GOAL-ORIENTED ADAPTIVE ALGORITHMS FOR STOCHASTIC COLLOCATION FINITE ELEMENT METHODS

by

THOMAS ROUND

A thesis submitted to the
University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

Supervisor: Dr. Alex Bespalov

School of Mathematics
College of Engineering and Physical Sciences
University of Birmingham
March 2024

ABSTRACT

In this thesis, our main objective is the numerical approximation of linear quantities of interest for problems involving elliptic partial differential equations (PDEs) which admit inputs with either an affine or non-affine dependence on random parameters. Our approach will involve a stochastic collocation finite element method, which is based on a deterministic finite element method (FEM), combined with a sparse grid collocation procedure. This method will be introduced in the context of adaptive algorithms, which are furnished by a posteriori error estimation techniques that are designed to independently influence both spatial and parametric adaptivity.

Firstly, we produce new results in the stochastic collocation FEM setting that reveal the impact of different strategic choices in the context of an adaptive algorithm. Specifically, we investigate choices of hierarchical spatial error estimation strategy, the use of different families of collocation points within the construction of sparse grids, and the use of the reduced margin (in comparison to the full margin) for the purposes of enriching the underlying sparse grid.

Secondly, we propose an error estimation strategy for the goal-oriented framework which utilises products of hierarchical estimators in the stochastic collocation FEM setting. This involves the use of a novel correction term to compensate for the lack of global Galerkin orthogonality in this setting. Our error estimation strategy drives a goal-oriented adaptive algorithm with innovative marking procedures to simultaneously handle the interplay between primal and dual, as well as spatial and parametric contributions to our error estimate. We provide an upper bound for the underlying error estimate, and demonstrate the performance of the resulting adaptive algorithm in extensive numerical experiments.

Thirdly, we suggest a dual-weighted residual estimate for the error in the quantity of interest, and a variant of this which utilises the primal problem as weightings for residuals associated with the dual problem. We prove the reliability of both the standard dual-weighted residual method, and the new symmetric dual-weighted residual, for the purposes of estimating the error in the quantity of interest. Further numerical experiments illustrate the performance of the corresponding adaptive algorithm.

ACKNOWLEDGEMENTS

I would like to take this opportunity to acknowledge the scholarship funding that I have received via the EPSRC for part of my research. I would also like to express my gratitude to a number of individuals whose support I have enjoyed throughout my doctoral studies.

Firstly, I would like to thank Alex Bespalov, who has supervised me throughout my research. He has continued to guide me throughout difficult times, offering his invaluable wisdom on many matters, and helping me to focus on the tasks at hand. I would also like to thank him for helping to facilitate my mini-symposium presentation of some of the preliminary results of this thesis at the 29th Biennial Numerical Analysis Conference at the University of Strathclyde.

I also wish to mention my thanks for Dirk Praetorius for his suggestions that have assisted me with parts of this work.

I would lastly like to thank my family for their continued belief in me that I, on occasion, was lacking myself.

Contents

Chapter 1

Introduction

Mathematical models involving partial differential equations (PDEs) are ubiquitous in a variety of applications. Indeed, much of our knowledge of the natural world (and other phenomena, for that matter) can be described using various laws and concepts that are used to derive models based on PDEs.

Fick's law governing the diffusion of particles according to a concentration gradient, as originally introduced in [1], is used to derive the commonly known diffusion equation. Analogous laws exist in applications which analyse electrical currents (Ohm's law), heat conduction (Fourier's law), and elsewhere. Considerations of conservation of mass and momentum in fluids lead to the Navier-Stokes equations, a derivation of which can be found in [2]. The work of De Broglie on wave-particle duality eventually allowed for the derivation of the Schrödinger equation in [3], an important innovation in the field of quantum mechanics.

However, many scenarios that we attempt to devise mathematical models for will involve random or uncertain features. Calculations involving financial derivatives, for example, will also typically involve considerations of volatility (see [4]). In fluid dynamics, we may wish to incorporate randomness into our models of flows in porous media (see [5]), because we do not have precise information about quantities such as porosity, even though they are not inherently random. Various other examples exist elsewhere, including material inconsistencies, noise terms involved in the propagation of waves, and others. In general, modelling uncertainty may be motivated in a number of cases, including underlying irregularities associated with the situation under consideration, or our own lack of a priori knowledge of the variables involved.

Where input data varies within a given spatial domain, it is often prudent to describe the uncertainty that we are attempting to model using random fields, as noted in [6]. These random fields may be described in multiple ways, including either polynomial chaos expansions (see [7] or [8]) or Karhunen-Loéve expansions (see [9]). In certain scenarios (including the modelling of groundwater flow), the random variables involved may also have significant variability, or other requirements such as being non-negative, such that we seek non-linear expansions to represent the associated random fields, as done in [10, 11], and various others. By using a suitable random field, we can formulate problems involving so-called 'parametric' PDEs, which incorporate the random fields and associated variables into our model.

For a given parametric PDE problem, we want to consider convenient numerical methods for discretising and solving the problem. Methods for solving PDEs in the deterministic setting, such as finite element methods (FEM), are well-understood (from, amongst other sources [12, 13, 14]), although convergence results associated with the corresponding adaptive algorithms have only been developed more recently. In particular, the first proofs of convergence for adaptive finite element methods are given in [15] and [16], while the optimal computational complexity of an adaptive algorithm was first demonstrated in [17]. Work on optimal convergence rates has been performed in [18, 19] and others, with optimal convergence rates for adaptive finite element methods being established for problems with more general source terms in [20].

In general, there are two principal approaches to adapting these methods to the context of parametric PDE problems. The first approach may be considered a direct analogue of the techniques we typically utilise in the deterministic setting, and involves the coupling together of the spatial and parameter-based discretisations. The most popular representative of these are the stochastic (or spectral) Galerkin type approximations which have their origins in [21], and rely on approximation emanating from a discretisation of an appropriate product of approximation spaces corresponding to the spatial and parameter domains. These methods have, in recent years, been applied to finite element analysis successfully, for example, in [22] and [23], with numerous examples since. These methods are often described as *intrusive*, owing to the fact that the implementations of these methods involve some level of modification to existing code that is utilised for the solution of the corresponding deterministic PDEs.

The alternative approach, which is often termed *non-intrusive*, attempts to take advantage of

existing deterministic code and parallelise it across various samples. The most commonly known examples of this are Monte Carlo methods, the origins of which are recounted in [24]. The basis of this method is using a series of spatial approximations derived from computations with random samples in order to calculate statistical moments related to the solution. Various aspects of Monte Carlo simulations are also discussed in, for example, [25], while numerous variants are also discussed in other works, such as those on quasi-Monte Carlo methods (including [26]), multi-level Monte Carlo methods (for example, [27]), and others.

Another non-intrusive method, stochastic collocation, is the main focus of our work. This approach generates approximations with functional dependence on random parameters using deterministic, rather than random, samples. Specifically, these approximations are represented as a finite expansion in terms of parameter-dependent polynomials, with spatial coefficients obtained from sampled approximations. Collocation-based approaches had already been successfully applied to parametric PDE problems in [28] and [29] by the time it was fully proposed as a method with a priori convergence analysis results in [30]. The ability to obtain a series of uncoupled, deterministic problems that can be solved in parallel, while remaining as accurate as stochastic Galerkin approaches was immediately identified as a key advantage of this method.

One immediate issue with this method was its use of full tensor product spaces, which would naturally lead to this approach becoming computationally expensive for large numbers of random variables. This issue was first addressed in [31], where a sparse grid structure (see [32]) was proposed in order to limit the number of computations required, making it competitive with Monte Carlo methods in terms of efficiency. Some problems involve heavy amounts of anisotropy, resulting from different random variables having different degrees of impact on the resulting parametric PDE problem, and so the incorporation of this into the sparse grid construction was considered immediately afterwards in [33]. Sparse approximations have also been utilised in works such as [34] for problems involving random source terms, and also in [35, 36] (although in these cases, the model problem is a deterministic elliptic PDE with a stochastic loading term).

While many of the works we have discussed were important to the development of stochastic collocation methods, the incorporation of these into an *adaptive* finite element scheme remained an open question. Among the works which addressed this are [37], which built on the

dimension-adaptive algorithm proposed in [38] to provide an adaptive algorithm driven by residual-based a posteriori error estimation techniques.

Convergence analysis associated with this approach has been performed in both [39] and [40]. Both works make differing assumptions; however, the end result is that the former extends the adaptive algorithm to additional considerations of mesh refinement, and derives some convergence rates in the process, while the latter modifies the adaptive algorithm and is independent of the choice of collocation points. Despite this progress, one notable limiting assumption appearing in both of these works (as well as in [37]) is that the diffusion coefficient associated with the parametric PDE problem admits an affine dependence on random parameters. There are many practical scenarios in which this restriction may be considered prohibitive, due to the underlying physical phenomena requiring a different model to accurately describe the behaviour that needs to be described.

The above problem was solved in [41], which extended the strategies previously developed using hierarchical estimators that allowed the resulting algorithms to cover problems involving coefficients with non-affine parameter dependence. The resulting adaptive strategies proved to be reliable, albeit with sub-optimal convergence rates for certain problems that involved parameter-dependent localised spatial features. This, in turn, has been discussed further in the context of multilevel adaptivity in [42]. Finally, further related convergence analysis has also been performed in [43].

These developments represent some of the most recent work on stochastic collocation FEM approximations and associated adaptive algorithms. However, in many practical applications, we may not specifically be interested in the approximation itself, or attempting to reduce the error of that approximation in some norm (such as the associated Bochner norm). Instead, we may be interested in a particular aspect of the solution, such as its average, or its flux through a given region, amongst many other quantities.

Since the late 1990s, various authors have experimented with procedures designed to estimate specific quantities of interest associated with specific deterministic problems, including drag and lift coefficients in viscous fluid flows [44], problems involving linear elasticity [45], and others. Today, this general area of research is typically known as *goal-oriented* adaptivity, and is relatively well understood in the deterministic setting despite being a more recent area of

research.

Many of the earlier works, such as those above, use a duality argument to describe weights associated with existing residual-based error estimation strategies. However, other works have suggested an alternative approach using orthogonality to induce an estimate based on symmetric products involving primal and dual approximation errors. Some of the earlier work on this approach began with [46], which investigated this approach in an elementary model problem in a single dimension, before quickly being extended to two dimensions in [47].

The above results did not, in general, prove convergence of the adaptive procedure; however, work related to this followed. Among the earlier examples, [48] obtains convergence results for the aforementioned 'dual-weighted' residual approach by driving adaptivity associated with the dual problem alone, with [49] also proving explicit convergence rates, albeit with strong regularity assumptions made on both the primal and dual solution. Convergence of the alternative estimator product approach was demonstrated in [50], with less restrictive assumptions, and further novel contributions to goal-oriented adaptivity continue to this day, with very recent extensions of this method to semi-linear PDE problems demonstrated in [51], for example.

In recent years, the subject of goal-oriented error estimation has also received some attention in the setting of parametric PDE problems. With respect to the intrusive stochastic Galerkin FEM setting, early attempts at goal-oriented error estimation include [52], which describes estimation for linear goal functionals, although the associated adaptive strategy is limiting due to the lack of information about the distribution of the total error. This work was extended in [53], which examined the estimation of linear goal functions in the context of nonlinear PDEs. Results for an analogue of the estimator product strategy were demonstrated in [54]. For the non-intrusive setting, some of the more recent works include [55] in the setting of multi-level Monte Carlo methods, and [56] in the stochastic collocation FEM setting.

1.1 Topics of the thesis

This thesis is primarily concerned with topics related to adaptive algorithms associated with the numerical approximations of solutions to parametric PDE problems. In the subsequent

chapters of this work, we will illustrate all of our ideas using various forms of a stationary diffusion problem involving homogeneous Dirichlet boundaries, in two spatial dimensions, and with deterministic source function.

Our objective is to use a given model problem as a tool to elucidate novel ideas and concepts related to the design of efficient and reliable adaptive algorithms in the stochastic collocation FEM setting. Our model problem represents a common choice that is used throughout the literature, including many of the references provided in the review of the literature we have provided above. Further, we would like to emphasise that while this choice of problem is simple in its design, many of the underlying algorithmic ideas presented in this work, novel or otherwise, are generic, and can therefore be adapted to other model problems with minor modifications. There are three main tasks that we must complete in order to satisfy our objective.

