + u_{\varepsilon, d} \text{common}) \|_q \]
\[ \leq \| u - (u_{\varepsilon, \text{opt}} + u_{\varepsilon, d}^{\text{unique}}) \|_q + \| u_{\varepsilon, \text{opt}} \|_q \quad (4.15) \]

Assuming that the adaptivity of the \( h \)-GSG method works perfectly, that is

\[ \gamma_{ij} = \| w_{i,j} \cdot \Psi_{ij} \|_q \leq \varepsilon, \quad \forall (i,j) \notin P_{\text{unique}}^{\varepsilon, d} \]
then

\[ \| u_{\varepsilon, \text{opt}}^{\text{unique}} \|_q = \left\| \sum_{(i,j) \in P_{\varepsilon, \text{opt}}^{\text{unique}}} w_{ij} \cdot \Psi_{ij} \right\|_q \leq \sum_{(i,j) \in P_{\varepsilon, \text{opt}}^{\text{unique}}} \|w_{ij} \cdot \Psi_{ij}\|_q \leq \#(P_{\varepsilon, \text{opt}}^{\text{unique}}) \cdot \varepsilon \] (4.16)

By definition of the optimal interpolant

\[ \| u - (u_{\varepsilon, \text{opt}} + u_{\varepsilon, \text{opt}}^{\text{unique}}) \|_q \leq \varepsilon \] (4.17)

Setting \( N(\varepsilon) = \#(P_{\varepsilon, \text{opt}}^{\text{unique}}) \) a we arrive at the assertion.

Theorem 4.9 states that the accuracy of the \( h \)-GSG interpolant is dependent on the number of points with cumulative \( \gamma_{ij} \leq \varepsilon \) that are not in the approximation but have \( \gamma_{ij} \approx \varepsilon \). The exact number \( N(\varepsilon) \) of these points is dependent on the smoothness of the function being approximated. The smoother the function, that is the faster the hierarchical coefficients decay, the smaller \( N(\varepsilon) \) will be.

### 4.5 Numerical Study

In this section we investigate the performance of the proposed \( h \)-GSG method when applied to a number of numerical examples. We analyze convergence, with respect to the order of the local polynomial basis and the dimensionality of the input space. The examples are drawn from the suite of functions first introduced by Genz [39] which are often used to analyze integration and interpolation methods. We investigate performance for functions of varying smoothness. Specifically we consider an infinitely differentiable function, a piecewise continuous function and finally a discontinuous function. First we consider the two-dimensional version of these functions which allows us to show the grids and sparse grid interpolants generated by the \( h \)-GSG algorithm. The proposed method is then compared against a number of other interpolation methods. Finally the utility of \( h \)-GSG is shown for high-dimensional approximation with hundreds of variables.

In the following we will consider the following three functions:

\[ f_4(\xi) = \exp \left( - \sum_{i=1}^{d} c_i^2 (\xi_i - w_i)^2 \right), \quad \xi \in [0, 1]^d \] (4.18)
4.5. NUMERICAL STUDY

\[ f_5(\xi) = \exp \left( - \sum_{i=1}^{d} c_i |\xi_i - w_i| \right), \quad \xi \in [0, 1]^d \]  

\[ f_6(\xi) = \begin{cases} 
0 & \text{if } \xi_1 > w_1 \text{ or } \xi_2 > w_2 \\
\exp \left( \sum_{i=1}^{d} c_i \xi_i \right) & \text{otherwise} 
\end{cases}, \quad \xi \in [0, 1]^d \]  

Unless otherwise stated, the coefficients \( w_i = 0.5, i = 1, \ldots, d \). The choice of \( c_i \) determines the effective dimensionality of the function and is defined differently for each problem. Although smooth, the mixed derivatives of the Gaussian function \( f_4 \) can become large and thus degrade performance if not compensated for by appropriate adaptivity. The discontinuities in functions \( f_5 \) and \( f_6 \) also degrade, with increasing magnitude, the efficiency of isotropic methods and subsequently can highlight the strengths and weaknesses of any interpolation method.

Here we will analyze convergence with respect to the following measures:

\[ \varepsilon_{L_{\infty}} = \max_{i=1, \ldots, N} |f(\xi_i) - g(\xi_i)| \]

\[ \varepsilon_{L_2} = \frac{1}{N} \left( \sum_{i=1}^{N} |f(\xi_i) - g(\xi_i)|^2 \right)^{1/2} \]

where \( f \) and \( g \) are the true function and approximation respectively. In all the following examples \( N = 1000 \). Error in the quadrature rule \( I_{\text{approx}} \) is also considered and measured by

\[ \varepsilon_{\text{integral}} = \frac{I_{\text{approx}} - I_{\text{exact}}}{I_{\text{exact}}} \]

where \( I_{\text{exact}} \) is the exact integral. Unless otherwise stated this value is calculated analytically.

4.5.1 2D Examples

Let us first consider each function defined on the unit hypercube \([0, 1]^2\). Figure 4.14 shows the function surfaces and the \( h \)-GSG grids for each of the three test functions. In each case the \( c_i = 10/2^{i+2} \), \( w_i = 0.5 \), and the functions are approximated using a linear and quadratic basis. The absolute error criterion is used and \( \varepsilon = 10^{-8} \).

Figure 4.14 (a) depicts the surface of the smooth function (4.18) and the piecewise-linear and quadratic grids used by \( h \)-GSG to resolve the function. The quadratic basis function is able to achieve comparable accuracy with significantly less number of points. The quadratic method is able to identify that the function variability is larger in the \( \xi_2 \) direction than in the \( \xi_1 \) direction and adds
more points in the $\xi_2$ direction accordingly. The linear method is able to identify the importance of each dimension too, but unlike the quadratic method the magnitude of the function derivatives has little effect on the placement of grid points.

The second example (Figure 4.14 (b)) is a piecewise continuous function (4.19) with discontinuities in its first derivatives along $\xi_1 = 0.5$ and $\xi_2 = 0.5$. The linear $h$-GSG grid concentrates grid points around the rapidly varying region associated with the discontinuous change in the derivative information. In comparison the quadratic basis requires significantly less function evaluations. The quadratic basis is able to obtain second order convergence in each of the four smooth quadrants whilst still approximating well at the discontinuities.

The difference in performance between the piecewise-linear and quadratic bases is reduced when $h$-GSG is applied to the discontinuous function (4.20), as shown in Figure 4.14 (c). The $p = 2$ method, however, is again more efficient. Again the dimension adaptive nature of the algorithm is evident. In the bottom-left quadrant, in which the function is smooth, slightly more points are invested in the steeper $\xi_2$ direction.

The three test functions (4.18)-(4.20) possess discontinuities that lie along axial directions. Let us now consider the function

$$f_l(\xi) = \frac{1}{|0.3 - \xi_1^2 - \xi_2^2|+0.1}, \quad \xi \in [0, 1]^2$$

(4.21)

which possesses a singularity that passes through both axial directions. Figure 4.5.1 depicts the surface of the function and the $h$-GSG grids obtained using a tolerance of $\varepsilon = 10^{-6}$ and linear and quadratic basis functions. The method is still able to resolve the function. Again points are concentrated along and adjacent to the discontinuity.

### 4.5.2 Increasing the Degree of the Local Polynomial Basis

Let us now consider some moderate-dimensional integrals and discuss the effect of the degree of the local polynomial basis on the efficiency of the proposed method. Let

$$c_l = \frac{1}{2^{l+2}}$$

Figures 4.16-4.18 depict the effect of decreasing the tolerance $\varepsilon$ when the proposed method is applied to the three test functions (4.18)-(4.20), respectively. Convergence is shown using a local polynomial basis of varying degree. Figure 4.16 illustrates convergence with respect to the $\mathcal{E}_{L\infty}$ measure for the smooth function $f_4$ for
4.5. NUMERICAL STUDY

(a) Equation (4.18) \( N = 6085 \ \varepsilon_{L_{\infty}} = 8.16 \times 10^{-6} \)
\( N = 833 \ \varepsilon_{L_{\infty}} = 1.63 \times 10^{-5} \)

(b) Equation (4.19) \( N = 4537 \ \varepsilon_{L_{\infty}} = 4.40 \times 10^{-6} \)
\( N = 641 \ \varepsilon_{L_{\infty}} = 2.68 \times 10^{-6} \)

(c) Equation (4.20) \( N = 9356 \ \varepsilon_{L_{2}} = 2.13 \times 10^{-7} \)
\( N = 6588 \ \varepsilon_{L_{2}} = 7.59 \times 10^{-8} \)

Figure 4.14: Surfaces of the three test functions (4.18)–(4.20) (left column) and the grids obtained using h-GSG with linear (middle column) and quadratic basis (right column) functions and \( \varepsilon = 10^{-8} \).

Figure 4.15: Surface of Equation (4.21) (left). Linear grid: \( N = 9,127, \ \varepsilon_{L_{2}} = 1.01 \times 10^{-4} \) (middle). Quadratic grid: \( N = 3,980, \ \varepsilon_{L_{\infty}} = 3.64 \times 10^{-4} \) (right)
In this case the effect of the higher-degree basis is clearly evident. The quadratic basis provides drastic improvement over the standard piecewise-linear basis and the quartic basis provides a further increase in efficiency. This result is mirrored when $h$-GSG is applied to the piecewise-continuous function (4.19) (refer to Figure 4.17). However when a jump discontinuity is present the performance of the quartic basis functions is reduced. Figure 4.18 shows the accuracy, with respect to $\varepsilon_{L_2}$, of the $h$-GSG method when applied to the discontinuous function (4.20).

The quadratic basis significantly increases the accuracy of the $h$-GSG method for smooth and discontinuous functions. This result holds true for any sparse grid approximation method, adaptive or isotropic. Higher order basis functions $p > 2$ provide further increases in the rate of convergence obtained for smooth problems, but performance is degraded for discontinuous problems. A truly $h$-$p$ adaptive scheme, which utilises high-degree basis functions in smooth regions and linear basis functions in discontinuous regions, is needed to further increase efficiency of the proposed method. The development of such a method requires the ability to determine when the entire support of a basis function intersects a discontinuity. A method for detecting discontinuities in high-dimensional spaces is presented in Chapter 5.

### 4.5.3 Efficient Termination of $h$-GSG

In Section 4.3.2 we proposed that the efficiency of the generalised sparse grid algorithm can be improved by only adding admissible indices with $r_i > \varepsilon$ to the active index set. Here we substantiate that claim. The $h$-GSG method discussed here and throughout this manuscript implements this modification.

Again let us consider the three test functions (4.18)-(4.20). But now let us investigate performance when $d = 100$ and

$$c_i = \lambda \exp\left(-\frac{35 \cdot i}{d}\right), \quad i = 1, \ldots, d$$

These coefficients range from $O(1)$ down to $O(10^{-16})$. Figures 4.19–4.21 illustrate the difference between the $h$-GSG method with and without the modification proposed in Section 4.3.2. Specifically the figures depict the error in the sparse grid interpolant as the algorithm evolves. The modification results in substantial improvement when applied to the smooth and piecewise-continuous functions. The unmodified algorithm adds many points corresponding to grid indices with
Figure 4.16: \( \varepsilon_{L_{\infty}} \) error for the \( h \)-GSG method for \( p \in \{1, 2, 4\} \) and \( \varepsilon = 10^{-10} \) for the smooth function \( f_4 \) with \( d = 10 \).

When applied to the discontinuous function the modification stops refinement earlier than the unmodified algorithm. The unmodified algorithm continues to add points belonging to grids with \( r_1 < \varepsilon \) and which contribute little to the integral of the function yet still significantly influence the accuracy of the interpolant. The points belonging to \( r_1 < \varepsilon \) mainly reside around the discontinuity. As the level of refinement increases, the contribution of these points to the integral decreases yet their effect on the interpolant may not. This effect is illustrated by Figure 4.22 which depicts the decrease in the error of the integral approximation with and without the modification when \( h \)-GSG is applied to the discontinuous function. Here it is clear that adding grids with \( r_1 < \varepsilon \) has little effect on the accuracy of the integral approximation. Also note that the effect of the termination condition decreases when the dimensionality \( d \) is small.
4.5.4 Comparison with other Methods

In this section we compare the $h$-GSG method to with the generalised sparse grid method [41], and the ASGC-HDMR method [78].

The ASGC-HDMR is bases upon the ASGC method of Ma and Zabaras [76]. The ASGC method uses the hierarchical surplus to guide refinement of the sparse grid. Numerous error indicators can be used. In an attempt to provide a fair comparison with the $h$-GSG method we use the error indicator

$$\gamma_{ij} = |w_{ij} \cdot v_{ij}|$$

If for any basis function $\Psi_{ij}$ associated with the grid point $\xi_{ij}$ the error indicator $\gamma_{ij}$ is above a pre-defined threshold $\varepsilon$, the grid point is flagged for refinement. Refinement involves generating the left and right children of the grid point $\xi_{ij}$ in each axial direction, according to the tree structure depicted in Figure 4.3. Due to the binary tree structure of the sparse grid any interior point will have $2d$ children. Boundary points associated with the level index $i$ with $k$ elements equal to 1 will have $2(d - k) + k$ children.

The ASGC method is implicitly a dimension-adaptive scheme. That is, as the
Figure 4.18: $\varepsilon_{L_2}$ error for the $h$-GSG method for $p \in \{1, 2, 4\}$ and $\varepsilon = 10^{-10}$ for the smooth function $f_6$ with $d = 10$.