Our first task is the development of robust a posteriori error estimation strategies to determine how accurate our approximations are. These strategies must accomplish two goals, the first of which is the provision of an efficient and reliable estimate at the global level in order to give a criterion for the purposes of terminating our algorithm when we have an accurate enough approximation to our original problem. Our other goal is the generation of localised indicators that allow us to identify the regions which admit the most significant sources of error in our approximation.

From here, our next task is the description of appropriate adaptive algorithms that are constructed using the standard four modules (SOLVE, ESTIMATE, MARK, REFINE) that are typical for adaptive finite element computations [14]. Amongst our requirements are a description of how our localised indicators are used to drive adaptivity via the marking and selective refinement of localised areas that admit the highest errors.

Thirdly, we are required to implement the aforementioned adaptive algorithm. Our attention will also be drawn to the practical consideration of how to compute certain quantities in an efficient manner, and the production of results that illustrate the desired features of our adaptive algorithm. These results must be explained and analysed in the context of representing the theoretical results that our algorithm is built upon.

Some of these tasks have been accomplished in the literature that we have previously cited;

for example, the numerical approximation of solutions to parametric PDEs via stochastic collocation FEM is well understood, and considerable work has been done for a posteriori error estimation for the numerical approximation. However, we seek to provide a comprehensive view of contemporary developments in this active area of research, and contribute our own new theoretical and practical results that fully address each of the three tasks set out above. To this end, our novel contributions are given hence.

1.2 Novel contributions

Firstly, on the subject of stochastic collocation FEM, we provide new experiments with the purpose of analysing aspects of adaptive algorithms in this setting. Among these results, which are presented in Section 3.7, are comparisons of different types of hierarchical spatial error estimation techniques for the purposes of estimating the spatial contributions to the error in a given approximation. We also compare the performance of Leja and Clenshaw–Curtis abscissas within the context of constructing the underlying sparse grids for the stochastic collocation FEM approximations. Moreover, we examine the use of the reduced margin and full margin for the purposes of creating enhanced collocation point index sets.

These results, collectively, provide new insight on the design and implementation of adaptive algorithms in the stochastic collocation FEM setting. This is all provided in addition to an updated theoretical view of some of the surrounding theory of the underlying index sets, including explanations and proofs related to the associated subject matter in the earlier sections of Chapter 3.

Secondly, in Chapter 4, we utilise the pre-existing framework of goal-oriented adaptivity in the deterministic setting (as seen in [50] and others) in order to estimate errors associated with approximating linear quantities of interest. In a similar manner to the ideas seen in works such as [57, 58], this approach introduces an ancillary term in order to compensate for the lack of global Galerkin orthogonality in the stochastic collocation framework. In the deterministic setting, the aforementioned works often refer to to this as an adjoint error 'correction' term, and we will adopt similar terminology throughout our own work.

Our strategy then compartmentalises the error into spatial and parametric components, and

allows for the use of existing error estimation tools, such as the techniques seen in [41], to be used in this setting. In addition, we propose algorithms based on this strategy that incorporate necessary features for the purposes of adaptivity, including consideration of spatial and parametric contributions to the error, as well as the interplay between primal and dual problems.

Adaptive algorithms involving goal-oriented error estimation for stochastic collocation FEM have previously been discussed in [56]. However, while that procedure uses sample-based estimates to generate a corresponding global approximation, our theory is the first in the stochastic collocation FEM setting to use a global estimator product-based a posteriori estimation strategy to drive a goal-oriented adaptive algorithm.

Most notably, our work also represents perhaps the most convenient theory to date that concerns goal-oriented error estimation for PDE problems, for multiple reasons. By working in the stochastic collocation FEM framework, we provide a theory that is immediately applicable to problems with non-affine dependence on random parameters from a bounded sample space. By contrast, the extension of stochastic Galerkin procedures to coefficients with non-affine dependence on random parameters is possible, albeit difficult, due to the resulting linear systems being fully coupled and requiring significant effort to solve efficiently. In addition, our results interface seamlessly with existing global hierarchical estimation techniques. This, in turn, allows our approach to be applied to a range of other parametric PDE problems which are not directly considered in this thesis.

The third novel contribution that we make is the proposition of an alternative approach to goal-oriented adaptivity which utilises a dual-weighted residual method for the purposes of error estimation. This approach, which is introduced in Chapter 5, invokes the same splitting as the estimator-product strategy in order to determine spatial and parametric contributions to the total error in an underlying stochastic collocation FEM approximation. Moreover, we introduce adaptive algorithms based on this strategy within the stochastic collocation FEM setting. Our approach inherits all of the advantages associated with this setting more generally, and hence, provides a promising tool in the context of goal-oriented adaptivity for parametric PDE problems.

Lastly, we produce code modules associated with both of our goal-oriented error estimation

strategies and their associated adaptive algorithms. Each of these is designed to integrate with the open-source toolbox[1] associated with the results in [41]. The reliability and efficiency of our estimation strategies are further demonstrated by numerical experiments produced using this code, which will be made available in due course. We use these results to provide analysis which illuminates intricacies associated with the design of our adaptive algorithms, and gives us insight into potential further applications of our work, as well as possible limitations.

1.3 Thesis structure

In Chapter 2, we restrict our attention to a deterministic model problem in order to recall key ideas that will be utilised in later chapters. We begin with some fundamental material from analysis, which we recall in Section 2.1. From here, we discuss the generation of finite element approximations in the deterministic setting in Section 2.2, along with an overview of adaptive algorithms in Section 2.3. Different components of the adaptive algorithm are discussed in Sections 2.4 - 2.6, followed by aspects of implementation in Section 2.7. We finally discuss the estimation of quantities of interest using finite element algorithms in Sections 2.8 and 2.9.

We turn our attention to the setting of parametric PDEs in Chapter 3. We introduce the probabilistic concepts that are required to establish our problem formulation in Section 3.1. We also provide a wider discussion of some of the approaches that have been utilised to deal with problems within the framework of parametric PDEs. In particular, Section 3.2 covers Monte Carlo methods, as well as the projection-based stochastic Galerkin method, as well as our primary interest, the recently developed stochastic collocation method. Our specific focus, of course, is the last of these three, and after examining the details of sparse grid interpolation in Section 3.3, we discuss stochastic collocation in more detail in Section 3.4. The associated adaptive algorithms are discussed in Section 3.5, while implementation of the algorithm is discussed in Section 3.6. The final part of this chapter, Section 3.7, is dedicated to illustrating the ideas of this chapter with experiments that compare the performance of different strategies for various components of the algorithm.

In Chapter 4, we introduce the goal-oriented setting for stochastic collocation FEM. We introduce the corresponding problem formulations in Section 4.1, including the definition of

[1]The code in question is available at `https://github.com/albespalov/Adaptive_ML-SCFEM`

the goal functional in this framework. Strategies for error estimation and related theoretical results are discussed in Sections 4.2 and 4.3. This includes proofs regarding the hierarchical splitting of the error, as well as the introduction of the novel correction term that plays an important role in this setting. Further details about the calculation of this correction term are then discussed in Section 4.4, including exact representations and quadrature-based approximations. We design the associated adaptive algorithm, with suitable modifications to components such as the marking procedure, in Section 4.5, and discuss our implementation of this algorithm in Section 4.6. The chapter concludes with Section 4.7, which provides numerical results that confirm our theoretical work, and that illuminates some nuances of the approaches that we discussed in the prior sections. This is achieved with a variety of test problems on different domains, with different goal functionals, and using different (including non-affine) diffusion coefficients.

Chapter 5 continues our work from the previous chapter by introducing a residual-based approach to goal-oriented adaptivity in the stochastic collocation FEM setting. We adopt the same problem formulation as in the previous chapter, although the residual-based error estimation strategy is markedly different and more intricate in places. We discuss the theoretical details of this in Section 5.1, invoking the hierarchical error splitting from the previous chapter to analyse spatial and parametric contributions to the resulting residual. A symmetric variant of this approach involving primal weights is discussed in Section 5.2, while we remark upon alternate representations of the right-hand side functionals in 5.3. The evaluation of residuals and corresponding weights associated with this method are discussed at length in Section 5.4. As with the calculation of the correction term in the previous chapter, exact representations and quadrature-based approximations are provided for this purpose. We then proceed to a discussion of the adaptive algorithm in Section 5.5, highlighting the differences between this procedure and the equivalent procedure for the product-based goal-oriented strategy of the previous chapter. This is followed by a description of our implementation in Section 5.6. We complete the chapter by providing numerical results that are designed to compare the dual-weighted residual approach with the approach we described in the previous chapter.

Finally, Chapter 6 presents concluding remarks, as well as a discussion of possible extensions to the theoretical results that we have provided throughout this work.

Chapter 2

Finite Element Algorithms

In the context of general numerical methods, finite element methods are exceptionally popular. Its versatility in terms of dealing with a vast array of problems posed on complex geometries, as well as its intuitive representation of the resulting approximation, are among the advantages noted in [59] and others.

In this chapter, we provide an overview of the standard theory surrounding finite element methods, and some associated concepts. This will begin with a small amount of theory concerning Sobolev spaces, following the general ideas presented in [60] and [61], amongst others. A recapitulation of some of the measure-theoretic ideas can also be found in [62]. From there, we will discuss finite element methods and their associated algorithms in more detail.

2.1 Fundamentals

We first recall that for $p \geqslant 1$, $L^p(D)$ is the Lebesgue space containing functions satisfying:

$$\|v\|_{L^p(D)} := \left(\int_D |v(\boldsymbol{x})|^p \, \mathrm{d}\boldsymbol{x} \right)^{1/p} < \infty, \tag{2.1}$$

or alternatively,

$$\|v\|_{L^p(D)} := \operatorname*{ess\,sup}_{\boldsymbol{x} \in D} |v(\boldsymbol{x})| < \infty \tag{2.2}$$

for $p = \infty$. We also may denote the L^2 norm by $\|\cdot\|_D$, or alternatively, $\|\cdot\|_{D_1}$ for some appropriate subdomain $D_1 \subset D$.

Given $\boldsymbol{x} = (x_1, x_2) \in \mathbb{R}^2$, we denote the *differential operator* of a function of two variables, $u(\boldsymbol{x}) \in L^p(D)$, by:

$$\mathfrak{D}^{\mathfrak{a}} u(x_1, x_2) := \frac{\partial^{\mathfrak{a}} u}{\partial x_1^{\mathfrak{a}_1} \partial x_2^{\mathfrak{a}_2}}, \quad \mathfrak{a}_1 + \mathfrak{a}_2 = \mathfrak{a}.$$

Further, the *weak derivative* of $v \in L^p(D)$, given by $w := \mathfrak{D}^{\mathfrak{a}} v(\boldsymbol{x})$, exists and is well-defined whenever:

$$\int_D u(\boldsymbol{x}) \mathfrak{D}^{\mathfrak{a}} v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = (-1)^{\mathfrak{a}} \int_D w(\boldsymbol{x}) v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}, \quad \forall \, v \in \underline{C}^{\infty}(D), \tag{2.3}$$

where $C^{\infty}(D)$ refers to the set of infinitely continuously differentiable functions on D, and $\underline{C}^{\infty}(D)$ refers to the space of functions contained in $C^{\infty}(D)$ that also admit compact support in D. Note that $\mathfrak{D}$ is well–defined as it is unique, at least for any set with a non-zero Lebesgue measure. Uniqueness almost immediately follows from the fact that $\underline{C}^{\infty}(D)$ is dense in $L^2(D)$ (see, for example, [63]). Using the above, we define the Sobolev space

$$H^1(D) := \{v : D \to \mathbb{R} : \mathfrak{D}^{\mathfrak{a}} v \in L^2(D), \, \forall \, \mathfrak{a} \leqslant 1\}, \tag{2.4}$$

and also the space

$$H_0^1(D) := \{v \in H^1(D) : v|_D = 0\}. \tag{2.5}$$

We also have the associated norm for this space

$$\|v\|_{H^1(D)} := \left(\sum_{|\mathfrak{a}| \leqslant 1} \int_D \|\mathfrak{D}^{\mathfrak{a}} v(\boldsymbol{x})\|_{L^2(D)}^2 \, \mathrm{d}\boldsymbol{x} \right)^{1/2}, \tag{2.6}$$

so that $\|v\|_{H^1(D)} < \infty$ whenever $v \in H^1(D)$. More generally, Sobolev spaces $W^{k,p}$ are defined by functions $v \in L^p(D)$ which satisfy

$$\|v\|_{W^{k,p}} := \sum_{\mathfrak{a} \leqslant k} \left(\|\mathfrak{D}^{\mathfrak{a}} v\|_{L^p(D)}^p \right)^{1/p} < \infty \tag{2.7}$$

for $k \in \mathbb{N}_0$ and $p \geqslant 1$, and for the case of $p = \infty$,

$$\|v\|_{W^{k,\infty}} := \sup_{\mathfrak{a} \leqslant k} \|\mathfrak{D}^{\mathfrak{a}} v\|_{L^p(D)}^p < \infty. \tag{2.8}$$

It is easily observed that taking $k = 0$ yields the Lebesgue spaces L^p for each $1 \leqslant p \leqslant \infty$. Further we typically write $H^k(D)$ to describe the case where $p = 2$ (with the 'H' notation being derived from the fact this special case is also a collection of Hilbert spaces). We can then readily see that the norm in Definition 2.6 is recovered from taking $p = 2$ and $k = 1$.

Let V be a linear vector space over the reals, with norm $\|\cdot\|_V$. Then the dual space, V' is the set of all bounded linear functionals on V. In other words,

$$V' := \{f : V \to \mathbb{R} : \, \exists \, C > 0 \text{ s.t. } |f(v)| \leqslant C \|v\|_V \, \forall \, v \in V\}. \tag{2.9}$$

Note that for the Sobolev space $H_0^1(D)$, we will often write the dual space corresponding to this as $H^{-1}(D)$ (further details with regards to this are available in [60], or alternatively [64]).

2.2 Finite element methods

In the work that follows, we consider a steady-state diffusion problem with homogeneous Dirichlet boundary conditions. Take $D \subset \mathbb{R}^2$ to be a connected, bounded and Lipschitz domain with boundary defined by ∂D. (If required, a definition of a Lipschitz domain, or domain with Lipschitz boundary, can be found in [65].) The closure of D will be denoted by $\bar{D} := D \cup \partial D$ where required. Let $a(\boldsymbol{x})$ be a real 2×2 matrix representing a diffusion coefficient. Also, assume a scalar force term $f(\boldsymbol{x})$. Consider the problem

$$\begin{aligned} -\nabla \cdot (a(\boldsymbol{x}) \nabla u(\boldsymbol{x})) &= f(\boldsymbol{x}), \quad \boldsymbol{x} \in D, \\ u(\boldsymbol{x}) &= 0, \quad \boldsymbol{x} \in \partial D. \end{aligned} \tag{2.10}$$

By employing the usual procedure of multiplying by an arbitrary 'test' function, $v(\boldsymbol{x})$, and integrating, we obtain

$$-\int_D \nabla \cdot (a(\boldsymbol{x}) \nabla u(\boldsymbol{x})) v(\boldsymbol{x}) \ \mathrm{d}\boldsymbol{x} = \int_D f(\boldsymbol{x}) v(\boldsymbol{x}) \ \mathrm{d}\boldsymbol{x}$$

which, using Green's first identity, as stated in [66], becomes

$$\int_D (a(\boldsymbol{x}) \nabla u(\boldsymbol{x})) \cdot \nabla v(\boldsymbol{x}) \ \mathrm{d}\boldsymbol{x} = \int_D f(\boldsymbol{x}) v(\boldsymbol{x}) \ \mathrm{d}\boldsymbol{x}.$$

Here, we can eliminate the boundary integral, and ensure that the integrals above are bounded if we choose an appropriate function space, V, for $u(\boldsymbol{x})$ and $v(\boldsymbol{x})$. In doing this, we can write the *weak formulation* of Problem (2.10), which is given by:

Find $u \in V := H_0^1(D)$ such that

$$B(u, v) = F(v), \quad \forall \, v \in V, \tag{2.11}$$

with

$$\begin{aligned} B(u, v) &:= \int_D a(\boldsymbol{x}) \nabla u(\boldsymbol{x}) \cdot \nabla v(\boldsymbol{x}) \ \mathrm{d}\boldsymbol{x}, \quad \forall \, v \in V, \\ F(v) &:= \int_D f(\boldsymbol{x}) v(\boldsymbol{x}) \ \mathrm{d}\boldsymbol{x}, \quad \forall \, v \in V. \end{aligned} \tag{2.12}$$

Note that B induces its own norm, known as the *energy norm*. This is defined by

$$|||w||| := B(w, w)^{1/2} = \left(\int_D a(\boldsymbol{x}) |w(\boldsymbol{x})|^2 \ \mathrm{d}\boldsymbol{x} \right)^{1/2} \tag{2.13}$$

We can justify that formulation (2.11) is well-posed using the result below.

Lemma 2.1 (Lax–Milgram Lemma)**.** [67] *Suppose that V is a Hilbert space, with dual space V', and that $B : V \times V \to \mathbb{R}$ is a bilinear form. Assume that:*

$$\exists\, \beta > 0 \text{ s.t. } |B(w_1, w_2)| \leqslant \beta \, \|w_1\|_V \|w_2\|_V, \quad \forall\, w_1, w_2 \in V, \tag{$\star$}$$

$$\exists\, \nu > 0 \text{ s.t. } B(w_1, w_1) \geqslant \nu \, \|w_1\|_V^2, \quad \forall\, w_1 \in V. \tag{$\star\star$}$$

Then for any $F \in V'$, we have that $\exists!\ w_1 \in V$ such that

$$B(w_1, w_2) = F(w_2), \quad \forall\, w_2 \in V.$$

Condition ($\star$) is referred to as boundedness, or continuity in some literature, such as [68] and [14]. Condition ($\star\star$) is a coercivity condition, also referred to as ellipticity (on the space V). Recall that for problem (2.11), $V = H_0^1(D)$, and that furthermore, this function space satisfies the required conditions of the lemma. It therefore follows that, as a result of the above lemma, our weak formulation admits a unique solution.

Our objective is now to utilise a finite element to approximate the solution to problem (2.11). In general, we will approximate the solution on a conforming triangulation of the form $\mathcal{T} = \{K_0, K_1, ..., K_{|\mathcal{T}|}\}$, where the elements of $\mathcal{T}$ are simplicies (or *elements*) of the triangulation, and $|\mathcal{T}|$ represents the number of elements in the triangulation. While dealing with the aforementioned conforming triangulation, we define a space of functions, $V_{\mathcal{T}}$, which contains our finite element approximation.

In this work, we primarily consider variations of the triangulation, $\mathcal{T}$, when obtaining different finite element approximations for a given problem. Naturally, it is also possible to consider another approach, where multiple finite element spaces of different polynomial degrees are utilised for a given triangulation, $\mathcal{T}$ [69]. The two approaches are often denoted as the $h-$ and $p-$versions of the finite element method (with the use of 'h' originating from its historical usage in denoting element diameters). These two methods can also be combined, in an approach that is often referred to as the $hp-$version of the finite element method [70]. Depending on our preferred method, we may wish to use an alternative notation for the underlying finite element space, such as $V_{\mathcal{T},p}$ for $hp-$FEM; however, we are not, in general, considering polynomial refinements so this is not necessary here.

In this instance, we define $V_{\mathcal{T}}$ as a function space consisting of piecewise (with respect to elements) polynomials of degree p from the larger function space $V = H_0^1(D)$. That is, we will have a space of the form

$$V_{\mathcal{T}} := \mathcal{S}_0^p(\mathcal{T}) := \left\{ w \in H_0^1(D) : w|_K = \sum_{i=0}^{p} \sum_{j=0}^{p-i} \alpha_{ij} {x_1}^i {x_2}^j \; \forall \, K \in \mathcal{T} \right\}, \quad p \in \mathbb{N}_0. \tag{2.14}$$

We will also define $\mathcal{S}_0^\infty(\mathcal{T})$ to be space of piecewise continuous functions in $H_0^1(D)$ that are continuous on each $K \in \mathcal{T}$. In this framework, we obtain the discretised weak formulation given below.

Find $u_{\mathcal{T}} \in V_{\mathcal{T}}$ such that

$$B(u_{\mathcal{T}}, v_{\mathcal{T}}) = F(v_{\mathcal{T}}), \quad \forall \, v_{\mathcal{T}} \in V_{\mathcal{T}} \subset V. \tag{2.15}$$

Note that, because $V_{\mathcal{T}} \subset V$, we are able to quickly deduce an orthogonality principle. We have that

$$B(u - u_{\mathcal{T}}, v) = 0, \quad \forall \, v \in V_{\mathcal{T}}. \tag{2.16}$$

This result is typically referred to as Galerkin orthogonality, and can be used to establish the *quasi-optimality* of the solution to (2.15) via the following lemma.

Lemma 2.2 (Céa's Lemma)**.** [71] *Suppose V is a Hilbert space with norm $\|\cdot\|_V$, and further, that $B : V \times V \to V$ is a bilinear form that is bounded (with constant β) and V-coercive (with constant ν). Also take $F : V \to \mathbb{R}$ to be a bounded linear functional. Then, the solution to (2.15) is a quasi-optimal approximation to the solution of formulation (2.11). That is,*

$$\|u - u_{\mathcal{T}}\|_V \leqslant \frac{\beta}{\nu} \|u - v\|_V, \quad \forall \, v \in V_{\mathcal{T}}. \tag{2.17}$$

In other words, the above Lemma states that $u_{\mathcal{T}}$ is the best available approximation for the solution to (2.11) in the space $V_{\mathcal{T}}$ (up to the constant β/ν). We can also prove a similar result in the energy norm.

Lemma 2.3. *In addition to the result of Lemma 2.2, if the bilinear form B is symmetric, then we have*

$$|||u - u_{\mathcal{T}}||| \leqslant |||u - v|||, \quad \forall \, v \in V_{\mathcal{T}}. \tag{2.18}$$

Proof. Take $v \in V_{\mathcal{T}}$. Using Galerkin orthogonality (in particular, $B(u - u_{\mathcal{T}}, v - u_{\mathcal{T}}) = 0$ as

$v - u_{\mathcal{T}} \in V_{\mathcal{T}}$), followed by the Cauchy-Schwarz inequality, we have

$$\begin{aligned} |||u - u_{\mathcal{T}}|||^2 &= B(u - u_{\mathcal{T}}, u - u_{\mathcal{T}}) \\ &= B(u - u_{\mathcal{T}}, u - v) + B(u - u_{\mathcal{T}}, v - u_{\mathcal{T}}) \\ &= B(u - u_{\mathcal{T}}, u - v) \\ &\leqslant |||u - u_{\mathcal{T}}||| \cdot |||u - v|||. \end{aligned}$$

The result immediately follows. □

These results are exceptionally useful as they demonstrate that if $\exists v \in V_{\mathcal{T}}$ such that $\|u - v\| \to 0$ or $|||u - v||| \to 0$, then the error of our finite element approximation in the corresponding norm must also tend to zero. Therefore, we cannot have a situation where some other member of $V_{\mathcal{T}}$ becomes arbitrarily close to approximating u unless $u_{\mathcal{T}}$ also does this.

Of course, the magnitude of the error in our approximations is a cause of concern, and while the results presented above are useful, they do not yield a computable quantity that estimates this error. It transpires that we can typically generate a simple a priori estimate for finite element approximations, the precise nature of which will depend on the regularity of the problem, and the degree of polynomials used for approximation. This is the subject of the theorem below.

Theorem 2.4. [14] *Suppose that the finite element function space $V_{\mathcal{T}}$ consists of polynomial approximations of degree p. Suppose also that $u \in V$ is the solution to problem (2.11) with, at least, $u \in H_0^{p+1}(D)$. Let $u_{\mathcal{T}} \in V_{\mathcal{T}}$ be the solution to the discrete problem (2.15).*

$$|||u - u_{\mathcal{T}}||| \leqslant Ch^p \|u\|_{H^{p+1}(D)}, \tag{2.19}$$

with $C > 0$ representing a constant that is independent of the solution $u \in V$, and the polynomial degree, $p \geqslant 1$.

A priori estimates of the type above allow us to describe expected rates of convergence for approximations. This is somewhat useful for practical purposes – for example, testing that adaptive algorithms (see subsequent sections in this thesis) have the intended convergence behaviour. However, the precise value of C, and the derivatives of u, are in general, unknown, and the extent to which they can be immediately controlled via useful bounds is somewhat limited.