The algorithm advances less importance is given to subproblems $W_i$ with “insignificant” variation as determined by the error criterion. The traditional ASGC method always evaluates all the children of an active node, but does not further refine any of the children that are in insensitive dimensions. This scheme allows for points to be invested in dimensions with no effect on the response surface (e.g. a dummy variable). Consequently, the efficiency of the method can be improved further by imposing further conditions on when the children of an active are built. With this goal Ma and Zabaras [78] combined ASGC with a dimension-adaptive scheme to only generate points in dimensions which contribute “significantly” to the mean of the function.

To adaptively select important dimensions, Ma and Zabaras consider the anchored ANOVA decomposition of a function. Specifically given a set $u \subseteq D = \{1, \ldots, d\}$ we can define the set of projections $P_u : V^{(d)} \rightarrow V^{[u]}$, by

$$P_u u(\xi_u) = u(\xi)|_{\xi = a \setminus \xi_u}$$

which projects the $d$-dimensional function onto a lower dimensional subspace. Here

$$u(\xi)|_{\xi = a \setminus \xi_u} = f(a_1, \ldots, a_{i-1}, \xi_n, a_{i+1}, \ldots, a_d)$$
Figure 4.19: The evolution of the $L_{\infty}$ error in the $h$-GSG interpolant when applied to the Gaussian function (4.18), with and without the proposed termination condition. Here $d = 100$ and $\varepsilon = 10^{-6}$. 
Figure 4.20: The evolution of the $L_\infty$ error in the $h$-GSG interpolant when applied to the Gaussian function (4.18), with and without the proposed termination condition. Here $d = 100$ and $\varepsilon = 10^{-6}$. 
Figure 4.21: The evolution of the $L_2$ error in the $h$-GSG interpolant when applied to the piecewise-continuous function (4.19), with and without the proposed termination condition. Here $d = 100$ and $\varepsilon = 10^{-6}$. 
Figure 4.22: The evolution of the $L_{\text{integral}}$ error in the $h$-GSG integral approximation when applied to the discontinuous function (4.20), with and without the proposed termination condition. Here $d = 100$ and $\varepsilon = 10^{-6}$. 
Using these projections the function $u$ can be decomposed into the finite sum

$$u(\xi) = \sum_{u \subseteq D} u_u(\xi_u)$$

(4.22)

where the sub-dimensional components can be defined recursively by

$$u_u(\xi_u) = P_u u(\xi_u) - \sum_{v \subset u} u_v(\xi_v)$$

or explicitly by

$$u_u(\xi_u) = \sum_{v \subset u} (-1)^{|u| - |v|} P_v u(\xi_v)$$

Here $\xi_u$ denotes the $|u|$-dimensional vector that contains the components of $\xi$ whose indices belong to the set $u$. The first term is the zero-th order effect which is a constant throughout the $d$-dimensional variable space.

Each sub-dimensional component $u_u$ is approximated using the ASGC method described above. Specifically

$$u_u = \sum_{|i_u| \leq t} w_{i_u} \psi_{i_u}$$

where the $n$-th element of the multi-indices $i_u$ and $j_u$ are only non-zero if $n \in u$.

In practice it is not necessary to compute all the terms $u_u$. Ma and Zabaras utilise a modification of the dimension-adaptive algorithm advocated by Griebel and Holtz [47] to adaptively select the important components. Given a subset $S$ of all indices $u \subseteq D$, the component $v$ is computed if the admissibility condition

$$u \in S \quad \text{and} \quad v \subset u \implies v \in S$$

is satisfied and if the dimension importance indicator $\eta_u \geq \varepsilon$. Here we choose

$$\eta_u = |\sum_{|i_u| \leq t} w_{i_j} \cdot \psi_{i_u} j_u|$$

The dimension-adaptive algorithm of ASGC-HDMR is outlined in Algorithm 3. Here we use the sub-dimensional components contained in $S$ and $R$. The components in $R$ were deemed to contribute negligibly to the integral. However, since we had to use computational resources to construct these components to determine their importance, they can be used in the final approximation with no extra effort. This will increase the accuracy of the final approximation slightly.

The adaptive procedures of the $h$-GSG and ASGC methods construct the sparse index set $I$ differently. $h$-GSG proceeds greedily and chooses index sets...
4.5. NUMERICAL STUDY

Algorithm 3 Dimension adaptive construction of the index set $S$

$S := \{\emptyset\}$
$\mathcal{A} := \{\{1\}, \ldots, \{d\}\}$
$\mathcal{R} := \{\}$
$d_s := 1$

while (True) do
  compute $\eta_u$ for all $u \in \mathcal{A}$ for which $\eta_u$ has not been computed
  for ($u \in \mathcal{A}$) do
    if $\gamma_u \geq \varepsilon$ then
      $S := S \cup \{u\}$
    else
      $\mathcal{R} := \mathcal{R} \cup \{u\}$
    end if
  end for
  $d_s := d_s + 1$
  update $\mathcal{A}$ according to the admissibility criterion for $|u|_1 = d_s$
end while

only if they are admissible, whereas the ASGC procedure adds indices level by level. Figure 4.23 depicts the different index sets generated by the ASGC-HDMR and $h$-GSG methods and the resulting grids for a two-dimensional problem. Here we have assumed that any grid not in the left index set has points whose error indicator $\gamma_j < \varepsilon$. Grey boxes represent grid indices that contain at least one point with $\gamma_j \geq \varepsilon$ and black boxes are indices that only contain points with $\gamma_j < \varepsilon$. White boxes represent grid indices only used by the ASGC-HDMR method. In the example shown, two such grid indices are present. The exact number of missing indices and associated grid points is problem dependent.

Figure 4.24 shows the five steps taken by the ASGC-HDMR algorithm to obtain the index set depicted in Figure 4.23. In the example shown, both dimensions are deemed important by ASGC-HDMR and thus we only show the construction of the component $u_{(1,2)}$. Striped boxes indicate indices to be refined, black boxes are grids that only contain points with $\gamma_j < \varepsilon$, and grey boxes represent indices already in the sparse grid.

Figures 4.24 and 4.23 suggest that given a predefined tolerance and a subdimensional problem for which all dimensions under consideration are important, the ASGC method will typically require more points than the $h$-GSG method. To clarify this point further, consider the refinement of active index sets for
Figure 4.23: A possible admissible index set generated by the $h$-GSG algorithm (left) and the corresponding index set generated by ASGC (right). Grey boxes represent grid indices that contain at least one point with $\gamma_{ij} \geq \varepsilon$ black boxes are indices that only contain points with $\gamma_{ij} < \varepsilon$. White boxes represent grid indices only used by the ASGC-HDMR method. Circles represent points with $\gamma_{ij} \geq \varepsilon$ and crosses depict points with $\gamma_{ij} < \varepsilon$. 
4.5. NUMERICAL STUDY

Figure 4.24: Five steps of the ASGC method. Striped boxes indicate indices to be refined, black boxes grids that only contain points with $\gamma_{ij} < \varepsilon$, and grey boxes represent indices already in the sparse grid.

the $h$-GSG and ASGC methods for a $d$-dimensional sub-problem for which all dimensions are important. This is the situation presented in Figure 4.23. ASGC and $h$-GSG both define an active index as any index $i$ with at least one point $\xi_{ij}$ with an error indicator that satisfies $\gamma_{ij} \geq \varepsilon$. Given a particular active index $i$, the $h$-GSG method will generate 1 to $d$ new grid indices depending on the existence of backwards neighbours $i - e_j, j = 1, \ldots, d$ according to the admissibility criteria (4.11). The ASGC method, however, will always add $d$ new indices. So given any sub-dimensional component of the ANOVA decomposition, $h$-GSG will typically construct a smaller index set and thus use less points.

To illustrate that the differences outlined above do indeed result in higher efficiency, consider the three test functions (4.18)-(4.20) with

$$c_i = \frac{1}{2^i+1}, \quad i = 1, \ldots, d$$

Figures 4.25-4.27 show the error of the $h$-GSG, ASGC-HDMR and GSG methods when applied to each of these three problems when $d = 10$. Specifically the plots depict the change in error associated with decreasing values of $\varepsilon$. Here we wish to compare the performances of the three types of adaptivity and accordingly restrict our attention to piecewise-linear decompositions of the spaces $w_i$. In
CHAPTER 4. H-ADAPTIVE SPARSE GRIDS

the smooth case (Figure 4.25) the $h$-adaptivity of the proposed method provides no improvement on the generalised sparse grid method. It is well documented that local adaptivity provides little improvement for smooth problems, e.g. [16]. For the piecewise-continuous function the GSG method also performs comparably to the $h$-GSG method (see Figure 4.25). However the $h$-GSG method is much more efficient than its GSG counterpart when the function being approximated is discontinuous, as shown in Figure 4.27. If the GSG method determines that the approximation is poor it will add points everywhere along the dimension selected for refinement. This is efficient when the function is smooth but extremely inefficient if only a small region of the input space is contributing to the function variability. Thus for the piecewise continuous function and discontinuous function, the $h$-GSG method is much more efficient than GSG.

Figure 4.25: Convergence rates of the $\varepsilon_{L_{\infty}}$ error for the Gaussian function $f_4$ with $d = 10$ and $p = 1$. Convergence is shown for the $h$-GSG, ASGC-HDMR, and GSG approximation methods. Each point corresponds to decreasing values of $\varepsilon$. These values start at $10^{-2}$ and successively decrease by a factor of 10.

The ASGC-HDMR method even in 10 dimensions is the least efficient of the three methods compared, for the smooth and piecewise-continuous functions. However the ASGC-HDMR variant performs equally well as $h$-GSG when applied
Figure 4.26: Convergence rates of the $\varepsilon_{L_2}$ error for the piecewise continuous function $f_5$ with $d = 10$ and $p = 1$. Convergence is shown for the $h$-GSG, ASGC-HDMR and GSG approximation methods. Each point corresponds to decreasing values of $\varepsilon$. These values start at $10^{-2}$ and successively decrease by a factor of 10.
Figure 4.27: Convergence rates of the $\varepsilon_{L_2}$ error for the discontinuous function $f_6$ with $d = 10$ and $p = 1$. Convergence is shown for the $h$-GSG, ASGC-HDMR and GSG approximation methods. Each point corresponds to decreasing values of $\varepsilon$. These values start at $10^{-2}$ and successively decrease by a factor of 10.
to the discontinuous function. In this moderate dimensional setting the accuracy of the ASGC-HDMR method is comparable to the \( h \)-GSG method. However as the dimensionality increases, the disparity between these two methods also increases. Now let

\[
c_i = \exp\left(-\frac{35 \cdot i}{d}\right)
\]

Figures 4.28-4.30 show the error of the \( h \)-GSG and ASGC-HDMR for \( d = 100 \). For all three test problems the linear \( h \)-GSG method is significantly more efficient than ASGC-HDMR. The relative accuracy of the \( h \)-GSG method is increased further if quadratic or quartic basis functions are used.

Figure 4.28: Convergence rates of the \( \varepsilon_{L_{\infty}} \) error for the piecewise continuous function \( f_4 \) with \( d = 100 \). Convergence is shown for \( h \)-GSG, with \( p = 1, 2, 4 \), and ASGC-HDMR, with \( p = 1 \) for decreasing values of \( \varepsilon \). Each point corresponds to decreasing values of \( \varepsilon \). These values start at \( 10^{-2} \) and successively decrease by a factor of 10.

Aside from the improvement in the rate of convergence the \( h \)-GSG has an additional advantage over ASGC-HDMR. The greedy nature of the proposed algorithm means that the error in the approximation during the evolution of the sparse grid algorithm decays more quickly than its ASGC-HDMR counterpart. This is illustrated in Figures 4.31–4.33. For a fixed computational budget, which
Figure 4.29: Convergence rates of the $\varepsilon_{L_{\infty}}$ error for the piecewise continuous function $f_5$ with $d = 100$. Convergence is shown for $h$-GSG, with $p = 1, 2, 4$, and ASGC-HDMR, with $p = 1$ for decreasing values of $\varepsilon$. Each point corresponds to decreasing values of $\varepsilon$. These values start at $10^{-2}$ and successively decrease by a factor of 10.
Figure 4.30: Convergence rates of the $\varepsilon_{L_2}$ error (a) for the discontinuous function $f_6$ with $d = 100$. Convergence is shown for $h$-GSG, with $p = 1, 2, 4$, and ASGC-HDMR, with $p = 1$ for decreasing values of $\varepsilon$. Each point corresponds to decreasing values of $\varepsilon$. These values start at $10^{-2}$ and successively decrease by a factor of 10.
may not be able to fully resolve the function being considered, the proposed algorithm will choose the best set of points that give the smallest approximation error.

![Graph showing the error in the h-GSG and ASGC-HDMR interpolants as the respective algorithms evolve when applied to the smooth function (4.18) with $\epsilon = 10^{-6}$.](image)

Figure 4.31: Error in the h-GSG and ASGC-HDMR interpolants as the respective algorithms evolve when applied to the smooth function (4.18) with $\epsilon = 10^{-6}$.