We therefore want to obtain estimates that are sufficiently useful in determining the magnitude of the error of our approximation. Furthermore, want to be able to find a way of generating accurate approximations without any of the required calculations becoming too computationally expensive. Hence, we turn to the concept of an adaptive algorithm, which allows us to generate successively more accurate approximations by only the refining parts of the triangulation, $\mathcal{T}$, which admit the greatest errors. The key to this algorithm will be suitable a posteriori error estimation techniques which will be discussed extensively in this thesis; however, in Section 2.3, we will begin to discuss the nature of the adaptive procedures more broadly.

2.3 Adaptive algorithms

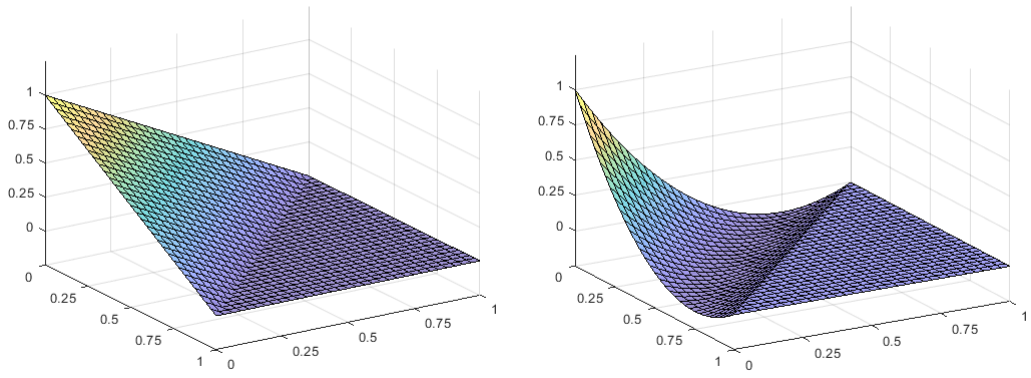
We are given data associated with a problem involving a particular differential equation and a certain tolerance, $\texttt{tol} = \varepsilon$. Our task is then to find a numerical approximation with an error less than ε. To begin, we must initialise our algorithm with a starting triangulation, $\mathcal{T}_0$, to represent the geometry of the domain of the problem. From here, the typical procedure invoked in an adaptive finite element algorithm consists of four different modules, that are repeated in the following order:

$$\text{SOLVE} \rightarrow \text{ESTIMATE} \rightarrow \text{MARK} \rightarrow \text{REFINE}.$$

This adaptive loop is repeated until the estimated global error is less than ε. When the estimated error reaches this critical threshold, the algorithm terminates, and we have our required approximation. We will proceed to describe the implementation of this in Section 2.7; this section will be dedicated to discussing the theoretical aspects of the adaptive loop.

The purpose of the SOLVE step in our adaptive routine is to compute a finite element approximation, $u_{\mathcal{T}} \in V_{\mathcal{T}}$. We have previously discussed relevant function spaces, $V_{\mathcal{T}}$, for a discretisation associated with a conforming triangulation for the domain, D. In particular, the space given in definition (2.14) defines finite element spaces that will be utilised frequently in the discretisations of various problems found in this work.

Figure 2.1 gives us some examples of basis functions that come from this set, in particular. The approximation $u_{\mathcal{T}} \in V_{\mathcal{T}}$ can be written as a sum of basis, or 'hat', functions from $V_{\mathcal{T}}$. Specifically,



(a) (b)

Figure 2.1: Figures (a) and (b) show examples of linear and quadratic basis functions, respectively. These basis functions are taken on a reference element with vertices $(0,0), (0,1)$ and $(1,0)$.

we have a representation of the form

$$u_{\mathcal{T}} = \sum_{j=1}^{N_{\mathcal{T}}} \boldsymbol{u}_j \varphi_j, \tag{2.20}$$

with $\boldsymbol{u} \in \mathbb{R}^N$ representing a vector of coefficients corresponding to the solution of the underlying problem. The basis functions, $\varphi_j \in V_{\mathcal{T}}$, are such that

$$\varphi_j(\boldsymbol{x}_i) = \delta_{ij}, \quad \forall\, \boldsymbol{x}_i \in \mathcal{N}_{\mathcal{T}}, \tag{2.21}$$

where δ_{ij} is the standard Kronecker delta, and $\mathcal{N}_{\mathcal{T}}$ represents nodes associated with the function space $V_{\mathcal{T}}$, with $N_{\mathcal{T}} := |\mathcal{N}_{\mathcal{T}}|$ representing the total number of degrees of freedom. In the case of piecewise linear approximations, these nodes will be the vertices of each element, which typically yields 3 degrees of freedom. By contrast, In the case of piecewise quadratic approximations, these nodes will be the vertices and midpoints of each element, which gives us 6 degrees of freedom, rather than 3.

This linear system will be of the form $\boldsymbol{A}\boldsymbol{u} = \boldsymbol{F}$, where $\boldsymbol{A}$ and $\boldsymbol{F}$ represent a *stiffness matrix* and *right-hand side vector*, respectively. These entities are defined by

$$\boldsymbol{A} = [\boldsymbol{A}_{ij}] := \int_D (A\nabla\varphi_i(\boldsymbol{x})) \cdot \nabla\varphi_j(\boldsymbol{x}) \; \mathrm{d}\boldsymbol{x} \quad \forall\, i,j = 1,...,N, \tag{2.22}$$

$$\boldsymbol{F} = [\boldsymbol{F}_j] := \int_D f(\boldsymbol{x})\, \varphi_j(\boldsymbol{x}) \; \mathrm{d}\boldsymbol{x} \quad \forall\, j = 1,...,N. \tag{2.23}$$

On solving this linear system, we will obtain our approximation, and we will be able to proceed to the next stage in our algorithm.

The ESTIMATE section is a key part of the adaptive algorithm, and typically provides a dual purpose. The first objective of the ESTIMATE module is to furnish our algorithm with a way of determining the global error of our solution. Typically, we will want a global error estimator, μ, that provides a relationship of the form

$$C^\flat \mu \leqslant |||u - u_{\mathcal{T}}||| \leqslant C^\sharp \mu \tag{2.24}$$

for some $C^\flat, C^\sharp > 0$, so that we have upper and lower bounds for the true error of our approximation. This estimator is compared with the predefined tolerance value to determine if the error of our approximation is sufficiently small. If it is, then we will terminate the algorithm at the end of that particular iteration. If it is not, however, then we will need a way of determining how to proceed with further iterations. In particular, it is clear that we will want to adapt our triangulation, $\mathcal{T}$. To do this successfully, we will want to know how the global error is distributed across the domain.

The leads us to the second purpose of our ESTIMATE module. In addition to a global estimator, we want to develop local error estimation indicators in order to determine the error associated with particular edges or elements of our triangulation. Typically, we will denote a local estimator μ_K, and the global estimator μ is often related to the local estimators via the definition

$$\mu := \left(\sum_{K \in \mathcal{T}} \mu_K^2 \right)^{1/2}. \tag{2.25}$$

The MARK module is associated with determining which areas of the triangulation $\mathcal{T}$ have error indicators that are 'sufficiently large' (typically in comparison to other areas within the triangulation). We can mark edges or elements, depending on the approach that we took in our ESTIMATE step, but for now, we will describe the selecting of elements as edge based marking can be done in a similar way. Typically, in order to do this, we employ a marking strategy to determine a threshold for which elements need to be marked for refinement. Examples of possible marking strategies are given in Section 2.5. The end result will be the development of a marked set, $\mathcal{M}_k \subset \mathcal{T}_k$ (with k denoting the current number of iterations of the adaptive loop) that will be utilised in the refinement step.

The final stage is the REFINE module. Assuming that we now have a marked set $\mathcal{M}_k$, the

triangulation $\mathcal{T}_k$ is modified to produce a new triangulation, $\mathcal{T}_{k+1}$. This new triangulation must maintain certain properties, and we will therefore have to ensure that we refine the triangulation in an 'appropriate' manner. More details on precisely how this might be done will be given in Section 2.6. Once the triangulation $\mathcal{T}_{k+1}$ is generated successively, we can proceed with a new iteration in the adaptive algorithm, following the same four steps as before.

At some point, we will obtain a situation at which our estimated global error $\mu < \varepsilon$. It is at this point that the algorithm terminates, and we will have finally produced an approximation, $u_{\mathcal{T}}{}^* \in V_{\mathcal{T}}{}^*$, which is estimated to satisfy problem (2.11) within a certain tolerance. In some of the subsequent Sections in this chapter, we will review key parts of the adaptive algorithm in more detail.

2.4 An overview of a posteriori error estimation

This section will be presented with the objective of providing an overview of a number of different a posteriori error indicators that are used in the context of adaptive algorithms for solving problem (2.11). Recall that we are seeking a situation where relation (2.24) is satisfied. In this relation, the existence of $C^\flat$ ensures that our estimator, μ, is *efficient*. This property allows us to set accurate stopping criteria so that our adaptive algorithms only run for as long as we require. Perhaps more importantly, the existence of $C^\sharp$ allows us to show that our estimator is *reliable*. If we can demonstrate this property, then we can ensure that the true error decays at least as quickly (up to a multiplicative constant) as our error estimator.

We may also compute a quantity known as the effectivity index, given by

$$\Theta_k := \frac{\mu_k}{|||u - u_{(\mathcal{T}_k)}|||}. \tag{2.26}$$

(The brackets in the above definition are provided for clarity.) The purpose of this definition is to yield a practical understanding of how accurately our estimators measure the true error of our solution. In practice, by calculating per-iteration effectivity indices, we may generate a sequence

$$(\Theta_k) = \Theta_1, \Theta_2, ...$$

It is clear that, based on the idea of relation (2.24), we want this sequence, in general, to remain

bounded as we refine our mesh. In the special case that we have

$$\lim_{k\to\infty} \Theta_k = 1,$$

we refer to our error estimator as *asymptotically exact*. In this instance, the error estimate perfectly estimates the energy of the true error when the mesh size of the triangulation tends to zero.

2.4.1 Residual-based estimators

The first type of estimator to be considered is a residual-based estimator. This utilises L^2-based representations for error *residuals* that can be categorised as element-based and edge-based contributions. In particular, recall the Galerkin orthogonality principle (2.16). This equation holds for any given $v \in V_{\mathcal{T}}$. However, if $v \in V \setminus V_{\mathcal{T}}$, then we are left with a non-zero *residual*, denoted $\mathfrak{R}$. Specifically, we have

$$\mathfrak{R}(u_{\mathcal{T}}) := F(v) - B(u_{\mathcal{T}}, v), \quad \forall\, v \in V. \tag{2.27}$$

Recalling problem (2.11), we can use Green's first identity in order to integrate by parts. In the process of doing this, we obtain

$$\begin{aligned}\mathfrak{R}(u_{\mathcal{T}}) &= \int_D fv \ \mathrm{d}x - \int_D (a\nabla u)\cdot\nabla v \ \mathrm{d}\boldsymbol{x} \\ &= \int_D fv + \sum_{K\in\mathcal{T}} \left\{ \int_K \nabla(a\nabla u_{\mathcal{T}})\cdot v \ \mathrm{d}\boldsymbol{x} - \int_{\partial K\setminus\partial D} ((a\nabla u_{\mathcal{T}})\cdot \boldsymbol{n}_K) v \ \mathrm{d}S \right\},\end{aligned}$$

where $\boldsymbol{n}_K$ is the outward unit normal corresponding to the element $K \in \mathcal{T}$. We recall that functions in the space (2.14) are defined in a piecewise manner, which allows for the possibility that these functions may not be smooth at the edges in our triangulation, $E \in \mathcal{E}_{\mathcal{T}}$. Specifically, the integrand contained within the boundary integral above would be different depending on which element we use to perform the calculation. This explains the reason for the decomposition of the integral, but we are also required to define a residual *jump* in order to ensure that this is well-defined.

For each edge, $E \in \mathcal{E}_{\mathcal{T}}$, we designate an associated unit normal vector $\boldsymbol{n}_E$ (for edges on ∂D, this must be the outward normal vector). We then define the *jump* of a function, $v \in \mathcal{S}_0^\infty(\mathcal{T})$, across an edge $E \in \mathcal{E}_{\mathcal{T}}$, in the direction of its associated unit vector $\boldsymbol{n}_E$, by the following:

$$[\![v]\!]_E := \lim_{\xi\to 0^+} v(\boldsymbol{x} - \xi\boldsymbol{n}_E) - \lim_{\xi\to 0^+} v(\boldsymbol{x} + \xi\boldsymbol{n}_E) \quad \forall\, \boldsymbol{x} \in E. \tag{2.28}$$

We immediately observe that if $v \in \mathcal{S}_0^\infty(\mathcal{T})$ and $w \in C^0(D)$, then, $\forall\, \boldsymbol{x} \in E$, we have

$$\begin{aligned}\llbracket vw \rrbracket_E &:= \lim_{\xi \to 0^+} v(\boldsymbol{x} - \xi \boldsymbol{n}_E) w(\boldsymbol{x} - \xi \boldsymbol{n}_E) - \lim_{\xi \to 0^+} v(\boldsymbol{x} + \xi \boldsymbol{n}_E) w(\boldsymbol{x} + \xi \boldsymbol{n}_E) \\ &= \lim_{\xi \to 0^+} v(\boldsymbol{x} - \xi \boldsymbol{n}_E) w(\boldsymbol{x}) - \lim_{\xi \to 0^+} v(\boldsymbol{x} + \xi \boldsymbol{n}_E) w(\boldsymbol{x}) \\ &= w \llbracket v \rrbracket_E \,.\end{aligned}$$

We also notice that for expressions involving normal derivatives, the choice of orientation of the normal vector for the purposes of calculating the jump of this expression is arbitrary. Indeed, given $v = \partial u_\mathcal{T} / \partial \boldsymbol{n}_E$, we have

$$\begin{aligned}\llbracket v \rrbracket_E &= \lim_{\xi \to 0^+} \frac{\partial u_\mathcal{T}}{\partial \boldsymbol{n}_E}(\boldsymbol{x} - \xi \boldsymbol{n}_E) - \lim_{\xi \to 0^+} \frac{\partial u_\mathcal{T}}{\partial \boldsymbol{n}_E}(\boldsymbol{x} + \xi \boldsymbol{n}_E) \\ &= \lim_{\xi \to 0^+} \frac{\partial u_\mathcal{T}}{\partial (-\boldsymbol{n}_E)}(\boldsymbol{x} - \xi(-\boldsymbol{n}_E)) - \lim_{\xi \to 0^+} \frac{\partial u_\mathcal{T}}{\partial (-\boldsymbol{n}_E)}(\boldsymbol{x} + \xi(-\boldsymbol{n}_E)).\end{aligned}$$

Applying definition (2.28) to our previous residual calculation (and omitting subscripts for brevity), we have that

$$\mathfrak{R}(u_\mathcal{T}) = \sum_{K \in \mathcal{T}} \int_K \left(f v + \nabla \cdot (a \nabla u_\mathcal{T}) \right) v \ \mathrm{d}\boldsymbol{x} + \sum_{E \in \mathcal{E}_\mathcal{T}} \int_E \frac{a}{2} \left\llbracket \frac{\partial u_\mathcal{T}}{\partial \boldsymbol{n}} \right\rrbracket v \ \mathrm{d}S.$$

This can be written in the form

$$\mathfrak{R}(u_\mathcal{T}) = \sum_{K \in \mathcal{T}} \int_K r_K(u_\mathcal{T}) v \ \mathrm{d}\boldsymbol{x} + \sum_{E \in \mathcal{E}_\mathcal{T}} \int_E j_K(u_\mathcal{T}) v \ \mathrm{d}S, \tag{2.29}$$

with element-based and edge-based residuals defined as

$$\begin{aligned} r_K(u_\mathcal{T}) &:= f + \nabla \cdot (a \nabla u_\mathcal{T}), \\ j_K(u_\mathcal{T}) &:= \begin{cases} \dfrac{a}{2} \left\llbracket \dfrac{\partial u_\mathcal{T}}{\partial \boldsymbol{n}} \right\rrbracket, & \partial K \subset \partial D, \\ 0, & \partial K \cap \partial D = \emptyset. \end{cases} \end{aligned}$$

Following the example of [72], we can define a locally defined per-element residual estimator by

$$\mu_K := \left\{ h_K^2 \| r_K(u_\mathcal{T}) \|_{L^2(K)}^2 + \frac{1}{2} \sum_{E \subset \partial K} h_E \| j_K(u_\mathcal{T}) \|_{L^2(E)}^2 \right\}^{1/2}, \tag{2.30}$$

where h_E refers to the length of each edge $E \in \partial K$ and h_K is the diameter of the element $K \in \mathcal{T}$. The corresponding global error estimator can be defined using definition (2.25). An upper bound for the error in terms of estimator (2.30) can be deduced using an approach that invokes the stability of problem (2.11), as well as the notion of quasi-interpolation operators (see, for example [73]). A lower bound on the error (and hence, efficiency) is established using basis or 'bubble' functions and the regularity of problem (2.11). Using these principles, we obtain the result below.

Theorem 2.5. [72] *Define the average of the scalar force function f across an element as $\bar{f}_K$. Also define the union of the set of elements sharing an edge with K:*

$$\Lambda_K^{\cup} := \bigcup_{K \in \Lambda_K} K, \quad \Lambda_K := \{K' \in \mathcal{T} : K' \cap K \in \mathcal{E}_{\mathcal{T}}\}.$$

We then have that $\exists\, C^\sharp, C^\flat > 0$, dependent only on the shape regularity of the triangulation $\mathcal{T}$ and the definition of B in problem (2.11), that satisfy

$$|||u - u_{\mathcal{T}}|||^2 \leqslant C^\sharp \sum_{K \in \mathcal{T}} \left(\mu_K^2 + h_K^2 \left\| f - \bar{f}_K \right\|_{L^2(K)}^2 \right) \tag{2.31}$$

$$\mu_K^2 \leqslant C^\flat \left(\|u - u_{\mathcal{T}}\|_{H^1(\Lambda_K^{\cup})}^2 + \sum_{K' \in \Lambda_K} h_{K'}^2 \left\| f - \bar{f}_{K'} \right\|_{L^2(K')}^2 \right) \tag{2.32}$$

The second term in each of the bracket of the right-hand sides of inequalities (2.31) and (2.32) are 'data oscillation' terms. The upper bound (2.31) is a global estimate for the domain, D, whereas the lower bound (2.31) is defined at a more local level. Specifically, for each element, the estimator depends only on that particular element and any elements that are directly adjacent to it. One important point to note is that the integrals contained within these estimators may be impossible (or at least computationally expensive) to calculate depending on the nature of the scalar force function, f. In this case, a suitable quadrature rule may be utilised.

2.4.2 Hierarchical estimators

Another approach to a posteriori error estimation is to attempt to find a second approximation to problem (2.11) and then to compare with our initial approximation. In this framework, we have the discrete formulation (2.15), and a new discrete formulation. We utilise a *detail space*, $\mathfrak{Y}$, that is a subspace of the original function space that problem (2.11) was posed on, and that is (near) orthogonal to the finite element space that the original discrete formulation is posed on. We then define

$$\widehat{V}_{\mathcal{T}} := V_{\mathcal{T}} \oplus \mathfrak{Y},$$

so that we have an enriched space, $\widehat{V}_{\mathcal{T}}$, that is a direct sum of the original discrete function space and the detail space. The detail space itself, $\mathfrak{Y}$, can consist of higher-order functions, or functions with piecewise linearity with respect to a uniformly refined triangulation, $\widehat{\mathcal{T}}$. If

the original finite element space consists of piecewise linear functions, $\mathcal{S}_0^1(\mathcal{T})$, then these two choices will consist of functions from the spaces $\mathcal{S}_0^2(\mathcal{T})$, and $\mathcal{S}_0^1(\widehat{\mathcal{T}})$, respectively. Examples of the former can be obtained from scaled products of basis functions from $\mathcal{S}_0^1(\mathcal{T})$,

$$\phi_{i,j} := 4\varphi_i\varphi_j, \quad \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{N}_E \quad (\boldsymbol{x}_i \neq \boldsymbol{x}_j). \tag{2.33}$$

By construction, $\phi_{i,j}(\boldsymbol{x}) \in \mathcal{S}_0^2(\mathcal{T})$, and furthermore,

$$\begin{aligned} &\phi_{i,j}(\boldsymbol{x}_k) = \delta_{kl}, \quad \boldsymbol{x}_l = \frac{\boldsymbol{x}_i + \boldsymbol{x}_j}{2}, \quad \boldsymbol{x}_k \in \widehat{\mathcal{N}}_K \setminus \{\boldsymbol{x}_l\}, \\ &\phi_{i,j}(\boldsymbol{x}_k) = 0, \quad \boldsymbol{x}_k \in \mathcal{N}_E, \end{aligned} \tag{2.34}$$

where $\widehat{\mathcal{N}}_K$ is the set of midpoints associated with an element, K, which contains the edge, E. In other words, each $\phi_{i,j}$ satisfies the Kronecker-delta property across a set of element midpoints and is zero at each node. We then have the formulation on the enriched space:

Find $\widehat{u}_\mathcal{T} \in \widehat{V}_\mathcal{T}$ such that

$$B(\widehat{u}_\mathcal{T}, \widehat{v}_\mathcal{T}) = F(\widehat{v}_\mathcal{T}), \quad \forall\, \widehat{v}_\mathcal{T} \in \widehat{V}_\mathcal{T}. \tag{2.35}$$

We can then compute our second approximation on $\widehat{V}_\mathcal{T}$, and compare this with our solution on $V_\mathcal{T}$. To this end, note that $V_\mathcal{T} \subset \widehat{V}_\mathcal{T} \subset V$, so that we have the Galerkin orthogonality relation

$$B(u - \widehat{u}_\mathcal{T}, \widehat{v}_\mathcal{T}) = 0, \quad \forall\, \widehat{v}_\mathcal{T} \in \widehat{V}_\mathcal{T}. \tag{2.36}$$

Recalling $|||u|||^2 = B(u,u)$ and assuming that B is symmetric, we have that

$$|||u - \widehat{u}_\mathcal{T}|||^2 + |||\widehat{u}_\mathcal{T} - u_\mathcal{T}|||^2 = |||u - u_\mathcal{T}|||^2 \tag{2.37}$$

via orthogonality. In other words, the true error of our original approximation can be decomposed into a sum of the error of approximation on $\widehat{V}_\mathcal{T}$ and the error reduction achieved between refinements. It is trivial that equation (2.37) allows us to see that the approximation $\widehat{u}_\mathcal{T} \in \widehat{V}_\mathcal{T}$ is at least as accurate as the approximation $u_\mathcal{T} \in V_\mathcal{T}$. We will assume, in fact, that the approximation on the enriched space is more accurate - that is, that $\exists\, \varkappa \in (0,1)$ such that

$$|||u - \widehat{u}_\mathcal{T}||| \leqslant \varkappa\, |||u - u_\mathcal{T}|||. \tag{2.38}$$

This condition is known as a *saturation assumption*, and we can immediately use this in our work on developing a hierarchical error estimator.

Our starting point is to prove the equivalence between the true error and error reduction. We note that equation (2.37) is enough to immediately demonstrate a lower bound for our true error. Assuming that inequality (2.38) holds, we then have

$$|||\widehat{u}_{\mathcal{T}} - u_{\mathcal{T}}||| = (1 - \varkappa^2)|||u - u_{\mathcal{T}}|||.$$

This yields the two-sided error bound

$$|||\widehat{u}_{\mathcal{T}} - u_{\mathcal{T}}||| \leqslant |||u - u_{\mathcal{T}}||| \leqslant (1 - \varkappa^2)^{-1/2} \, |||\widehat{u}_{\mathcal{T}} - u_{\mathcal{T}}|||. \tag{2.39}$$

We observe that the norm $|||\widehat{u}_{\mathcal{T}} - u_{\mathcal{T}}|||$ represents the error reduction obtained by refinement, in the energy norm. From the above, we can conclude that this error reduction represents an efficient and reliable estimate of the true energy error of our original approximation $u_{\mathcal{T}} \in V_{\mathcal{T}}$.