### 4.5.5 High-Dimensional Interpolation

In Section 4.5.4 we used h-GSG for interpolation for a 100-dimensional problem. In this section we show that the proposed method can be applied to much higher-dimensional problems. Again consider the discontinuous function (4.20) which is the most difficult function to approximate of the three test functions used thus far. Again let

$$c_i = \lambda \exp\left(-\frac{35 \cdot i}{d}\right)$$

(4.23)

But now we have introduced the parameter $\lambda$ which controls the effective dimensionality of the function. Figure 4.34 depicts the coefficient decay for various values of $d$ and $\lambda = 1$. Table 4.2 shows the number of function evaluations required to approximate $f_6$ and the resulting relative error in the approximated
Figure 4.32: Error in the $h$-GSG and ASGC-HDMR interpolants as the respective algorithms evolve when applied to the piecewise-continuous function (4.19) with $\varepsilon = 10^{-6}$. 
Figure 4.33: Error in the $h$-GSG and ASGC-HDMR interpolants as the respective algorithms evolve when applied to the discontinuous function (4.20) with $\varepsilon = 10^{-6}$. 
4.5. NUMERICAL STUDY

Figure 4.34: Coefficients $c_i = \exp\left(-\frac{35 \cdot i}{d}\right)$ for $d = 100$ to 700.

integral when $\varepsilon = 10^{-4}$, quadratic basis functions are used, and $\lambda = 1$. An analytical expression for the integrand can be obtained easily due to the exponential nature of the function. Arbitrary precision arithmetic was used to evaluate the numerical value of the reference integrals. Due to the large range of values that $f_6$ can take in high dimensions we use the relative error indicators

$$ r_1 = \left| \frac{\sum_{j \in B_i} w_{i,j} \cdot v_{i,j}}{w_{0,0} \cdot v_{0,0}} \right|, \quad \gamma_i = \left| \frac{w_{i,j} \cdot v_{i,j}}{w_{0,0} \cdot v_{0,0}} \right| $$

to respectively guide difference space selection and local refinement. The $h$-GSG algorithm is terminated when

$$ \left| \frac{\sum_{i \in \mathcal{I}} r_1}{w_{0,0} \cdot v_{0,0}} \right| < \varepsilon $$

where $r$ is the global error indicator used in Algorithm 1. An error of the order $10^{-2}$ is achieved for up to 700 dimensions using less than 300,000 function evaluations.

The accuracy of the integral approximation decays with increasing dimensionality. This is likely caused by the particular error indicators ($\gamma_{i,j}$ and $r_1$) used to guide adaptivity. At the moment a point $\xi_{i,j}$ is refined if $\gamma_{i,j} \geq \varepsilon$ and a grid
index $i$ is flagged for refinement only if $r_i \geq \varepsilon$. This approach works well when $d < 400$ but could be improved upon when the dimensionality is higher. By excluding points if they have $\gamma_{ij}$ less than the desired accuracy $\varepsilon$ we are potentially ignoring a significant number of points whose combined contribution to the integral is greater than $\varepsilon$. As the dimensionality increases more and more points will be excluded from consideration thereby causing the accuracy of the approximant to decrease. This remark is consistent with Theorem 4.9 which states that the accuracy of the $h$-GSG approximation is dependent on the number of points with $\gamma_{ij}$ close to $\varepsilon$. As the number of these points increases the accuracy of the approximation decreases.

The decrease in accuracy depicted in Table 4.2 could be addressed by utilising more appropriate error criteria than those used here. One simple approach would be to refine when $\gamma_{ij} \geq C_{\gamma} \cdot \varepsilon$ and $r_i \geq C_r \cdot \varepsilon$, where $C_{\gamma}$ and $C_r$ are positive constants less than one. Many other forms of error indicators can also be used. The construction of efficient and robust error indicators is problem dependent and must be based upon the properties of the function under consideration. If no information on the function is available, we have shown that the error indicators used here will still perform well.

Table 4.2: Errors in the $h$-GSG approximation of (4.20) for $d = 100$ to 700. $\varepsilon = 10^{-5}$

<table>
<thead>
<tr>
<th>$d$</th>
<th>$N$</th>
<th>$\varepsilon_{\text{integral}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>3,376</td>
<td>$3.81 \times 10^{-4}$</td>
</tr>
<tr>
<td>200</td>
<td>12,488</td>
<td>$1.67 \times 10^{-3}$</td>
</tr>
<tr>
<td>300</td>
<td>31,533</td>
<td>$1.71 \times 10^{-4}$</td>
</tr>
<tr>
<td>400</td>
<td>62,404</td>
<td>$8.44 \times 10^{-5}$</td>
</tr>
<tr>
<td>500</td>
<td>109,356</td>
<td>$4.57 \times 10^{-3}$</td>
</tr>
<tr>
<td>600</td>
<td>176,842</td>
<td>$7.97 \times 10^{-3}$</td>
</tr>
<tr>
<td>700</td>
<td>269,665</td>
<td>$1.68 \times 10^{-2}$</td>
</tr>
</tbody>
</table>

Here we note that the rate of convergence of the $h$-GSG method is governed by the implicit weighting of the importance of each dimension. In this case the importance is controlled by the coefficients $c_i$. To illustrate this dependence, Table 4.3 shows the efficiency of $h$-GSG as the magnitude of the coefficients is increased. As the "importance" of each dimension increases the number of sub-dimensional components increases. In this case an increase in $c_i$ also increases
4.6 MULTI-DIMENSIONAL STOCHASTIC APPLICATION

the function variability which also requires additional points to achieve a set level of accuracy.

Table 4.3: Errors in the $h$-GSG approximation of (4.20) for increasing dimension importance $\varepsilon = 10^{-6}$. Importance is increased by increasing $\lambda$ in Equation (4.23).

<table>
<thead>
<tr>
<th>$\lambda$</th>
<th>$N$</th>
<th>$\varepsilon_{\text{integral}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9,226</td>
<td>$1.66 \times 10^{-4}$</td>
</tr>
<tr>
<td>2.5</td>
<td>34,977</td>
<td>$2.96 \times 10^{-5}$</td>
</tr>
<tr>
<td>5</td>
<td>175,201</td>
<td>$6.53 \times 10^{-4}$</td>
</tr>
<tr>
<td>7.5</td>
<td>659,368</td>
<td>$1.93 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

4.6 Multi-dimensional Stochastic Application

The storage of carbon-dioxide in sub-surface geological formations has been proposed as a means of reducing the concentration of green-house gases in the atmosphere. The feasibility of carbon sequestration is dependent on the permeability of the medium in which it is to be stored. Accurate and computationally efficient modelling of flow through porous media is important for predicting the storage capacity of various underground reservoirs and the potential for long-term capture.

In this section we consider a model simulating flow through porous media. This model is used to predict the steady state velocity field produced by the time-invariant inflow and outflow of water in a two-dimensional spatial domain. This velocity field is used to model the advection of a tracer through the domain. The time it takes the tracer to reach a given measurement site from the point of release is referred to as the breakthrough time. Unlike other physical quantities, such as pressure, the measurement of breakthrough times in the field is relatively straightforward.

Using Bayesian inference the field measurements of breakthrough time and the model can be used to estimate the permeability of the medium through which the water is flowing. Approximation of the permeability can help the utility of the site under investigation for carbon sequestration.

Bayesian inference can require millions of model evaluations. The model considered here takes approximately a second to run. Consequently the use of a
surrogate that can be constructed efficiently and evaluated quickly could potentially decrease inference process significantly. The increase in efficiency gained by using surrogates in Bayesian parameter estimation, in place of their more time-consuming physical models, has been shown previously [77, 79]. With the aim of estimating permeability, in this section we use \( h\)-GSG to construct a surrogate of a model simulating flow through porous media. Note that here we focus on the construction of the surrogate and not the inference process. Inference of the permeability from field data is the subject of current research.

### 4.6.1 Model Formulation

Consider the following model which simulates flow through a porous medium, in the rectangular domain \( D = [0, 1.5] \times [0, 1] \), with an underlying permeability field \( K(x) \) represented by a spatially varying random field. The random permeability field is represented using the Karhunen-Loève (KL) expansion (2.3) introduced in Section 2.2.2. The KL expansion is derived from a separable exponential covariance function

\[
\text{Cov}(x, y) = \sigma^2 \exp\left(-\frac{|x_1 - y_1|}{L} - \frac{|x_2 - y_2|}{L}\right)
\]

where \( L \) is the correlation length and \( \sigma^2 \) is the variance of the random field. The resulting KL expansion is given by

\[
X(x, \xi) = \log(K(x, \xi)) = \sum_{n=1}^{d} \sqrt{\lambda_n} \phi_n(x) \xi_n, \quad x \in D
\]

(4.24)

Here we assume that the ranges of the random variables \( \xi \) are contained in the interval \([-1, 1]\). The eigenvectors \( \phi_n \) and the corresponding eigenvalues \( \lambda_n, n = 1, \ldots, d \) are obtained numerically. Two realisations of the permeability field are shown in Figure 4.35.

In two-dimensions, the velocity field \( u(x, \xi) = \{u(x, \xi), v(x, \xi)\} \), of an incompressible fluid, can be approximated by

\[
\nabla \cdot u(x, \xi) = f(x), \quad x \in D, \quad \xi \in \Gamma
\]

(4.25)

\[
u(x, \xi) = -K(x, \xi) \cdot \nabla p(x, \xi), \quad x \in D, \quad \xi \in \Gamma
\]

(4.26)

where \( p(x, \xi) \) represents the pressure field within \( D \) and the source term \( f(x) \) models the inflow and outflow at designated well locations. Equation (4.25) is based upon the conservation of mass and equation (4.26) is Darcy’s Law which
states that the fluid velocity is proportional to the fluid pressure. Once the velocity field has been determined, the concentration of the tracer is modelled using the continuity equation

$$\frac{\partial}{\partial t} c(x, \xi) + \frac{\partial}{\partial x_1} F + \frac{\partial}{\partial x_2} G,$$

(4.27)

where \( \{F,G\} = \{u(x, \xi)c(x, \xi), v(x, \xi)c(x, \xi)\} \) are the fluxes in the \( x_1 \) and \( x_2 \) directions respectively.

We wish to model flow driven solely by the source term \( f(x) \) and hence set

$$u(x) \cdot \mathbf{n} = 0, \quad x \in \partial D$$

where \( \mathbf{n} \) is a vector normal to the boundary \( \partial D \). This no-flow condition prevents water from entering or leaving the domain through the boundaries.

The spatial domain \( D \) (Figure 4.37) is decomposed into a rectangular mesh with 30 cells in the \( x_1 \) direction and 20 cells in the \( x_2 \) direction. The governing equations (4.25)–(4.26) are then solved using the two-point flux approximation (TPFA) finite-volume method. Flow is driven through this domain by pumping water in through the cell at the bottom-left corner (cross) of the domain and extracted water from the top-right corner cell (circle). No water is allowed to pass through the external boundaries. The concentration \( c \) is defined at cell-centers while the velocities exist on a staggered mesh, as shown in Fig. 4.36.

Upwinded finite differences are used to approximate the spatial derivatives in the \( x_1 \) and \( x_2 \) direction, i.e.,

$$F_{i+1/2, j} = \begin{cases} u_{i+1/2, j}c_{i,j} & \text{if } u_{i+1/2, j} \geq 0 \\ u_{i+1/2, j}c_{i+1,j} & \text{otherwise} \end{cases}$$
Figure 4.36: A portion on the mesh, along with the collocations for the concentration \( \mathcal{C} \), and the velocity \( \mathbf{u} = \{u_1, u_2\} \).
4.6. **MULTI-DIMENSIONAL STOCHASTIC APPLICATION**

An identical upwinding approach is adopted for calculating $G_{i,j+1/2}$. The partial (spatial) differences in Eq. 4.27 are evaluated at $(i, j)$ using central differences applied to $F_{i+1/2,j}$ and $G_{i,j+1/2}$. The equation is advanced in time using a time-step controlled fourth-order Runge-Kutta algorithm.

![Figure 4.37: the computational domain $D$ used to solve equations (4.25), (4.26) and (4.27). Dots represent the sites at which breakthrough times are measured. The cross marks the cell through water is pumped into the domain and the circle represents the cell where water is extracted.](image)

### 4.6.2 Results

In this section we wish to illustrate the ability of the proposed $h$-GSG method to construct an efficient high-dimensional surrogate of the porous media flow model (4.25)–(4.26). With this aim we construct interpolants of the speed field

$$s(x, \xi) = (u(x, \xi)^2 + v(x, \xi)^2)^{1/2}$$

for three different correlation lengths $L \in \{1, 0.5, 0.3\}$.

The correlation length determines the variability of the permeability field and the effective dimensionality of the parametric space. As the correlation length decreases the decay rate of the eigenvalues $\lambda_i$ decreases as does the importance of the associated variable $\xi_i$. The magnitude of the eigenvalues for the three different correlation lengths are shown in Figure 4.38.

For each correlation length $L \in \{1, 0.5, 0.3\}$ we truncate terms in the KL expansion (4.24) with $\lambda_i = O(10^{-14})$. The resulting KL expansions have $d = 35$, 60, and 100 terms, respectively. Setting $\varepsilon = 10^{-6}$ we use $h$-GSG to construct interpolants over the input space $\Gamma$ for each of the 600 cells used to discretise the
spatial domain $D$. The decoupled nature of stochastic collocation means that one could simply construct an independent sparse grid in each cell, however greater efficiency can be achieved by considering the cells simultaneously. Every time the true model is evaluated a complete time and spatial history becomes available. This information is stored at the corresponding point in the sparse grid. The decision to refine is based upon the worst case error criteria

$$r_i = \max_{x \in \mathcal{X}, t \in T} r_i(x, t) = \max_{x \in \mathcal{X}, t \in T} \sum_{j \in B_i} w_{ij}(x, t) \cdot v_{ij}$$

$$\gamma_{ij} = \max_{x \in \mathcal{X}, t \in T} \gamma_{ij}(x, t) = \max_{x \in \mathcal{X}, t \in T} |w_{ij}(x, t) \cdot v_{ij}|$$

where $\mathcal{X}$ and $T$ are the set of spatial cells and time steps used by the numerical method described in Section 4.6.1. Thus a point is identified for refinement if the approximation is poor in at least one of the spatial cells during at least one time step.

Figure 4.39 depicts the mean speed field and the variance in the speed field for the three different correlation lengths $L \in \{1, 0.5, 0.3\}$. The speed fields are obtained by plotting the mean and variance at each spatial cell. The velocity field is a steady state and thus time can be ignored. Decreasing the correlation length has no visible effect on the mean speed field. But decreasing the permeability field does increase the variance in the speed field. The flow speeds are largest in
the regions adjacent to the source and sink cells. The smallest mean speeds and variance in speed are present in the middle of the domain. The number of grid points (model evaluations) required to resolve these fields for the three correlation lengths $L \in \{1, 0.5, 0.3\}$ are 399, 2,382, and 68,788 respectively. The $\varepsilon_{L_2}$ error in the three approximations are $3.23 \times 10^{-6}$, $8.48 \times 10^{-6}$, and $5.99 \times 10^{-5}$. Even when $L = 0.3$ and 41,272,800 degrees of freedom are necessary to construct the approximation, the $h-GSG$ method obtains a high-degree of accuracy. Here we note that all the computations in this thesis are performed on a single desktop CPU. Much larger problems could be addressed if parallel computing were to be employed.

We can also use $h$-GSG to construct a surrogate for estimating breakthrough times at 20 sites distributed throughout the spatial domain $D$. The location of these sites are shown in Figure 4.37. After releasing the tracer at the point of inflow, the tracer is deemed to reach a site $x$ when the concentration $c(x) = 0.05$. The simulation is run for 40 days. If the tracer has not reached a site the breakthrough time is set to 40. This introduces a discontinuity in the first and higher-order derivatives which necessitates the use of a $h$-adaptive scheme, such as the $h$-GSG method.

We use $h$-GSG to construct surrogates for $L = 1.0$ and $L = 0.5$ and as before set $\varepsilon = 10^{-6}$ and $d = 35$ and $d = 60$ for these two cases. The resulting grids contain 3,127 and 378,102 points, respectively. Approximating breakthrough times is much more difficult than approximating speed. However these results show that obtaining a surrogate is possible. Future research will focus on using these surrogates in Bayesian parameter estimation to significantly reduce the computational power required to produce accurate estimates of the KL expansion variables from observational data. Given that the posteriori estimates of the input densities obtained by Bayesian inference are often heavily influenced by the prior densities it will be interesting to investigate what accuracy in the surrogate has a measurable effect on the final densities obtained.

4.7 Conclusions

This chapter presented an $h$-adaptive generalised sparse grid ($h$-GSG) method for interpolating high-dimensional functions with discontinuities. The proposed algorithm extends and improves upon existing approaches by combining the strengths of the generalised sparse grid algorithm and hierarchical surplus-guided
Figure 4.39: Mean speed field (left column) and the variance in the speed field (right column) for three different correlation lengths $L$. 

(a) $L = 1.0$

(b) $L = 0.5$

(c) $L = 0.3$
4.7. CONCLUSIONS

\( h \)-adaptivity.

The underlying generalised sparse grid algorithm greedily selects the subspaces that contribute most to the variability of a function. The hierarchical surplus of the points within each subspace is used as an error criterion for \( h \)-refinement with the aim of concentrating computational effort within rapidly varying or discontinuous regions. This approach limits the number of points that are invested in ‘unimportant’ subspaces and regions within the high-dimensional domain.

A high-degree basis is used to obtain a high-order method that, given sufficient smoothness, performs significantly better than the traditional piecewise-linear basis. When discontinuities are present in the function surface or its derivatives, performance deteriorates. However, it was shown numerically that even in such situations the quadratic basis will still result in higher-rates of convergence than that achieved by using piecewise-linear interpolation.

The importance of the model (function) variables which are the subject of UQ analysis are governed by natural yet unknown weights. In these cases, the proposed method can utilise this implicit weighting to determine and restrict effort to the effective dimension of the model. This property allows the \( h \)-GSG method to be applied to non-smooth functions with as many as 700 variables.
Chapter 5

Discontinuity Detection

5.1 Introduction

The sparse grid collocation scheme, presented in Chapter 4, employs localised bases to limit and localize Gibbs type phenomenon is not the only approach that can be utilized to deal with low regularity. Multi-element formulations [3, 8, 32, 31, 117] have proven an effective alternative. These methods discretize the input domain into disjoint regions of smoothness, on which high-order global polynomial methods are employed. Although such discretization is straightforward conceptually, it is numerically challenging. Current techniques adaptively discretize the parametric domain into disjoint sets of multi-dimensional rectangles. This is inefficient when the discontinuities do not lie parallel to the coordinate axes.

Efficient identification and location of discontinuities is extremely useful. Ideally, if knowledge of the location of the discontinuities is available, one can avoid the tensor structure of the sub-domain decomposition and seek to discretize the high-dimensional domain into a minimal number of sub-domains whose interfaces lie exactly along the discontinuities.

An initial attempt to identify and locate discontinuities in high-dimensional spaces was made in [4], where the polynomial annihilation edge detection method [5] was used. The polynomial annihilation edge detection is a high-order method and has certain advantages over the traditional methods [20, 106, 37]. It was originally developed for edge detection in one and two dimensions but can be extended to higher dimensions by applying the algorithm dimension-by-dimension directly to a gPC approximation [4]. However, the procedure proposed in [4] relies on examination of the function values on (local) tensor grids and thus restricts the
dimensionality that can be handled.

In this chapter we present a much improved algorithm over that in [4]. The proposed method is closely related to sparse grid stochastic collocation [129]. Sparse grids [11, 40, 41] have been frequently used to provide approximations of high-dimensional functions, especially in stochastic simulations in recent years. Refer to Chapter 4 for a detailed discussion.

The primary contribution of this chapter is the development of a highly efficient method for detecting the presence of discontinuities in high dimensional spaces, based on nodal solutions of sparse grids. If a discontinuity is detected, the method can accurately identify the location and resolve the geometry of the discontinuity by employing a locally-adaptive sparse grid algorithm.

In many practical cases, the discontinuity resides on a manifold of smaller dimensionality. In such cases, the proposed algorithm becomes “optimal”, in the sense that the number of function evaluations grows linearly as the dimensionality of the space increases. The present algorithm can be applied to functions with hundreds of variables which represents a significant improvement over the earlier work in [4].

This chapter proceeds by first introducing one-dimensional polynomial annihilation. This one-dimensional formulation is then extended to higher dimensions by adopting a similar adaptive procedure to that employed by the $h$-GSG method presented in Chapter 4. This procedure identifies the lower-dimensional manifold upon which the discontinuities reside. A simple method for domain classification is then outlined. The remainder of the chapter is devoted to the investigation of the numerical properties of this method for a number of different test functions. We conclude by applying the proposed method to a model simulating the genetic toggle switch in Escherichia coli [37].

\section*{5.2 Polynomial Annihilation Edge Detection}

Let us first consider a piecewise continuous function $f(x) : [-1,1] \rightarrow \mathbb{R}$ in one dimensional (random) space known only on the set of discrete points $^1$

$$S = \{x_1, x_2, ..., x_{Q^{(1)}}\} \subset [-1,1].$$  \hspace{1cm} (5.1)

$^1$There are $Q^{(1)}$ reconstruction points in one dimension. We note, however, that in higher dimensions, $Q^{(d)}$ may not be related to the total number of quadrature points in the stochastic collocation method.
5.2. POLYNOMIAL ANNIHILATION EDGE DETECTION

Assume that \( f \) has well-defined one-sided limits, \( f(x^\pm) \), at any point \( x \) in the domain. We denote by \( J \) the set of the points of discontinuity of \( f \), that is, \( J = \{ \xi : -1 < \xi < 1 \} \), where \( \xi \) is a jump discontinuity point in \( f \). The local jump function is defined as

\[
[f](x) = f(x^+) - f(x^-) = \begin{cases} 
0, & \text{if } x \neq \xi, \\
[f](\xi), & \text{if } x = \xi.
\end{cases}
\]

Hence if \( f \) is continuous at \( x \), the jump function \([f](x) = 0\); if \( x \) is a point of discontinuity of \( f \), then \([f](x)\) is equal to the jump value there.

The polynomial annihilation edge detection method, introduced in [5], seeks an approximation to \([f](x)\) that converges rapidly to zero away from the jump discontinuities. It can be described as follows: Assume \( x \) is a point inside the domain, i.e., \( x \in (-1, 1) \). For a given positive integer \( m < Q^{(1)} - 1 \), we choose a local stencil

\[
S_x = \{ x_j | x_j \in S \} = \{ x_0, \ldots, x_m \}
\]

of the nearest \( m + 1 \) grid points around \( x \). Let \( \{ p_i \}_{i=0}^m \) be a basis for \( \Pi^1_m \), the univariate polynomial space of degree up to \( m \). The polynomial annihilation edge detection method is given by

\[
L_m f(x) = \frac{1}{q_m(x)} \sum_{x_j \in S_x} c_j(x) f(x_j),
\]

where the coefficients \( c_j(x) \) are chosen to annihilate polynomials of degree up to \( m - 1 \) and are determined by solving the linear system

\[
\sum_{x_j \in S_x} c_j(x) p_i(x_j) = p_i^{(m)}(x), \quad \forall \, i = 0, \ldots, m.
\]

Here \( p_i^{(m)}(x) \) denotes the \( m \)th derivative of \( p_i(x) \). Notice that the solution to (5.5) exists and is unique. Next we define

\[
S_x^+ = \{ x_j \in S_x | x_j \geq x \} \quad \text{and} \quad S_x^- = S_x \setminus S_x^+.
\]

The normalization factor in (5.4), given by

\[
q_m(x) = \sum_{x_j \in S_x^+} c_j(x),
\]

ensures that \( L_m f(x) \) has correct value at the jump discontinuities. Note that (5.7) is non-zero by design. In [5] it was shown that if the maximum separation \( h(x) \) is defined as

\[
h(x) = \max \{|x_i - x_{i-1}| : x_{i-1}, x_i \in S_x \},
\]

(5.8)
then (5.4) satisfies the following property:

\[
L_m f(x) = \begin{cases} 
[f](\xi) + O(h(x)), & \text{if } x_{j-1} \leq \xi, x \leq x_j, \\
O(h^{\min(m,k)}(x)), & \text{if } f \in C^k(I_x), k > 0.
\end{cases}
\] (5.9)

Here \(I_x\) is the smallest closed interval such that \(S_x \subset I_x\). Hence the polynomial annihilation edge detection method converges to \([f](x)\) with a rate depending on \(m\) and the local smoothness of \(f\).

We pause to note two important distinctions between the polynomial annihilation edge detection method, (5.4), and more traditional edge detection methods \([20, 106]\). The polynomial annihilation edge detection method is a high order reconstruction of the jump function, \([f](x)\) defined in (5.2). Other techniques more commonly used in image processing are edge detectors. Specifically, they attempt to identify the set of edges \(J\) with regard to certain thresholds that are usually determined from some underlying assumptions about the particular image (typically digital) and outside influences (e.g. noise). The high-order design of (5.4) is well suited when the underlying structure of the piecewise-smooth function has some variability. Also, fewer points are needed in each direction to resolve the corresponding jump function, which can dramatically reduce computational costs. The second critical advantage of the polynomial annihilation edge detection method over traditional techniques is that it does not require a uniform point distribution. This is particularly useful when combined with adaptive methods which do not possess an equidistant point distribution.

The order \(m\) in (5.4) affects the resolution of the jump function both close to and away from any discontinuities. Small \(m\) might cause a steep gradient to be misidentified as an edge (due to low resolution). On the other hand, the inherent nature of high order polynomial approximation causes oscillations to occur in the vicinity of the discontinuities when \(m\) is large. These oscillations may also be misinterpreted as edges. To prevent inaccuracies due to either of these problems, the minmod function, which is typically utilized in flux limiting methods to reduce oscillations when solving numerical conservation laws, was used in \([5]\) to enhance the performance of the polynomial annihilation method.\(^2\) Specifically, we apply

\[
MM(L_m f(x)) = \begin{cases} 
\min_{m \in \mathcal{M}} L_m f(x), & \text{if } L_m f(x) > 0, \quad \forall m \in \mathcal{M}, \\
\max_{m \in \mathcal{M}} L_m f(x), & \text{if } L_m f(x) < 0, \quad \forall m \in \mathcal{M}, \\
0, & \text{otherwise},
\end{cases}
\] (5.10)

where \(\mathcal{M} \subset \mathbb{N}\) of positive integers. The minmod function controls the oscillations

\(^2\)The minmod technique was introduced in the context of edge detection in \([12, 38]\).
while still maintaining a high order of convergence away from the jump discontinuities. It furthermore reduces the need for outside thresholding. Examples of its effectiveness can be found in [5].

Finally, we would like to point out that the polynomial annihilation edge detection method was originally developed to determine multi-dimensional jump discontinuities from randomly distributed grid points. However, the sparse grid collocation method uses structured grids in each dimension. Hence to simplify programming and increase computational efficiency, we choose to use a dimension by dimension approach. As an additional advantage, the method easily lends itself to parallel processing.

5.3 Edge Tracking

The one-dimensional polynomial annihilation method presented in Section 5.2 can easily be extended to two dimensions using a triangulation procedure which identifies jump discontinuities from its surrounding points (triangle vertices) [5]. In principle, similar element-based techniques could be used for \( d > 2 \), but in practice such algorithms are very complicated and difficult to implement. To overcome this difficulty, Archibald et al. [4] applied one-dimensional polynomial annihilation to tensor product grids in a line-by-line fashion. Due to the curse of dimensionality inherent in tensor product formulations such an approach was limited to small and moderate number of dimensions. This problem was alleviated slightly by applying the detection algorithm on a gPC stochastic collocation approximation that can be often evaluated much more quickly than the underlying function being represented.