Although the error reduction can be used as a hierarchical estimator, the computational expense associated with this calculation is a significant problem. At each iteration of an adaptive algorithm, we already calculate $u_{\mathcal{T}} \in V_{\mathcal{T}}$ by virtue of it being our approximation to the solution of problem (2.11). However, the error reduction also requires us to calculate the enhanced approximation $\widehat{u}_{\mathcal{T}} \in \widehat{V}_{\mathcal{T}}$ during each iteration. If possible, we want to avoid computing this second approximation at each iteration in the algroithm.

One suitable solution to this would be to invoke a hierarchical splitting with near orthogonal function spaces that satisfy a strengthened version of the Cauchy-Schwarz inequality - namely (see [74], amongst others), that $\exists\, \lambda \in [0, 1)$ such that

$$|B(u_{\mathcal{T}}, v_{\mathcal{T}})| \leqslant \lambda |||u_{\mathcal{T}}||| |||v_{\mathcal{T}}||| \quad \forall\, u_{\mathcal{T}} \in V_{\mathcal{T}},\ v_{\mathcal{T}} \in \mathfrak{V}. \tag{2.40}$$

We proceed by invoking the standard hierarchical approach presented in [75] amongst others. Let $\widehat{e}_{\mathcal{T}} \in \widehat{V}_{\mathcal{T}}$ be the unique solution to

$$B(\widehat{e}_{\mathcal{T}}, \widehat{v}_{\mathcal{T}}) = F(\widehat{v}_{\mathcal{T}}) - B(u_{\mathcal{T}}, \widehat{v}_{\mathcal{T}}) \quad \forall\, \widehat{v}_{\mathcal{T}} \in \widehat{V}_{\mathcal{T}}. \tag{2.41}$$

Via problem (2.35), we have that

$$B(\widehat{e}_{\mathcal{T}}, \widehat{v}_{\mathcal{T}}) = B(\widehat{u}_{\mathcal{T}} - u_{\mathcal{T}}, \widehat{v}_{\mathcal{T}}) \quad \forall\, \widehat{v}_{\mathcal{T}} \in \widehat{V}_{\mathcal{T}}. \tag{2.42}$$

Hence, by taking $\widehat{v}_{\mathcal{T}} = \widehat{e}_{\mathcal{T}} := \widehat{u}_{\mathcal{T}} - u_{\mathcal{T}}$, it is clear that

$$B(\widehat{e}_{\mathcal{T}}, \widehat{e}_{\mathcal{T}}) = |||\widehat{e}_{\mathcal{T}}|||^2 = |||\widehat{u}_{\mathcal{T}} - u_{\mathcal{T}}|||^2. \tag{2.43}$$

In dealing with this setup, we would prefer to have a suitable function, $e_{\mathfrak{Y}} \in \mathfrak{Y}$ to utilise as our estimator. We can, in fact, project problem (2.41) onto our detail space, $\mathfrak{Y}$, in order to define a modified problem that requires us to find such a function. To be precise, we consider the problem

Find $e_{\mathfrak{Y}} \in \mathfrak{Y}$ such that

$$B(e_{\mathfrak{Y}}, v_{\mathfrak{Y}}) = F(v_{\mathfrak{Y}}) - B(u_{\mathcal{T}}, v_{\mathfrak{Y}}) \quad \forall\, v_{\mathfrak{Y}} \in \mathfrak{Y}. \tag{2.44}$$

There will necessarily be a loss of accuracy in doing this; however, this loss in accuracy is captured by inequality (2.40), as the following result shows. This then leads us to the key result for our hierarchical estimator.

Theorem 2.6. *Suppose that the saturation assumption given in inequality (2.38) holds, and that we also have the strengthened Cauchy-Schwarz inequality (2.40). Further, define $e_{\mathfrak{Y}} \in \mathfrak{Y}$ as the solution to problem (2.44), and take the error estimate*

$$\mu := |||e_{\mathfrak{Y}}|||. \tag{2.45}$$

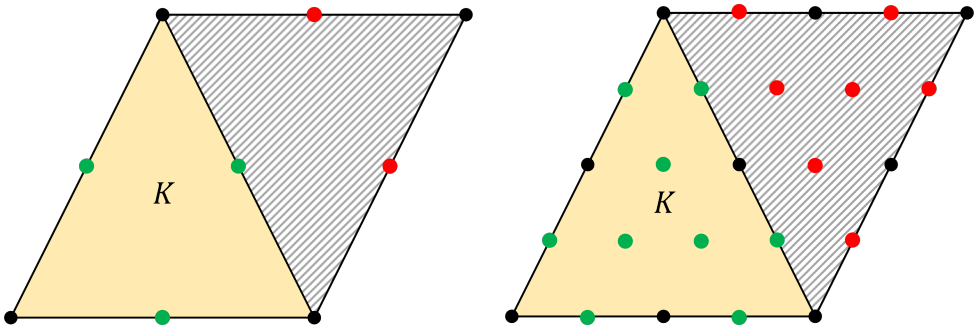
Then we have the two-sided error bound

$$\mu \leqslant |||u - u_{\mathcal{T}}||| \leqslant \frac{1}{\sqrt{1-\varkappa^2}\sqrt{1-\lambda^2}}\mu. \tag{2.46}$$

This theorem yields the efficiency and reliability of estimator (2.45). Note that, unlike with the previously discussed residual estimator, both bounds presented in the theorem above are global, and cannot, by themselves, be used as local estimators in the context of the adaptive algorithm. To fully furnish our adaptive algorithm with the information required for the estimation stage, further work is therefore needed. In addition, we are still required to solve linear systems associated with problem (2.44) throughout the adaptive process.

2.4.3 Two-level error estimators

We now briefly discuss a third error estimation technique. This 'two-level' error estimation technique begins with the ideas introduced for hierarchical error estimators, but does not have all of the problems associated with estimator (2.45). In particular, having observed that estimate (2.39) provides a hierarchical estimator by itself, we now seek an alternative way of avoiding the computation of the quantity $\widehat{u}_{\mathcal{T}}$, and having to solve the additional problem (2.41).



(a) (b)

Figure 2.2: A representation of the idea underpinning inequality (2.47). Figures (a) and (b) show the nodes associated with linear and quadratic basis functions, respectively

In this case, we would proceed by proving some intermediary results involving upper and lower bounds on norms involving the numerical solutions in the detail space $\mathfrak{Y}$, and orthogonal projections onto finite element spaces. On doing this, we are able to establish the core estimation result that we require. This type of strategy goes back to the work found in [76] and others, typically in the context of boundary integral operators. Since then, this approach has successfully been applied to finite element analysis in the deterministic setting, and is, in fact, a special case of the results proved in [54].

Illustrating the final result requires us to describe a few elementary concepts. Let $\mathfrak{F}$ be the set consisting of all basis functions from the detail space $\mathfrak{Y}$, and define $\mathfrak{K}_{\mathfrak{Y}} > 0$ to be a constant given by:

$$\mathfrak{K}_{\mathfrak{Y}} = \min\{k \in \mathbb{N} : |\{\varphi_j \in \mathfrak{F} : \text{int}(\text{supp}(\varphi_j) \cap K) \neq \emptyset\}| \leqslant k, \quad \forall\, K \in \mathcal{T}|\} \tag{2.47}$$

We also denote the number of basis functions across our entire triangulation as $\mathfrak{K}$. Using these definitions, we now describe the two-level error estimator and the associated bounds using the following theorem.

Theorem 2.7. *Suppose that $u \in V$ is the solution to problem (2.11), and that $u_{\mathcal{T}} \in V_{\mathcal{T}}$ and $\widehat{u}_{\mathcal{T}} \in \widehat{V}_{\mathcal{T}}$ are the solutions to problem (2.15) and (2.35), respectively. Let φ_j be basis functions corresponding to this set $\mathfrak{F}$.*

Define the estimator

$$\mu := \sqrt{\sum_{j=1}^{|\mathfrak{K}|} \frac{|F(\varphi_j) - B(u_{\mathcal{T}}, \varphi_j)|^2}{\left\|a^{1/2}\nabla\varphi_j\right\|^2_{L^2(D)}}}. \tag{2.48}$$

Then we have that $\exists\, C > 0$, *such that*

$$\frac{1}{\mathfrak{K}_{\mathfrak{Y}}}\mu^2 \leqslant |||\widehat{u}_{\mathcal{T}} - u_{\mathcal{T}}|||^2 \leqslant C\mu^2, \tag{2.49}$$

where $\mathfrak{K}_{\mathfrak{Y}} > 0$ *is as defined in equation (2.47).*

This result is exceptionally useful as we can use this to attain a bounds on the error in the energy norm. In particular, if saturation assumption (2.38) holds, then it follows from equation (2.37) that

$$\frac{1}{\mathfrak{K}_{\mathfrak{Y}}}\mu^2 \leqslant |||u - u_{\mathcal{T}}|||^2 \leqslant \frac{C}{1-\varkappa^2}\mu^2. \tag{2.50}$$

We also note that while estimator (2.48) is a reliable and efficient global estimator, we can also use this for local error estimation. We observe that this estimator includes all basis functions for each element for a given triangulation. In principle, we can modify the estimator (2.48) so that it only corresponds to basis functions associated with local refinement of, say, a given edge in the triangulation, taking each per-function contribution:

$$\mu_j := \frac{|F(\varphi_j) - B(u_{\mathcal{T}}, \varphi_j)|}{\left\|a^{1/2}\nabla\varphi_j\right\|_{L^2(D)}}, \quad \forall\, j = 1, ..., |\mathfrak{K}|. \tag{2.51}$$

Given the skeleton (or set of edges), $\mathcal{E}$, we can, for edges, $E_i \in \mathcal{E}$, write a direct sum of the form:

$$\bigoplus_{i:E_i\in\mathcal{E}} \mathfrak{F}_{E_i} = \mathfrak{F},$$

where $\mathfrak{F}_{E_i}$ correspond to basis functions from the detail space $\mathfrak{Y}$ associated with a particular edge E_i. Suitable edge indicators can then be written as:

$$\mu_{E_i} := \sqrt{\sum_{j:\varphi_j\in\mathfrak{F}_{E_i}} \mu_j^2}, \quad \forall\, E_i \in \mathcal{E}. \tag{2.52}$$

Local two-sided bounds analogous to estimator (2.48) can then be established. We also note that element-based indicators can be generated using a similar procedure, and with similar results. By iterating these ideas, we can obtain estimators and bounds analogous to those in Theorem (2.7) for arbitrary conforming triangulation obtained as a result of refining the current triangulation, $\mathcal{T}$.

2.5 Marking strategies

We now turn to marking strategies. In what follows, we will primarily discuss the marking of elements in a triangulation $K \in \mathcal{T}_k$. However, we also note that it is perfectly feasible to

use edges, $E \in \mathcal{E}_k$, for these purposes as well. Let $|\mathcal{T}_k|$ represent the number of elements (or simplicies) in the triangulation at refinement level k. Also let $|\mathcal{M}_k|$ be the number of elements marked for refinement at iteration k.

In terms of what this marking strategy needs to accomplish, it is clear that we want to ensure through our adaptive process that the error is distributed across the domain roughly evenly. If it were not, then our calculations may not be particularly efficient, as we will have made additional calculations in regions of comparatively low error – something which our adaptive algorithm is attempting to avoid. Therefore, typical marking strategies that we use will attempt to use some heuristic method of evenly distributing the error as we proceed.

We have multiple examples of choices for our marked set $\mathcal{M}_k$, some of which are more sensible than others for our purposes. The first (and simplest) example of a possible choice for $\mathcal{M}_k$ would be to select

$$\mathcal{M}_k := \mathcal{T}_k. \tag{2.53}$$

This denotes the marking of every element for refinement - that is, we would perform a global refinement of our mesh. Even with this choice, we can (eventually) ensure that our global error decays to a given tolerance. However, this doesn't really describe an *adaptive* finite element method, and we would prefer to have a more competent strategy that allows use to refine our mesh locally.

One common strategy that can be used for locally refining our triangulation would be a 'maximum' strategy. Following the principle set out in [77], we define a parameter $\theta \in (0, 1)$. We then select the set

$$\mathcal{M}_k := \{K \in \mathcal{T}_k : \mu_k(K) > \theta \cdot \max_{K \in \mathcal{T}_k} \mu_k(K)\} \tag{2.54}$$

The idea of this strategy is that we compare the errors across elements to that of the element with the largest error. We then mark elements that have an error that is at least some fraction of the largest error on any element in the triangulation $\mathcal{T}_k$.