Here we attempt to extend this method by improving the refinement algorithm used to resolve discontinuities, whilst maintaining the tensor product structure. Starting with the hypercube \([-1, 1]^d\), Archibald’s edge detection method identifies any hypercube containing a jump discontinuity. Every cell consisting of \( 2^d \) points and containing a discontinuity is identified and flagged for refinement. When such a cell is identified a tensor product quadrature rule is applied to this cell, and each sub cell consisting of \( 2^d \) points that contains an edge is identified and the process repeated.

Here we propose an alternate procedure for detecting jump discontinuities in a moderate number of dimensions. Specifically we use a low-order sparse grid to provide an initial grid to be used for edge detection. Polynomial annihilation is
then used to refine this grid using an edge tracking procedure.

5.3.1 Edge Tracking Algorithm

Sparse grids provide an efficient means of sampling a high-dimensional input space. Here we use sparse grids to provide a set of points which provide a reasonable coverage of the input space, and which can be used for initially identifying any discontinuities. The intervals of the sparse grid can then be refined in a line-by-line fashion, using polynomial annihilation, until an edge is resolved up to a pre-specified tolerance $\delta$. This typically occurs first only at one or a small number of locations within the domain. At these locations a small tensor product grid is generated and assessed to determine the local extension of the discontinuity. Every time an interval is created with a length smaller than $\delta$ its local neighbourhood is searched for neighbouring edge intervals. Such an edge tracking procedure removes the need for some points further away from the discontinuity which are necessary when refining in a top-down fashion, as with the cell division method. As the local procedure evolves, the larger grid is continually adapted to identify additional edge intervals for local refinement.

We remark that, when constructing the local stencils $S_{x,i}$ and $S_{x,j}$ used to build $L$, special care must be taken in regions adjacent to the boundaries of the computational domain. In such situations any point in $S_{x,i}$ or $S_{x,j}$ outside the domain is discarded. It is possible that during refinement some points may have already been added to the sparse grid, so adequate book keeping is required to keep track of the points added.

Example 5.3.1. Consider the following function

$$
    f(x) = \begin{cases} 
        1, & \sum_{i=1}^{d} x_i^2 < r^2 \text{ and } x_1 < 0 \\
        x_1, & \text{otherwise.} 
    \end{cases} 
$$

(5.11)

Figure 5.1 displays the evolution of the Edge Tracking Algorithm as it is applied to Example (5.3.1) for $d = 2$, $r = 0.5$, $\delta = 2^{-4}$, $m = 2$, and $l_{\text{init}} = 3$. Figure 5.1(a) depicts the initial grid and Figure 5.1(b) displays the adaptive grid after three iterations. Here edge points are displayed in green, and new midpoints to perform boundary detection are displayed in red. At this iteration, the first set of edge points are identified and a local stencil is added in the neighborhood of these edge points. Figure 5.1(c) displays the adaptive grid after eight iterations. Notice that the adaptive grid tracks the location of the edge and only the midpoints at the front of the edge tracking are evaluated for edges. Figure 5.1(d) depicts
Algorithm 4 Given an initial grid, a desired resolution of $\delta$ and a desired boundary order of accuracy $m$, the following steps are iterated until no new grid points are determined to be added:

1. Generate an initial isotropic sparse grid of level $l_{\text{init}}$ and add all the points to $\mathcal{S}$

2. Define $\tilde{\mathcal{S}}$ as the set of any two points in $\mathcal{S}$ that are nearest neighbours in an axial direction.

3. Find the set of new grid points to be added

$$\mathcal{A} = \left\{ \mathbf{x} = (y_1, \frac{y_1 + z_1}{2}, \ldots, y_d) : (y, z) \in \tilde{\mathcal{S}}, \text{ } MM(L_m f(x_i)) > O(|y_i - z_i|^m), \text{ and } |y_i - z_i| \geq 2\delta \right\}$$

where $i$ is the axial direction along which the discontinuity was found

4. While $\mathcal{A} \neq \emptyset$

   - Add the new points in $\mathcal{A}$ to the grid $\mathcal{S}$
   - Update $\tilde{\mathcal{S}}$
   - Find the set of new grid points to be added $\mathcal{A}$
   - Find the set of edge points given by

$$\mathcal{E} = \left\{ (y, z) : (y, z) \in \mathcal{S}, x_i = \frac{y_1 + z_1}{2}, \text{ } MM(L_m f(x_i)) > O(|y_i - z_i|^m), \text{ and } |y_i - z_i| \leq \delta \right\}$$

   - For each edge point $\mathbf{z}$ in $\mathcal{E}$, refine the grid locally

     - Define $\mathcal{S}_{x,i}$ to be the local one-dimensional four point stencil containing the edge $\mathbf{z}$ ($z_i - \delta, z_i - \delta/2, z_i, z_i + \delta/2$)
     - Define $\mathcal{S}_{x,j}$, $j = 1, \ldots, d$, $i \neq j$ to be the one-dimensional three point stencil ($z_j - \delta, z_j, z_j + \delta$)
     - Construct the tensor product grid $\mathcal{L} = \mathcal{S}_{x,1} \otimes \cdots \otimes \mathcal{S}_{x,i} \otimes \cdots \otimes \mathcal{S}_{x,d}$ and add the $4 \cdot 3^{d-1}$ points to the grid $\mathcal{S}$
     - Identify all intervals of $\mathcal{L}$ containing a discontinuity and add the associated edge point to $\mathcal{E}$. 

CHAPTER 5. DISCONTINUITY DETECTION

Figure 5.1: (a) Initial grid with the limit of resolution set to $\delta = 2^{-4}$; (b) adaptive grid after three iterations - here edge points are displayed in green, and new midpoints to perform boundary detection are displayed in red; (c) adaptive grid after eight iterations; (d) final grid. To obtain the same resolution using an isotropic sparse grid requires 7169 function evaluations.

The final sparse grid, where it can be seen that the edge in Example (5.3.1) is captured with a uniform grid with a width of order $m + 1$. Some 137 points are needed to capture the discontinuity to the desired resolution. To obtain the same resolution using an isotropic sparse grid requires 7169 function evaluations.

To facilitate convergence analysis we use distance to the nearest edge to assess the accuracy of the representation of a discontinuity. Using the tensor-product edge tracking procedure, such a distance function can be approximated by

$$D_E(x) = \min \left\{ \left\| x - \left( y_1, \ldots, \frac{y_i + z_i}{2}, \ldots, y_d \right) \right\|_2 : (y, z) \in E \right\}$$

(5.12)
Example 5.3.2. Consider the following function:

\[ f(x) = \begin{cases} 
1, & \sum_{i=1}^{k} x_i^2 < r^2, \ k \leq d \\
0, & \text{otherwise}
\end{cases} \quad (5.13) \]

which has a simple analytic expression of the distance function to the nearest edge

\[ D_f(x) = \left| r - \left( \sum_{i=1}^{k} x_i^2 \right)^{\frac{1}{2}} \right|. \quad (5.14) \]

Table 5.1 reports the number of points in the grid generated by the edge tracking algorithm for Example (5.3.2) and the resulting error in the estimation of the distance function to the nearest edge given in Equation (5.17) as dimensionality is increased. Here we choose the initial grid to be a first level sparse grid, \( m = 2, \delta = 2^{-4} \) and \( k = d \). It can be seen that the error in the estimation of the distance function to the nearest edge is maintained at \( \mathcal{O}(\delta) \), and the number of grid points is proportional to the dimension and hyper-dimensional surface area of the functional edge.

Table 5.1: The number of points \( N \) and the error in the approximated distance function of Example (5.3.2) with \( \delta = 1/32 \) and \( r = 1/8 \), for increasing dimension \( d \).

<table>
<thead>
<tr>
<th>( d )</th>
<th>( N )</th>
<th>( \epsilon_{max} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>57</td>
<td>0.03171</td>
</tr>
<tr>
<td>3</td>
<td>293</td>
<td>0.03070</td>
</tr>
<tr>
<td>4</td>
<td>1585</td>
<td>0.02848</td>
</tr>
<tr>
<td>5</td>
<td>8253</td>
<td>0.02516</td>
</tr>
<tr>
<td>6</td>
<td>40617</td>
<td>0.03163</td>
</tr>
<tr>
<td>7</td>
<td>189717</td>
<td>0.03191</td>
</tr>
</tbody>
</table>

Figure 5.2 further illustrates the dependency of the number of points required to resolve a discontinuity on the dimension, hyper-dimensional surface area and the resolution \( \delta \). Figures 5.2 (a) shows that the number of points required to resolve a discontinuity, up to a certain resolution, is dependent on the hyper-dimensional surface area which grows with dimensionality. Figure 5.2 (b) highlights that for fixed surface area the number of points required is inversely proportional to \( \delta \). The small difference in the number of points required for \( \delta = 1/16 \).
and $\delta = 1/32$ is because the small discontinuity arising from a radius of $1/16$ is indistinguishable at these coarser resolutions.

If the discontinuity exists along all axial directions the dependence of the number of points necessary on the hyper-dimensional surface area cannot be broken. However, if a discontinuity only exits in a small subset of dimensions, this dependence can be broken and the curse of dimensionality inherent in the tensor product formulation can be delayed significantly. Consider Example (5.3.2) with $k = 2$ and $d = 3$ the discontinuity manifold is a cylinder. Figure 5.3 shows the results of the edge tracking procedure applied under these conditions. For any value of $x_2$ (the perpendicular axis) a discontinuity can be found along the remaining axial directions. However after searching along lines parallel to the perpendicular axis no discontinuity is present. In such situations the number of points needed to capture the discontinuity manifold could be drastically reduced by preventing refinement in the perpendicular axis. This proposition is the basis of the discontinuity detection method presented in the following section.

5.4 High-Dimensional Discontinuity Detection

In a $d$-dimensional space any discontinuity will lie on a $d - 1$ manifold. The number of points needed to characterise such a manifold is proportional to its volume. Thus the number of points needed to represent a discontinuity will
increase rapidly with dimensionality. The main aim of the discontinuity detection methods proposed here is to find the lower-dimensional structure of the manifold embedded in the higher-dimensional space. Specifically the proposed algorithms attempt to work in the low-dimensional latent space of the manifold to provide an efficient characterisation of the discontinuity manifold.

Dimension-wise decompositions of the input-output relationship have arisen as a successful means of delaying or even breaking the curse of dimensionality for certain applications. Such an approach represents the model outputs as a high-dimensional function \( f(x) \) dependent on the model inputs \( x = (x_1, \ldots, x_d) \). It uses an ANOVA type decomposition

\[
f(x_1, \ldots, x_d) = f_0 + \sum_{n=1}^{d} f_i(x_i) + \sum_{i,j=1}^{d} f_{i,j}(x_i, x_j) + \cdots + f_{i,\ldots,d}(x_i, \ldots, x_d) \quad (5.15)
\]

which separates the function into a sum of subproblems. The first term is the zero-th order effect which is a constant throughout the \( d \)-dimensional variable space. The \( f_i(x_i) \) terms are the first-order terms which represent the effect of each variable acting independently of all others. The \( f_{i,j}(x_i, x_j) \) terms are the second-order terms which are the contributions of \( x_i \) and \( x_j \) acting together, and so on.

In practice only a small number of interactions contribute significantly to the system response and consequently only a low-order approximation is needed to represent the input-output mapping accurately [122]. Moreover only a small subset of the remaining terms may actually control the location of discontinuities and/or kinks.
In this section we propose a locally-adaptive approach to determine and resolve the location of any discontinuities. The method only refines in dimensions deemed to contribute to the location of the discontinuity, thereby expending more effort in regions which likely contain a discontinuity. The polynomial annihilation method is used to guide refinement that invests the majority of function evaluations in regions surrounding discontinuities and neglects the grid points in the smooth regions.

5.4.1 Adaptive Sparse Grids

The discontinuity detection method proposed here is closely related to adaptive sparse grid approximation. In comparison to isotropic sparse grids, presented in Section 4.1, which treat all dimensions and regions of the input space equally, adaptive sparse grids spend more effort sampling a function in areas and dimensions of interest. Here we extend the adaptive sparse grid framework used for interpolation and integration [46, 76] to discontinuity detection.

The hierarchical formulation (4.6) lends itself well to local refinement. Setting $\epsilon > 0$ adaptively selects which nodes are added to the sparse grid. Subsequently the points in the sparse grid become concentrated in discontinuous or rapidly varying regions. The number of points in the spare grid depends on the value of $\epsilon$ and the level of interpolation $l$.

When adaptive sparse grid approximation is applied to functions in regions of high regularity the hierarchical surpluses $w$ decay rapidly as the level of interpolation $l$ increases. However in regions containing discontinuities the hierarchical surpluses decay much slower. Consequently the amplitude of the hierarchical surpluses provides an estimate of the local error and can be used as means for determining the need for refinement. Such a procedure is able to resolve discontinuities but it cannot directly identify their location.

The sparse grid approximation procedure is presented in Algorithm 5. The goal is to find the set of points such that the corresponding interpolation error is as small as possible for a given amount of work. We start with the root point located at the middle of the hypercube $[-1, 1]^d$. We then determine the $2d$ neighbours and evaluate the function at these points. If the hierarchical surplus at any of these points is above the threshold, i.e. $w_k^l \geq \epsilon$, the point is added to the active point set $A$. Otherwise it is added to the sparse grid $S$. Once all points have been assessed, the level of interpolation is increased from $l = 1$ to $l + 1$, the points in $O$ are added to the sparse grid $S$, the active point set is added to the old index set.
5.4. HIGH-DIMENSIONAL DISCONTINUITY DETECTION

\( O \), and \( A \) is cleared. The 2d children of each of the points in \( O \) are determined and checked for refinement and the appropriate points added to \( A \). This process is repeated until the maximum level of interpolation \( l_{\text{max}} \) is reached or no points with \( u_k^l \geq \epsilon \) remain.

**Algorithm 5** Adaptive sparse grid refinement algorithm

\[
\begin{align*}
& l := 1 \\
& x := (0, \ldots, 0) \\
& S := \emptyset \\
& O := \emptyset \\
& A := \{x\} \\
\text{while} \ ( A \neq \emptyset \text{ and } l \leq l_{\text{max}} ) \text{ do} \\
& \quad S := O \\
& \quad O := A \\
& \quad \text{for} \ ( x \in O ) \text{ do} \\
& \quad \quad \text{for} \ ( \text{dir} \in \{1, \ldots, d\} ) \text{ do} \\
& \quad \quad \quad C = \text{FindAxialChildren}(x, \text{dir}) \\
& \quad \quad \quad \text{for} \ ( y \in C ) \text{ do} \\
& \quad \quad \quad \quad \text{Refine}(y, \text{dir}) \\
& \quad \quad \end{align*}
\]

5.4.2 Sparse Grid Discontinuity Detection

When endeavouring to locate discontinuities, we are no longer concerned with representing a function with some level of accuracy but rather determining the location of possible discontinuities in \( u \). With this aim we propose using the polynomial annihilation method to determine if a point is close to a discontinuity instead of using the hierarchical surplus \( u_k^l \) or similar measures.

One major advantage of employing the polynomial annihilation method is that it does not require a function evaluation to determine if a point is close to a discontinuity. A function evaluation is only performed once the choice to refine has been made. This is in contrast to many other refinement strategies, locally
Algorithm 6 Refine(y,dir)

if ( y ∉ S ∪ A ) then
  s,f,i_{cur} = FindLocalAxialStencil(y,dir,m)
  if NearEdge(s,f,i_{cur},m,ε) then
    if ( Support(y,dir) ≤ 2δ ) then
      \[ E := E \cup \{ y \} \]
    else
      \[ A := A \cup \{ y \} \]
    end if
  end if
end if

adaptive sparse grid interpolation for example, which must evaluate the function
at a point to determine whether refinement is necessary.

The fundamental heuristic of the proposed procedure is to use the basic adap­
tive sparse grid procedure (Algorithm 5), in conjunction with the polynomial an­
nihilation method, to determine whether the refinement of a point in the sparse
grid will lead to a more accurate characterization of any discontinuities. This

A fundamental property of sparse grids is that for each point in the sparse
grid, all points in the hierarchical tree of ancestors must exist. To ensure this
condition is met, whenever a point is added to the sparse grid all its missing
ancestors must be added. This can easily be done recursively.

Note an m-th order univariate PA method requires a local stencil with at
least \( m + 1 \) points with known function values. Special treatment is needed when
refining any point \( x \) in the axial direction \( i = 1, \ldots, d \), if the \( i \)-th coordinate of
the point \( x_i \) is 0. The children of these points will only have one neighbouring
point (its parent) in one or more of the axial directions. When refining the root
node \((0, \ldots, 0)\) we simply add every child. However if the children, grandchildren,
etc. were refined in a similar manner, the dimensionality of the sparse grid would
grow indefinitely and the number of points evaluated would grow exponentially.
To avoid this problem we do not refine any boundary point \((\{ x \mid \exists i \text{ such that } x_i =
-1 \text{ or } x_i = 1 \})\) in the \( j \)-th direction if \( x_j = 0 \).

It is possible that during refinement some points may have been already been
added to the sparse grid so adequate book keeping is required to keep track of
the points added. Furthermore the points in the local stencil may not be equally
Algorithm 7 The following outlines the procedure used to combine polynomial annihilation with sparse grid adaptivity to capture discontinuities up to a predefined resolution $\delta$. We call this method the Sparse Grid Discontinuity Detection (SGDD) method.

1. Initialise the active coordinate index set $u = \emptyset$

2. Construct the first level of the sparse grid
   - Add the root point $(0, \ldots, 0)$ to the sparse grid $S$
   - Evaluate and add the $2d$ children of the root point to the active point set $A$
   - Set the level of interpolation to one, i.e. $l = 1$

3. While $l \leq l_{\text{max}}$ and $A \neq \emptyset$
   - Copy the points in $A$ to $O$ and clear $A$
   - For each point in $O$
     - Generate its $2d$ children
     - Use polynomial annihilation to determine which children are near a discontinuity
     - If a point is near an edge along an axial direction $i$, set $u = u \cup \{i\}$
     - If a point is near an edge and its support is larger than $\delta$, evaluate and add the point to $A$
     - If a point is near an edge and its support is not larger than $\delta$, add the point to $E$
   - Find, evaluate and add any missing ancestors of points in $A$ to $A$
   - Copy the points in $O$ to the sparse grid $S$ and clear $O$
   - Increment the level of interpolation i.e. $l = l + 1$
CHAPTER 5. DISCONTINUITY DETECTION

spaced. So the data structures used to store the points must allow efficient searching and inserting of points. With this aim we use hash-tables with unique keys based upon the coordinates of each point.

Once an interval is created whose length is smaller than the pre-specified threshold $\delta$, the midpoint of the interval is added to the set of edge points $E$. This means that no interval will be created that has a length smaller than $\delta$.

This algorithm starts with a first-level sparse grid with $2d + 1$ points. The algorithm will identify and locate discontinuities provided this initial grid possesses at least one interval with a discontinuity. If no discontinuity is found the level of the initial sparse grid can be increased to provide a more comprehensive search for an edge.

As the algorithm evolves it identifies all the dimensions along which a discontinuity can be found. We refer to these as active dimensions, the coordinate indices of which are stored in $u$.

An example of the adaptive-edge detection algorithm in 1D is shown in Figure 5.4 and an example in 2D is shown in Figure 5.5. The adaptive nature of the proposed algorithm automatically selects and refines the dimensions that cause a discontinuity. Boundary points are added in the other dimensions to determine if an edge is present in those directions but the overwhelming number of function evaluations is spent in the aforementioned dimensions.

Figure 5.6 shows the grid points used by Algorithm 7 to characterise the discontinuity of the function given in Example (5.3.2) with $k = 2$ and $d = 3$. The sparse grid detection scheme only resolves the discontinuity along 2D slices at $x_2 = (-1, 0, 1)$. Consequently the number of points used, 747, is much less than the 7007 points used by the edge tracking scheme 4. See also Figure 5.3.

5.5 High-Dimensional Edge Tracking

To further reduce the number of points required to identify the location of a discontinuity a heuristic can be used to identify and truncate the unimportant dimensions. Here we suggest that, if a minimum level of refinement has been reached in a certain axial direction and no edge has yet been found, it is likely no discontinuity exists in that direction. As each such direction is identified they should be flagged so that no more points are invested in these un-important dimensions.

In this vein we propose a high-dimensional edge tracking scheme that combines
5.5. **HIGH-DIMENSIONAL EDGE TRACKING**

Figure 5.4: Five steps of the edge detection algorithm applied to a 1D domain with a discontinuity at $x = 1/3$ and $\delta = 0.25$. Squares represent points in the current sparse grid $S$, black circles represent points in the old points set, $O$ crosses represent the children of the points in $O$, and the square surrounding the cross indicates the mid point of the interval containing the discontinuity which is added to $A$. Empty circles represent the points in the edge point set $E$.

The strength of the adaptive sparse grid discontinuity detection method 7, to detect and refine active dimensions, and the efficiency of the tensor-product edge tracking scheme 4.

The edge tracking scheme 4 proposed in Section 5.3 uses local tensor product grids around known edge points to determine neighbouring edge points. This method is very efficient when the dimensionality of the computational domain is small. However the tensor-product construction suffers the curse of dimensionality, which limits the practical application to a moderate number of dimensions. The algorithm uses a low-order sparse grid to determine an initial set of intervals containing discontinuities, which are then refined up to a pre-defined size. Once an edge point is identified the tracking scheme is invoked to propagate knowledge of the discontinuity location out from this point.

We can extend the utility of the edge tracking scheme by only tracking a discontinuity in active dimensions, i.e. dimensions along which a discontinuity has been discovered. The initial sparse grid can be used to identify these dimensions. However the complete identification of an edge relies on the intervals between the points containing at least one edge in each active dimension. This is possible provided the initial level of the sparse grid is high. As the dimensionality is increased, however, generating the number of points in the sparse grid quickly...
Figure 5.5: Four steps of the edge detection algorithm. The solid line represents the discontinuity, the squares represent points in the current sparse grid $S$, black circles represent points in the old points set $O$, crosses represent the children of the of points in $O$ and a circle surrounding a cross indicates the mid point of an interval containing the discontinuity which is to be added to $A$. 
becomes infeasible. We propose using the sparse grid detection algorithm 7 to provide the initial grid and then use the edge tracking algorithm 4 to resolve the discontinuity completely.

Algorithm 8 proceeds in the manner specified by Algorithm 7, except that now every time a new edge point is found the edge tracking algorithm 4 is used to track the edge in the adjacent regions.

The set $\mathbf{u}$ can be used to reduce the number of points invested in the inactive dimensions. Instead of generating the $2^d$ children of every active point, we only generate $2|\mathbf{u}|$ children by refining only in the active dimensions whose coordinate indices are in $\mathbf{u}$. The cardinality of $\mathbf{u}$ is referred to as the truncation dimension in the sparse grid approximation literature [47]. Accordingly we will refer to the process of restricting refinement to active dimensions as truncation.

Initially the set $\mathbf{u}$ is empty so we cannot apply truncation immediately. We must wait until all the important dimensions have been identified. When this occurs $\mathbf{u}$ is said to be complete. We can use the active dimension coordinate index set when $\mathbf{u}$ is complete. Here we deem $\mathbf{u}$ complete when every dimension, whose coordinate index is in $\mathbf{u}$, has at least one associated interval with a length $\delta$.

To further reduce the number of points needed to characterise a discontinuity we must prevent refinement on unimportant boundaries. A boundary is unimportant if the $i$-th coordinate $i = 1, \ldots, d$ is equal to -1 or 1 and $i \notin \mathbf{u}$. Figure 5.6 illustrates how the sparse grid detection algorithm 7 refines boundaries of unimportant dimensions, the top and bottom faces of the cube.

Once truncation is invoked the missing ancestors of an active point are no
Algorithm 8 The following outlines the procedure used to combine sparse grid edge detection with edge tracking to capture discontinuities up to a pre-defined resolution $\delta$. We coin this method the High-dimensional Edge Tracking (HET) method.

1. Initialise the active coordinate index set $u = \emptyset$

2. Construct the first level of the sparse grid
   - Add the root point $(0, \ldots, 0)$ to the sparse grid $S$
   - Evaluate and add the $2^d$ children of the root point to the active point set $A$
   - Set the level of interpolation to one, i.e. $l = 1$

3. While $l \leq l_{\text{max}}$ and $A \neq \emptyset$
   - Copy the points in $A$ to $O$ and clear $A$
   - For each point in $O$
     - Generate its children
       * If $u$ is complete find children in each direction $i \in u$
       * If $u$ is not complete find children in each direction $i \in \{1, \ldots, d\}$
     - Use polynomial annihilation to determine which children are near a discontinuity
     - If a point is near an edge along an axial direction $i$, set $u = u \cup \{i\}$
     - If a point is near an edge and its support is larger than $\delta$, evaluate and add the point to $A$
     - If a point is near an edge and its support is not larger than $\delta$
       * Add the point to $E$
       * Define $S_{x,i}$ to be the local one-dimensional four point stencil containing the edge $z_i - \frac{\delta}{2}, z_i - \delta, z_i + \frac{\delta}{2}$
       * Define $S_{x,j}, j = 1, \ldots, d$, $i \neq j$ to be the one-dimensional three point stencil $(z_j - \frac{\delta}{2}, z_j, z_j + \frac{\delta}{2})$
       * Construct the tensor product grid $L = S_{x,1} \otimes \cdots \otimes S_{x,i} \otimes \cdots \otimes S_{x,d}$ and add the $4 \cdot 3^{d-1}$ points to the grid $S$
       * Identify all intervals of $L$ containing a discontinuity and add the associated edge point to $E$.
     - If $u$ is not complete find, evaluate and add any missing ancestors of points in $A$ to $A$
   - Copy the points in $O$ to the sparse grid $S$ and clear $O$
   - Increment the level of interpolation, i.e. $l = l + 1$
longer added. These points are necessary when refining in a top-down fashion like that of Algorithm 7 but are not necessary when edge tracking is used concurrently. However it must be noted that because missing ancestors are no longer added, situations may arise when the local stencil used by the polynomial annihilation method contains less than the required three points. In such cases the missing points must be added. Denote the $d$-dimensional coordinate of the child being assessed by $x$ and let $i$ be the coordinate index of the dimension along which the polynomial annihilation is begin applied. Add any of the following points which are not in the grid $(x_1, \ldots, x_{i-1}, -1, x_{i+1}, \ldots, x_d)$, $(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_d)$ and $(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_d)$ The algorithm is complete when no active points and no new edge points can be found.