We can vary θ in order to control how aggressively we want to pursue marking of elements at each iteration. For higher values of θ, relatively few elements will be marked, and as $\theta \to 1$, the number of elements marked will decrease towards 1 (the element with the largest error indicator). Conversely, for lower values of θ, more elements will be marked at each iteration,

and in the limiting case $\theta \to 0$, we have

$$\lim_{\theta \to 0} |\mathcal{M}_k| = |\mathcal{T}_k|,$$

so that this strategy begins to resemble uniform refinement for small θ.

We can also attempt to define our marked set more explicitly on the principle of distributing the error evenly. This yields the 'equidistribution' strategy, which we begin to motivate using the proposition below.

Proposition 2.8. *Assuming that the error was equidistributed (that is, the indicators $\mu_k(K)$ have the same error), the resulting marked set is of the form*

$$\mathcal{M}_k := \left\{ K \in \mathcal{T}_k : \mu_k(K) > \frac{\theta\varepsilon}{\sqrt{|\mathcal{T}_k|}} \right\} \tag{2.55}$$

where $\theta > 0$, and ε is a tolerance indicator.

Proof. Assuming the error indicators were equidistributed, we would have that $\exists\, c > 0$ such that

$$\mu_k(K) = c, \quad \forall\, K \in \mathcal{T}_k.$$

Squaring both sides and summing over all elements $K \in \mathcal{T}_k$, we then have

$$\sum_{K \in \mathcal{T}_k} \mu_k^2(K) = c^2\, |\mathcal{T}_k|.$$

Suppose that ε is a desired tolerance for the quantity set out in definition (2.25). We have that on a triangulation $\mathcal{T}_k$ achieving this tolerance,

$$\mu_k^2(K) = \frac{\varepsilon^2}{|\mathcal{T}_k|}$$

from which it follows that we can define a threshold value

$$\bar{\mu_k} := \frac{\theta\varepsilon}{\sqrt{|\mathcal{T}_k|}}$$

for some $\theta > 0$, and mark elements with error larger than this threshold value. The marked set (2.55) follows. □

Thus, the motivation for this strategy is very clear; however, the idea set out in marked set (2.55) has one major problem. Specifically, the presence of ε means that this set is not invariant under

scaling of the problem. Hence, in practice, a sufficiently small tolerance parameter will lead to almost every element being marked on coarse grids. A possible modification to set (2.55), which improves on the above proposition, involves substituting ε for an indicator-dependent quantity. This yields the marked set

$$\mathcal{M}_k := \left\{ K \in \mathcal{T}_k : \mu_k(K) > \theta \sqrt{\frac{\sum_{K \in \mathcal{T}_k} \mu_k^2(K)}{|\mathcal{T}_k|}} \right\}, \tag{2.56}$$

which represents a more sensible choice for a marking strategy.

We also have the 'equilibration' or 'Dörfler' marking strategy, presented in [15]. This strategy involves continuously adding elements $K \in \mathcal{T}_k$ to the marked set $\mathcal{M}_k$, in order of size, until such a point that

$$\sum_{K \in \mathcal{M}_k} \mu_k^2(K) \geqslant \theta \sum_{K \in \mathcal{T}} \mu_k^2(K) \tag{2.57}$$

holds for some pre-selected $\theta \in (0, 1)$. Note that while θ takes the same range of values as in marked set (2.54), increasing and decreasing θ actually has the converse effect – that is, large values of θ yield a large set $\mathcal{M}_k$ and smaller values lead to very few elements being marked. This method of marking is exceptionally useful as it leads to useful results about convergence of adaptive algorithms, as mentioned, for example, in [16, 19] and others.

Finally, we have one last possibility for our marked set. We may also choose a percentile-based 'hybrid' strategy that can be applied in conjunction with some of the previously outlined strategies. The theoretical motivation for such a strategy is not immediately obvious; however, such a motivation is given in [72] for a phenomenon that can occur in many convection dominated problems amongst others.

In these problems, it will be the case that there are some elements with smaller errors, and some elements of large errors, as in other scenarios; however, there may also be a very small number of elements with errors of an even greater order of magnitude. The result of this is that while we want to refine the elements in the second set (those which still have relatively large errors), they will not be refined due to the existence of the small number of elements with extremely large errors. Clearly, this leads to an inefficiency in our algorithm. To rectify this, we can choose a parameter $\varepsilon \in (0, 1)$, although this value will typically be around $\varepsilon = 0.05$. Then define the marked set as

$$\mathcal{M}_k := \mathcal{M}^+ \cap \mathcal{M}^- \tag{2.58}$$

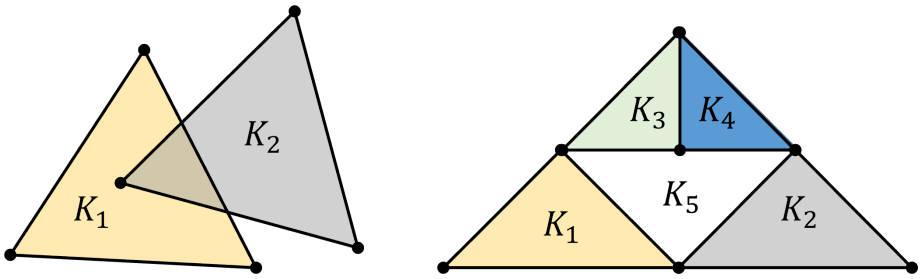


Figure 2.3: Examples of triangulations which are not conforming triangulations. Note, in particular, the hanging node in example (b).

where

$$\mathcal{M}^+ := \{K \in \mathcal{T}_k : \exists\, \mathcal{K} = \{K_1, ..., K_{\mathfrak{m}}\} \text{ with } \mu_k(K) > \mu_k(K_i)\ \forall\, K_i \in \mathcal{K}\},$$

$$\mathfrak{m} := \lceil |\mathcal{T}_k|(1-\varepsilon) \rceil,$$

and $\mathcal{M}^-$ is a marking set obtained from another strategy, such as (2.54), applied to the set $\mathcal{T}_k \setminus \mathcal{M}^+$. In other words, we take a small fraction, ε, of the elements with the largest errors and mark them first, before applying a different marking strategy to the remaining set of elements.

2.6 Mesh refinement

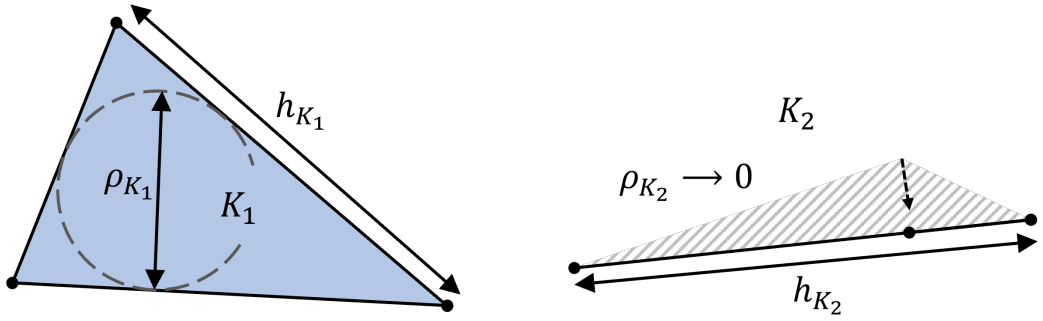
In producing successive triangulations, there are two key features that we need to maintain. The first of these features is a *conforming triangulation* – that is, for a given triangulation, $\mathcal{T}$, we require that the intersection of any two simplicies must be a non-empty 'sub-simplex'. In 2 dimensions, this will either be a single vertex or a single edge (see Figure 2.3).

Another property of our triangulation that we must control throughout the adaptive process is the fact that the elements in our triangulation must maintain their shape. We let individual elements have diameter h_K, and we also associate with our elements the quantity ρ_K, defined as

$$\rho_K = \sup\{2r : \mathfrak{B}_r \subset K\},$$

where $\mathfrak{B}_r$ refers to a ball of radius r. A given triangulation $\mathcal{T}_k$ is *shape regular*, if $\exists\, \vartheta_k > 0$ such that the quantity

$$\mathfrak{S}_k := \max_{K \in \mathcal{T}_k} \left\{ \frac{h_K}{\rho_K} \right\} < \vartheta_k.$$



(a) (b)

Figure 2.4: Measurements associated with the simplex K_1 and degenerate simplex K_2. A triangulation with K_2 has lost shape regularity.

In order to maintain shape regularity throughout the entire adaptive algorithm, we require that the sequence $(\mathfrak{S}_k)_{k=0}^n$ is bounded uniformly by a constant $\vartheta > 0$. In this case, we refer to the sequence of triangulations $(\mathfrak{S}_k)_{k=0}^n$ as being *uniformly shape regular*. Practically speaking, this necessitates that we avoid creating new elements in refined triangulations that are increasingly long in comparison to their width (see Figure 2.4). We also note that the quantity $\max_{K\in\mathcal{T}_k}\{h_K\}$ is often referred to as the *mesh width*, or *mesh size*, of a given triangulation, $\mathcal{T}$.

There are a different ways for us to refine a given element. We will mainly focus on bisection refinement techniques due to the fact that they are commonly used, and allow for local refinements without any major difficulties.

To illustrate the idea behind bisection, consider a conforming triangulation $\mathcal{T}_k$, and suppose that we have an element $K \in \mathcal{T}_k$ (or an edge belonging to this element) that has been marked for refinement. We bisect an edge belonging to K by connecting the opposite vertex to the midpoint of the edge. When we have bisected all of the edges required by our marked set $\mathcal{M}_k$, we may have hanging nodes left by this process. We therefore complete the procedure by bisecting edges opposite hanging nodes in order to restore the conformity of the triangulation.

We must be careful as to how we select the edges that we wish to bisect, however. Depending on our choice of edges, we could potentially continually perform a bisection routine in such a way that we would lose shape regularity for our triangulations (see Figure 2.5 for a demonstration).

We must therefore bisect our elements in a way that prevents this from happening. One

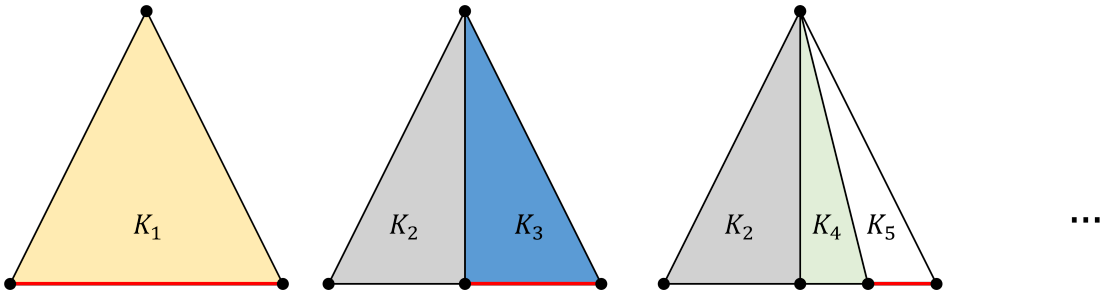


Figure 2.5: A poor choice of edges when bisecting elements during successive iterations can lead to a loss of shape regularity.

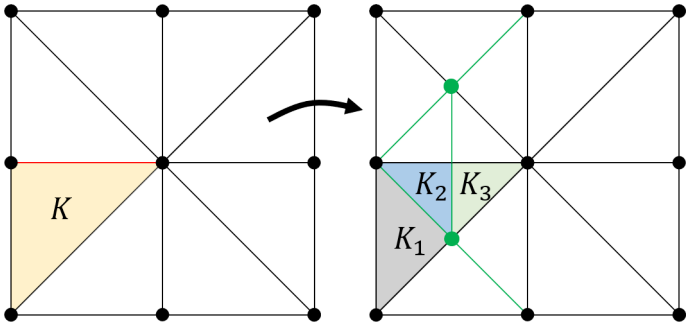


Figure 2.6: The result of a longest edge bisection procedure, assuming that we eventually want to bisect the red edge. The new vertices and edges we generate in the process are marked in green.

possible way of resolving any potential issues is by invoking a procedure known as *newest vertex bisection*. For this technique, the edge that is subject to refinement is always placed opposite the newest vertex produced via previous refinement. An initial vertex of each element in the initial triangulation, $K \in \mathcal{T}_0$ must be chosen. After this, however, the newest vertex, and hence, the edge that is selected for refinement, is inherited automatically by this rule. A comparison between this technique and other mesh refinement methods may be seen in [78]. Note also that this method is always guaranteed to terminate, and that we are never required to continue bisecting ad infinitum to, for example, resolve issues pertaining to mesh conformity. For a proof of this, see [79].