Figure 5.7 shows four steps of the high-dimensional edge tracking algorithm applied to Example (5.3.2) with $k = d = 2$, $r = 0.5$ and $\delta = 1/16$. Figure 5.8 shows the final grid when HET is applied to Example (5.3.2) with $k = 2$, $d = 3$, $r = 0.5$ and $\delta = 1/16$. The number of points in this grid, 379, is smaller than the number of points in the grid resulting from sparse grid detection. We also see that points are no longer wasted on the top and bottom faces of the cube, that is, on the unimportant boundaries.

## 5.6 Post-Processing

### 5.6.1 Estimating Distance Functions

The distance function

$$\text{dist}_\varepsilon(x) = \inf_{x_\varepsilon \in \mathcal{E}} \|x - x_\varepsilon\|_2$$

measures the shortest distance from a point $x$ to the surface of the discontinuity. Using the set of edge points determined by the proposed algorithm the distance function to the nearest edge can be approximated by

$$\text{dist}_\varepsilon(x) = \inf_{x_\varepsilon \in \mathcal{E}} \left( \sum_{i \in \mathbf{u}} (x_i - x_{\varepsilon,i})^2 \right)^{\frac{1}{2}}$$

Here $\mathbf{u} \subseteq \{1, \ldots, d\}$ denotes the set of coordinate indices of the dimensions along which a discontinuity exists. In the following we will consider the convergence of the maximum error

$$\epsilon_{\max} = \max_{x \in \chi} \text{dist}_\varepsilon(x)$$
Figure 5.7: (a) adaptive grid with the limit of resolution set to $\delta = 2^{-4}$ at level 5; (b) adaptive grid at level 7; (c) adaptive grid at level 9; (d) final grid. Here edge points $\mathcal{E}$ are displayed in black, edge points to be used for tracking are shown in green, active sparse grid points $\mathcal{A}$ are displayed in red, and points in the sparse grid $\mathcal{S}$ are displayed in blue.
which is closely related to the Hausdorff distance of two curves. Here the test set $X$ is a set of 1000 points uniformly sampled from the hypercube $[-1, 1]^d$.

A slight improvement in the accuracy of (5.17) can be obtained by adding certain edge points in the $\delta$ neighbourhood of previously identified edge points. Specifically, if an edge point $x$ identifies a discontinuity in the $i$-th direction, search each axial dimension $j \neq i$ for neighbouring edge points. If a neighbouring edge point $y$ is a distance of $2\delta$ away, i.e. $y = (x_1, \ldots, x_{j-1}, x_j - 1, x_{j+1}, \ldots, x_d)$, add the point $(x_1, \ldots, x_{j-1}, x_j \pm \delta, x_{j+1}, \ldots, x_d)$ to $E$. In Figure 5.10 these points correspond to the edge points with no associated interval points.

5.6.2 Domain Classification

The sparse grid edge discontinuity detection method can be used to classify the computational domain into its continuous sub-domains. Once the points in the grid have been classified, any number of existing classification techniques can be used to construct a classification function to make predictions for new, yet unknown, data.

Here we present a modification of the classification algorithm presented in [4]. The objective of this algorithm is to classify each grid point into an appropriate class. We assume that the problem is well resolved so that the jump discontinuity lies in the interior of the grid and each region is well connected. The points are classified iteratively. The algorithm starts by assigning the root point $(0, \ldots, 0)$ a class value of 1. We then proceed to classify its neighbouring points in each active axial direction $i \in \mathbf{u}$ to the left and to the right. Recall a dimension is
deemed active if an discontinuity was found in that direction during the evolution of the discontinuity detection algorithm. If an edge point exists between a point and its neighbour, the neighbour remains unclassified and the search in that direction is terminated. As each point is classified its neighbours are determined and then classified. This procedure is repeated until every point in Class 1 has been classified. This occurs when the algorithm stops generating new points for classification. The algorithm then proceeds to search for points in Class 2. A point of the yet unclassified points, which has non-zero coordinates only in the active dimensions, is marked as belonging to Class 2. The same search procedure used to classify points in Class 1 is then used to find and classify neighbours of points in Class 2. After all points in Class 2 have been classified we check to see if any unclassified points remain. If so, we choose and assign a point to Class 3 and continue otherwise until the classification is complete. The pseudo-code for this procedure is outlined in Algorithm 9.

Algorithm 9 Classify()

\[
\begin{align*}
&k=1; \\
&U := S \\
&\text{for } (x \in U) \text{ do} \\
&\quad \mathcal{N} := \{x\}; \\
&\quad U := U \setminus \{x\}; \\
&\quad \text{while } (\mathcal{N} \neq \emptyset) \text{ do} \\
&\quad\quad \text{Class}(x) = k; \\
&\quad\quad \text{for } (x \in \mathcal{N}) \text{ do} \\
&\quad\quad\quad \text{for } (\text{dir} \in \{1, \ldots, d\}) \text{ do} \\
&\quad\quad\quad\quad z := \text{FindLeftEdgePt}(x, \text{dir}); \\
&\quad\quad\quad\quad L := \text{FindLeftNeighboursUpToEdge}(x, z, \text{dir}); \\
&\quad\quad\quad\quad z := \text{FindRightEdgePt}(x, \text{dir}); \\
&\quad\quad\quad\quad R := \text{FindRightNeighboursUpToEdge}(x, z, \text{dir}); \\
&\quad\quad\quad\quad \mathcal{N} := \mathcal{N} \cup (L \cup R) \\
&\quad\quad\quad \text{end for} \\
&\quad\quad \text{end for} \\
&\quad \text{end while} \\
&k = k + 1; \\
\end{align*}
\]

Note that not all points in the grid will be classified. Only the grid points in the hyperplane of the active dimensions are considered. The unclassified points
5.6. POST-PROCESSING

can be discarded. To classify a new data point only the coordinates of the active dimensions will determine how the point is classified.

Without loss of generality consider the discontinuity, shown in Figure 5.9, that divides the two-dimensional domain into two mutually exclusive regions. The same procedure can be extended to higher dimensions and multiple function classes. We pick a first point from the list of all unclassified points, say the point \( p_0 \), and assign it a class value \( 1 \). We consider all the points to the right of \( p_0 \), i.e., the points along the straight line \( p_0 - t_0 \), consisting of the group of points \( \{ q_0, r_0, s_0, t_0 \} \). There are no discontinuities along this line. Hence all of the points are classified as Class 1 points, and each point is assigned a class value of 1. Next we consider the group of points along the straight line \( p_0 - p_4 \), i.e., the group \( \{ p_1, p_2, p_3, p_4 \} \). Once again every point in this group gets classified as Class 1. Nine of a total of twenty five points have been classified thus far. The algorithm proceeds by acting on each one of these new eight points in a similar manner as for the point \( p_0 \). For instance, considering the vertical direction of the point \( q_0 \) would determine that the point \( q_1 \) belongs to the first class. The point \( q_2 \) does not belong to Class 1 because it is separated from \( q_1 \) by an edge point (represented by an empty circle). Considering the horizontal direction of the point \( q_0 \) would determine that the points \( \{ p_0, r_0, s_0, t_0 \} \) belong to Class 1. Naturally there will be points that are classified more than once, but they will be classified into the same class. Therefore this algorithm need only be concerned with the first classification.

Once all the grid points have been classified, any number of classification algorithms can be used to construct a classification function to make predictions for new, yet unknown, data. Support vector machines [22] and sparse grid classification [36, 53] are two such techniques. Principal manifold learning [30] could also be used to construct an approximation of the lower-dimensional manifold. In this paper we use a simple heuristic to classify points. The class of a new point is simply the class of its nearest neighbour in the grid. This will incorrectly classify points in the \( \delta \) neighbourhood of the discontinuity but this is also a failing of the aforementioned methods. We remark again that the inactive dimensions have no influence on the classification of new data points. The discontinuity detection Algorithm 7 is a dimension reduction technique that allows important, active dimensions to be identified and the unimportant dimensions to be ignored. Subsequently any post-processing procedure, be it classification or estimating distances to an edge, can be performed on the lower-dimensional latent space.
5.7 Numerical Results

In this section we provide several numerical tests to illustrate the implementation and convergence of the proposed methodology. While several examples are presented in earlier sections, here we focus on the performance of the algorithms in high-dimensional spaces.

5.7.1 Benchmark Problems

We first consider the following examples

Example 5.7.1. Square

\[ f(x) = \begin{cases} 
1, & |x_1| < \frac{1}{2} \text{ and } |x_2| < \frac{1}{2} \\
0, & \text{otherwise.}
\end{cases} \]  \hfill (5.19)

Example 5.7.2. Plane

\[ f(x) = \begin{cases} 
1, & x_1 + x_2 > \frac{3}{4} \\
0, & \text{otherwise.}
\end{cases} \]  \hfill (5.20)

Example 5.7.3. Triangle

\[ f(x) = \begin{cases} 
0, & x_1 + x_2 > \frac{1}{4} \text{ and } x_1 - x_2 > \frac{1}{4} \text{ and } x_1 < -0.5 \\
1, & \text{otherwise.}
\end{cases} \]  \hfill (5.21)
5.7. NUMERICAL RESULTS

Example 5.7.4. Sphere

\[ f(x) = \begin{cases} 
1, & \sum_{i=1}^{k} x_i^2 < r^2, \quad k \leq d \\
0, & \text{otherwise.} 
\end{cases} \] (5.22)

Figure 5.10 plots the final grid generated by Algorithm 8 and the associated approximate distance function for Examples 5.7.1–5.7.4.

First let us investigate the error of the approximated distance function using the HET method, as \( \delta \) decreases. Table 5.2 shows the number of points needed to capture the discontinuity present in Example 5.7.4 with \( k = 3 \leq d \) and \( r = 1/8 \) for varying dimensionality \( d \) of the input space. The number of points required increases only linearly with the dimensionality of the input space. The vast majority of points are invested in the active dimensions. A graphical illustration of the linear increase in the number of points with dimension is shown in Figure 5.11.

<table>
<thead>
<tr>
<th>( d )</th>
<th>( N )</th>
<th>( \epsilon_{\text{max}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>573</td>
<td>0.03206</td>
</tr>
<tr>
<td>10</td>
<td>883</td>
<td>0.03160</td>
</tr>
<tr>
<td>15</td>
<td>1193</td>
<td>0.03156</td>
</tr>
<tr>
<td>20</td>
<td>1503</td>
<td>0.03075</td>
</tr>
<tr>
<td>25</td>
<td>1813</td>
<td>0.03055</td>
</tr>
<tr>
<td>50</td>
<td>3363</td>
<td>0.03137</td>
</tr>
<tr>
<td>100</td>
<td>6463</td>
<td>0.03128</td>
</tr>
</tbody>
</table>

Table 5.2: \( \epsilon_{\text{max}} \) error in the approximated distance function of the 3-dimensional spherical discontinuity as the dimensionality of the input space is increased for Example 5.7.4 with \( k = 3 \leq d \) and \( r = 1/8 \),

Table 5.3 displays the number of points required by the HET method as \( \delta \) decreases, when applied to Example 5.7.4 with \( k = 3 \) and \( d = 20 \). The error in the distance function is maintained at \( O(\delta) \).

Figure 5.12 (a) plots the \( \epsilon_{\text{max}} \) error of the HET method when applied to Example 5.7.4 with \( k = 3 \) and \( d = 20 \). To illustrate the efficiency of the HET method we also plot the error of the SGDD method and the truncated SGDD method, TSGDD. For all methods the error in the approximated distance function decreases with \( \delta \). The HET method is by far the most efficient. TSGDD is the second-most efficient method and SGDD requires the most amount of points to characterise the discontinuity.
Figure 5.10: Estimated distance functions, determined from Algorithm 5 in conjunction with Algorithm 6, for $d = 2$ and $\delta = 2^{-4}$, as it is applied to Examples 5.7.1 (a), 5.7.2 (b), 5.7.3 (c), and 5.7.4 with $k = d = 2$ (d). Black points represent points in the sparse grid $S$, red points represent edge points $E$, and the green points the intervals on which the edge points are defined. Edge interval points are also in $S$. 
5.7. NUMERICAL RESULTS

Figure 5.11: Example 5.7.4 with \( k = 3 \leq d \) and \( r = 1/8 \) for varying dimensionality \( d \) of the input space. Here the discontinuity resides on a small subset of the dimensions and the total number of points required grows linearly with the dimensionality.

<table>
<thead>
<tr>
<th>( \delta )</th>
<th>( N )</th>
<th>( \epsilon_{\text{max}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 1/8 )</td>
<td>1027</td>
<td>0.09112</td>
</tr>
<tr>
<td>( 1/16 )</td>
<td>1503</td>
<td>0.03164</td>
</tr>
<tr>
<td>( 1/32 )</td>
<td>2513</td>
<td>0.01577</td>
</tr>
<tr>
<td>( 1/64 )</td>
<td>17747</td>
<td>0.007779</td>
</tr>
<tr>
<td>( 1/128 )</td>
<td>37633</td>
<td>0.003872</td>
</tr>
</tbody>
</table>

Table 5.3: \( \epsilon_{\text{max}} \) error in the approximated distance function as \( \delta \) is decreased for the 3-dimensional spherical discontinuity embedded in a larger input space, Example 5.7.4 with \( k = 3 \leq d \) and \( r = 1/8 \).

Now we analyse the ability of the three methods to classify a set of randomly generated data points. Here we randomly generate 1000 points that lie a maximum distance of 0.125 away from the true discontinuity and classify each point according to the classification algorithm presented in Section 5.6.2. Figure 5.12 (b) plots the percentage of new points that are classified incorrectly as the size of \( \delta \) decreases. Again HET is the most efficient method, followed by the TSGD and SGD methods. Note that if 1000 sample points were simply generated uniformly in \([-1, 1]^{20}\), all three methods classify all the points correctly.
5.7.2 Multidimensional Stochastic Application

We now consider a stochastic differential-algebraic system of equations

\[ \begin{align*}
    \frac{du}{dt} &= \frac{\alpha_1}{1 + v^\delta} - u, \\
    \frac{dv}{dt} &= \frac{\alpha_2}{1 + w^\gamma} - v, \\
    w &= \frac{u}{(1 + |IPTG|/K)^\eta},
\end{align*} \tag{5.23} \]

where \( \alpha_1, \alpha_2, \beta, \gamma, \eta \) and \( K \) are parameters and \(|IPTG|\) is the system input. This model is for a genetic toggle switch in *Escherichia coli* [37]. Stochastic simulation of this system was conducted in [127], where the parameters \( p = (p_1, \ldots, p_6) = (\alpha_1, \alpha_2, \beta, \gamma, \eta, K) \) are modelled as random variables in the form of \( p = \langle p \rangle (1 + \sigma y) \), where \( \langle p \rangle = (156.25, 15.6, 2.5, 1, 2.0015, 2.9618 \times 10^{-5}) \) are the mean values and \( y = (y^{(1)}, \ldots, y^{(6)}) \) are random variables uniformly distributed in \([-1, 1]^6\).

The system will reach steady state, which exhibits a switch property, i.e. “on” and “off”, depending on the intensity of the input \(|IPTG|\). The switch behavior suggests that the solution, \( v \), depends on the parameters in a non-smooth manner. However, it is not clear either from the physics or the mathematical point of view which parameters are associated with the jump behavior.

Here we provide a comparison of the adaptive-domain classification method proposed by Archibald et al. [4] with the HET method. Archibald et al. carry out the multi-dimensional edge detection method for \( v(y) \) in four dimensions.
neglecting the influence of $\beta$ and $\gamma$ which are fixed at their nominal values. Using the High-dimensional Edge Tracking method, 31,104 points were needed to capture the discontinuity to a resolution of $\delta = 2^{-4}$. Applying the classification procedure outlined in Section 5.6.2 to 1000 point randomly generated in $[-1, 1]^4$, 98.1% of points were classified correctly. The computational cost of the Algorithm is better than the adaptive-domain classification method [4], where the number of nodes in the HET generated grid was 31,379 as compared to 91,250 grid points for the adaptive-domain classification method.

5.8 Conclusions

In this Chapter we present a method for identifying and locating discontinuities in high-dimensional spaces. The method extends, and improves significantly, the previous algorithms using polynomial annihilation for detecting discontinuities in low dimensions.

The proposed discontinuity detection method proposed is closely related to adaptive sparse grid approximation. A locally adaptive approach is used to determine and resolve the location of any discontinuities. The method only refines in dimensions deemed to contribute to the location of those discontinuity, thereby spending more effort in regions which likely contain a discontinuity. The polynomial annihilation method is used to guide refinement that invests the majority of function evaluations in regions surrounding discontinuities and neglects the grid points in the smooth regions.

Upon detecting the discontinuities, the adaptive sparse grid approach is further combined with a tracking scheme to efficiently capture the location and geometry of the discontinuities which often reside on a subset of the dimensions. The tracking scheme uses local tensor product grids around known edge points to determine neighbouring edge points. This process is performed iteratively, thereby efficiently investing points along the discontinuity and out from the original edge points.

The grid generated by the discontinuity method can be used to classify the computational domain into its continuous sub-domains. Once the points in the grid have been classified, any number of existing classification techniques can be used to construct a classification function to make predictions for new, yet unknown, data. We developed a simple heuristic that classifies new data based upon the classification of the nearest grid point. Such a procedure is shown to
provide an accurate classification function. The discontinuity grid can also be used as a distance function that calculates the distance of a new data point from a discontinuity.

The accuracy of the classification and distance functions is proportional to the resolution $\delta$ used to resolve the discontinuity. The number of grid points is proportional to the dimension and hyper-dimensional surface area of the functional edge. Analogously, for fixed surface area the number of points required is inversely proportional to the resolution $\delta$ used to resolve the discontinuity. When the dimensionality of the discontinuity resides in a small subset of dimensions of the input space the proposed method is very efficient. In these situations the algorithm becomes "optimal," that is the total number of points required grows linearly with the dimensionality.
Chapter 6

Conclusions

This thesis presented various numerical methods to facilitate the efficient quantification of uncertainty in mathematical models. We began by discussing the various definitions of uncertainty that are used in the existing literature and the various schemes which can be used to classify the various types and sources of uncertainty (see Figures 1.1–1.3).

The main aim of this thesis is to quantify statistical uncertainty which can be quantified by probabilistic information or other numerical properties such as ranges. With this in mind we identify the stochastic collocation method as an efficient means of quantifying such uncertainty and discuss various methods for representing uncertainty in the modelling process.

The vast majority of stochastic collocation (SC) methods and Uncertainty Quantification (UQ) methods in general have focused on aleatory uncertainty. The meaningful quantification of aleatory uncertainty requires detailed probabilistic knowledge of the nature of the uncertain inputs. However often such information is limited. The absence of any lack of knowledge or information in any phase or activity of the modeling process is referred to as epistemic uncertainty.

In this thesis we proposed a framework for quantifying epistemic uncertainty. The methodology presented is a generalization of traditional aleatory uncertainty quantification that allows one to seamlessly switch between epistemic, aleatory and mixed epistemic-aleatory uncertainty analysis. The validity and effectiveness of our approaches is illustrated through several examples.

The approach is based on solution of an “encapsulation problem” which generates a solution to the governing equations in a domain that encloses the true probability space with overwhelming probability. No distributional information
about any of the variables need be assumed, only estimates of the ranges of the variables are needed. Once the bounds have been specified, a polynomial approximation can be constructed in the encapsulation domain. The polynomial approximation to the encapsulation problem is chosen to converge pointwise throughout the input space. Thus it is also accurate on a subset of this space. As long as this pointwise convergence is obtained, the polynomial solution of the encapsulation problem can be used as an effective model for the true solution on the true domain.

Once the polynomial approximation within the encapsulation domain has been computed, it can be employed within epistemic analyses such as interval analysis or evidence theory, within sensitivity analysis studies to explore the importance of epistemic parameters and allocate experimental resources, and ultimately within a posteriori aleatory analyses following collection of additional experimental data. In this thesis, we focused on the a posteriori evaluation of solution statistics and demonstrate convergence of statistics following the introduction of additional information on functional dependence or correlation, distributional form, or both.

The encapsulation approach can readily handle dependencies between input variables. If the functional dependence becomes known a posteriori, then the relationships can be used to collapse the dimensionality of the polynomial approximation, and then quadrature methods can be applied on this lower-dimensional space to obtain estimates of moments. We show that exponential convergence can be obtained for correlated normals by utilizing a Cholesky decomposition to map a set of independent variables (needed to construct the polynomial approximation) to a set of dependent variables. Although not presented, such a procedure could be extended to correlated non-normals provided a variable transformation exists, for example the Nataf transformation.

If the distributional form of the epistemic variables becomes known a posteriori, then solution statistics can be evaluated as a post-processing step. While the polynomial basis selected a priori will not in general be optimal for this a posteriori post-processing, this suboptimal weighting of polynomial approximation errors is an algorithmic cost. That cost must be paid for having imperfect characterization of uncertainties at the time of approximation construction. The modeler can minimize this penalty by tailoring the basis to the available information to the extent possible. But in the case of a pure interval-based epistemic uncertainty description, there is no justification to weight errors unequally within the interval and a Legendre basis is the logical choice. It is demonstrated that exponentially-fast convergence rates can nonetheless be obtained for the a pos-
teriori solution statistics despite the lack of complete information at expansion construction time.

Stochastic collocation methods are one way of constructing the polynomial approximation necessary to solve the encapsulation problem. A number of SC methods exist. However the use of these methods is often restricted to functions with high-degrees of smoothness and/or with a moderate number of uncertain variables. In this thesis we present an $h$-adaptive generalised sparse grid ($h$-GSG) method for interpolating high-dimensional functions with discontinuities. The proposed algorithm extends and improves upon existing approaches by combining the strengths of the generalised sparse grid algorithm and hierarchical surplus guided $h$-adaptivity.

The underlying generalised sparse grid algorithm greedily selects the dimensions and variable interactions that contribute most to the variability of a function. The hierarchical surplus of the points within each subspace is used as an error criterion for $h$-refinement with the aim of concentrating computational effort within rapidly varying or discontinuous regions. This approach limits the number of points that are invested in 'unimportant' dimensions and regions within the high-dimensional domain.

A high-degree basis is used to obtain a high-order method which, given sufficient smoothness, performs significantly better than the traditional piecewise-linear basis. When discontinuities are present in the function surface or its derivatives, performance deteriorates. However, it was shown numerically that even in such situations the quadratic basis will still result in higher-rates of convergence than achieved by using piecewise-linear interpolation.

The importance of the model (function) variables which are the subject of UQ analysis are governed by natural yet unknown weights. In these cases, the proposed method can utilise this implicit weighting to determine and restrict effort to the effective dimension of the model. In this thesis we show the utility of the $h$-GSG method for non-smooth functions with as many as 700 variables.

Although the $h$-GSG method is an improvement on existing sparse grid SC methods, it is likely that higher rates of convergence could be obtained by decomposing the input domain into regions of high-regularity on which independent global polynomial methods can be applied. Although the discretization is straightforward conceptually, the implementation of multi-element techniques is numerically challenging. The existing literature has restricted attention to high-dimensional rectangular elements. The size and position of these elements is determined adaptively by sub-dividing elements in which the local error is large.
Multi-element techniques can be improved by determining the optimal domain decomposition a priori and removing the restriction of rectangular elements. With this goal we present a method for identifying and locating discontinuities in high-dimensional spaces. The method extends, and improves significantly, the previous algorithms using polynomial annihilation for detecting discontinuities in low dimensions.

The proposed discontinuity detection method proposed is closely related to adaptive sparse grid approximation. A locally adaptive approach is used to determine and resolve the location of any discontinuities. The method only refines in those dimensions deemed to contribute to the location of the discontinuity, thereby spending more effort in regions which likely contain a discontinuity. The polynomial annihilation method is used to guide refinement that invests the majority of function evaluations in regions surrounding discontinuities and neglects the grid points in smooth regions.

Upon detecting the discontinuities, the adaptive sparse grid approach is further combined with a tracking scheme to efficiently capture the location and geometry of the discontinuities that often reside on a subset of the dimensions. The tracking scheme uses local tensor product grids around known edge points to determine neighbouring edge points. This process is performed iteratively, thereby efficiently investing points along the discontinuity and out from the original edge points.

The grid generated by the discontinuity method can be used to classify the computational domain into its continuous sub-domains. Once the points in the grid have been classified, any number of existing classification techniques can be used to construct a classification function to make predictions for new, yet unknown, data. We developed a simple heuristic that classifies new data based upon the classification of the nearest grid point. Such a procedure is shown to provide an accurate classification function. The discontinuity grid can also be used as a distance function which calculates the distance of a new data point from a discontinuity.

The accuracy of the classification and distance functions is proportional to the resolution \( \delta \) used to resolve the discontinuity. The number of grid points is proportional to the dimension and hyper-dimensional surface area of the functional edge. Analogously, for fixed surface area the number of points required is inversely proportional to the resolution \( \delta \) used to resolve the discontinuity. When the dimensionality of the discontinuity resides in a small subset of dimensions of the input space the proposed method is very efficient. In these situations the
algorithm becomes “optimal,” that is the total number of points required grows linearly with the dimensionality.

The discontinuity detection method provides a stepping stone towards multi-element methods with arbitrarily shaped sub-domains. Current research is focusing on integrating discontinuity detection with interpolation on irregularly shaped domains.

Here we note that the use of the $h$-adaptive generalised sparse grid method and the discontinuity method are not restricted to uncertainty quantification. These methods can be used for interpolation of any high-dimensional functions with finite support. It is this idea that facilitates the extension of stochastic collocation methods to quantifying epistemic uncertainty.
Bibliography


BIBLIOGRAPHY


[110] L. P. Swiler, T. L. Paez, R. L. Mayes, and M. S. Eldred. Epistemic un-
certainty in the calculation of margins. In *AIAA Structures, Structural

[111] G. Tang, Swiler. L. P., and M. S. Eldred. Using stochastic expansion meth-
ods in evidence theory for mixed aleatory-epistemic uncertainty quantifica-
Orlando, FL, April 12-15 2010.

parametric uncertainty analysis of numerical geophysical model. *Journal of

[113] R.A. Todor and C. Schwab. Convergence rates for sparse chaos approxi-
mations of elliptic problems with stochastic coefficients. *IMA Journal of

[114] R. Van der Merwe, T. Leen, Z. Lu, S. Frolov, and A. Baptista. Fast neu-
ral network surrogates for very high dimensional physics-based models in

for multidimensional PDEs. *Journal of Computational Physics*, 138:16–56,
1997.

[116] W.E. Walker, P. Harremoës, J. Rotmans, J.P. Van der Sluis, M.B.A.
Van Asselt, P. Janssen, and M. Krayer von Krauss. Defining uncertainty:
a conceptual basis for uncertainty management in model-based decision

nomial Chaos method for stochastic differential equations. *Journal of Com-

2006.

spatially-dependent random coefficients: algorithms, error analysis and