Another potential refinement strategy is a method known as *longest edge bisection*. This procedure, which is a variant of newest vertex bisection, selects the longest edge in a particular element to be the refinement edge (and the newest vertex is chosen to be opposite this edge). The motivation for doing this is that the largest angle within an element will be bisected, and this in turn will preserve shape regularity in the process.

A demonstration of a potential outcome from this procedure is given in Figure 2.6. Note that in this example, the designated edge for bisection for the element K is the diagonal edge. We must apply the bisection recursively in order to refine edges which are not currently the longest edge for a given element. Either one, two, or three bisections can be performed in this way.

Work has been done on the complexity and general practicalities of refinement, including the following result, which gives us an upper bound on the size of the number of elements in our triangulation after k iterations.

Proposition 2.9. [18] *Using a suitably well-defined newest edge bisection process, there exists a sequence of conforming triangulations $\mathcal{T}_0, \mathcal{T}_1, ..., \mathcal{T}_k$ that satisfy the property that $\exists\, \mathcal{C} > 0$, depending only on $\mathcal{T}_0$ such that*

$$|\mathcal{T}_k| \leqslant |\mathcal{T}_0| + \mathcal{C} \sum_{i=1}^{k-1} |\mathcal{M}_i|. \tag{2.59}$$

The above result is useful as it ensures that we can provide some form of control over the number of elements in a triangulation given a certain number of iterations. We can therefore guarantee that we will not produce new elements ad infinitum during the refinement step and we still have a tractable discretisation for our next iteration.

Of course, in order to produce an optimal final triangulation, we must also employ mesh coarsening routines alongside our refinement procedure (see [17, 18]). There are additional complications involved with transient problems due to the fact that singularities may not be located in the same position at different points in the time domain being studied, a point which is noted in [72] amongst others. Hence, mesh coarsening has seen some usage, particularly in the context of parabolic problems, for example in [80] and [81]. However, this complicates our adaptive routine beyond what is necessary for the discussion of the material in this thesis and is not as important for elliptic problems due to their stationary nature. We will therefore omit this from the remainder of our discussions.

2.7 Implementation via T-IFISS

In this section, we will briefly discuss the MATLAB-based software T-IFISS (Triangular Incompressible Flow and Iterative Solver Software), as first presented in [82]. This is a toolbox

that is designed to solve problems of the form seen in equation (2.10) – that is, two-dimensional steady-state diffusion problems.

A given test problem is selected, and the algorithm then proceeds to the main driver for these problems, `adiff_adaptive_main`. This driver acts as the central part of the adaptive loop, and all of the major functions required for the different modules in the adaptive algorithm are called via this script. The script begins with some further initialisation, including the selection of estimation strategies and initial mesh size, amongst other parameters. After an initial domain has been generated, we begin with the `while` loop that contains the core part of the adaptive algorithm.

As discussed previously, the solution step is implemented first in our adaptive loop. This stage is implemented in the usual way, with the generating of a linear system to be solved consisting of the stiffness matrix, $\boldsymbol{A}$, and right-hand side vector, $\boldsymbol{F}$. The required entities are computed locally in a loop over a set of Gaussian points to allow for non-constant diffusion coefficients. The local contributions to the global matrices and source vector can then be assembled. The solution is then found by calculating $\boldsymbol{u} = \boldsymbol{A}^{-1}\boldsymbol{F}$ to find the coefficients in representation (2.20).

In the case of linear (P_1-based) approximations to the solution of a given problem, we have a choice of three different estimation strategies. The first strategy is a local hierarchical error estimation variant that is calculated by using basis functions that have been localised to sub-elements (elements that would be obtained under uniform refinement) in order to resolve element-wise residuals. This technique, which is explained further in [83] allows for a choice of piecewise linear or piecewise quadratic basis functions. This estimation technique is implemented via the function `diffpost_p1_with_p1` or `diffpost_p1_with_p2` (dependent on the choice of basis functions). The implementation of this estimate proceeds by assembling an element-wise stiffness matrix from sub-element contributions. Following this, per-element right-hand side vectors are created by comparing element-based and edge-based residuals. A matrix system is therefore produced, which can be solved to produce element-wise error estimates.

A second approach that has been employed is a standard global hierarchical estimator (see [75]) that uses piecewise linear basis functions associated with sub-elements. This time, however, a sparse linear system corresponding to the global residual problem must be resolved. The

result can then be localised to yield either element-based or edge-based error indicators. The implementation is contained in the function `diffpost_p1_with_p1_linsys`.

The final strategy is a two-level error estimate that, at least in the case of piecewise linear approximations, utilises basis functions associated with edge midpoints of elements in the triangulation. Like with the previous strategy, we can derive both element-based and edge-based indicators from this estimation technique, which is performed via the function `diffpost_p1_with_p1_2level`. More details on this estimate can be found in [84] and others.

For quadratic (P_2-based) approximations to the solution of a given problem, the first of the above three techniques has typically utilised, except for the fact that the local hierarchical estimator uses piecewise quartic basis functions corresponding to midpoints of sub-elements. A two-level error estimation technique for use with quadratic approximations has also recently been implemented.

After the estimation process has completed, we save the data that has been collected about the current iteration. In particular, we know the global error estimate at this point in the adaptive loop, and we can therefore decide to terminate the algorithm if necessary. Following this, the marking and refinement stages proceed in a way that has been described in Sections 2.5 and 2.6. In the T-IFISS software, the maximum strategy and equilibration (or Dörfler) strategy have both been implemented. The function `marking_strategy_fa` contains the marking strategies and assists us in determining what our marked set $\mathcal{M}_k$ will be. The refinement step proceeds via the function `mesh_ref`, which uses the longest edge bisection procedure.

If the global estimated error is not below the required error tolerance, we return to the start of the adaptive loop. If we have achieved the required tolerance, then after some post-processing, the final data associated with the problem is displayed. This data includes a visualisation of the solution, and the final estimated error across the domain. We can also see the a plot of the convergence rate, the final mesh, and other data concerning the solution (for example, time elapsed or the final estimated energy of the error). A plot of effectivity indices as calculated using definition (2.26) can also be obtained.

2.8 Goal-oriented error estimation

We recall the weak formulation that we described using problem (2.11). Specifically, we know that the solution that $u \in V$ satisfies

$$B(u, v) = F(v), \quad \forall\, v \in V. \tag{2.60}$$

Of course, we can solve this problem to obtain the solution $u \in V$; however, in practice, this may not be our final objective. Instead, we may be interested in some other quantity that is related to the solution, but that is not the solution itself. Obviously, if this is the case, we can simply use the solution from our original problem in order to evaluate the quantity that we are interested in. That being said, if we have a priori knowledge of the quantity we want to evaluate, it is sensible to modify our approach in order to obtain an adaptive procedure that is specifically designed to estimate this quantity to a given accuracy as fast as possible. This is the purpose of *goal-oriented* error estimation.

We begin by letting $Q(u)$ describe a solution-dependent, bounded, linear *goal functional* or *quantity of interest*. We assume that, like with the right-hand side functional F, we have $Q : V \to \mathbb{R}$. In general, we have a wide degree of flexibility as to what we want to choose for our quantity of interest. The mean of a solution over a small region within the domain of the original problem, for example, is a suitable candidate, and can be written as

$$Q(u) := \frac{1}{|D_0|} \int_{D_0} u(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \tag{2.61}$$

for some sub-domain of interest, $D_0 \subset D$.

We can also use the goal-oriented framework for point-wise estimation; however, our method of doing so is not as obvious as in our previous example. Suppose that we want to estimate solution to our original problem at given point $\boldsymbol{x}_0 \in D$. It may be tempting to define a quantity of interest such as

$$Q(u) := u(\boldsymbol{x}_0). \tag{2.62}$$

The issue, as noted in [47], is that as a consequence of the Sobolev Embedding Theorem, the above linear functional is not, in general, bounded. Indeed, we recall from formulation (2.11) that $u \in H^1(D) = W^{1,2}(D)$; however, in 2 or more dimensions, this is insufficient to prove $u \in C_0(\bar{D})$, so the pointwise value $u(\boldsymbol{x}_0)$ may not be well-defined. (A more comprehensive

discussion of the relevant Sobolev inequalities may also be found in [64].) We therefore introduce the functional

$$Q(u) := \int_D u(\boldsymbol{x}) \cdot q_{\mathfrak{r}}(\boldsymbol{x} - \boldsymbol{x}_0)\, \mathrm{d}\boldsymbol{x}, \tag{2.63}$$

where we define

$$q_{\mathfrak{r}}(\boldsymbol{x}) := \begin{cases} C \exp\left(\dfrac{|\boldsymbol{x}|^2}{\mathfrak{r}^2} - 1\right)^{-1}, & \text{if } |\boldsymbol{x}| < \mathfrak{r}, \\ 0, & \text{if } |\boldsymbol{x}| \geqslant \mathfrak{r}. \end{cases} \tag{2.64}$$

We select the constant C in such a way that it satisfies

$$\int_D q_{\mathfrak{r}}(\boldsymbol{x} - \boldsymbol{x}_0)\, \mathrm{d}\boldsymbol{x} = 1 \tag{2.65}$$

for a selected radius, $\mathfrak{r} > 0$. In practice, there is some flexibility over our choice $\mathfrak{r}$; however, we will not vary this parameter during the experiments presented in later chapters of this work. The function $q_{\mathfrak{r}}$ is known as a mollifier, and may be viewed as a weighted average of u over a neighbourhood of the point of interest, $\boldsymbol{x}_0$. Note that if u is constant in the ball $\mathfrak{B}(\boldsymbol{x}_0, \mathfrak{r}) \subset D$, then we have $Q(u) = u(\boldsymbol{x}_0)$. Furthermore, we have the property

$$\lim_{\mathfrak{r} \to 0} Q(u) = u.$$

Importantly, the newly defined functional (2.63), unlike the quantity (2.62), is now a bounded linear functional on $H^1(D)$ and is therefore a more suitable candidate for our quantity of interest. There are other compactly supported L^2 approximations of the Dirac delta function that can be utilised in place of definition (2.64), such as the cone described by the function

$$q_{\mathfrak{r}}(\boldsymbol{x}) := \begin{cases} C\left(1 - \dfrac{|\boldsymbol{x}|}{\mathfrak{r}}\right), & \text{if } |\boldsymbol{x}| < \mathfrak{r}, \\ 0, & \text{if } |\boldsymbol{x}| \geqslant \mathfrak{r}. \end{cases} \tag{2.66}$$

Again, the constant, C, must be chosen so that condition (2.65) is satisfied. In the case of definition (2.64), we find that $C \approx \frac{2.1436}{\mathfrak{r}^2}$, whereas for definition (2.66), we instead have $C = \frac{3}{\pi \mathfrak{r}^2}$.

We note that the mollifier described by definition (2.64) is convenient to use due to its property of being infinitely smooth. While different approximations of the Dirac delta function can be utilised, this function is not more difficult to implement than other, simpler alternatives. This choice is also used in other works pertaining to goal-oriented adaptivity, such as [47], and will be utilised in later chapters of our work for the purposes of pointwise estimation.

In order to begin with our goal-oriented estimation strategy, we introduce a new problem formulation that involves this goal functional in our work:

Find $z \in V$ such that

$$B(v, z) = Q(v), \quad \forall\, v \in V. \tag{2.67}$$

We refer to this new problem as a *dual problem*. The original weak formulation described by equation (2.60) is often referred to as a *primal problem*. Of course, this new problem will admit a new solution, and this new solution is denoted $z \in V$.

Many of the same techniques that we discussed with respect to dealing with problem (2.60) can also be applied to problem (2.67). We immediately observe, for example, we can write discrete formulations associated with both of these problems. This yields Galerkin approximations satisfying

$$\begin{aligned} B(u_{\mathcal{T}}, v_{\mathcal{T}}) &= F(v_{\mathcal{T}}), \quad \forall\, v_{\mathcal{T}} \in V_{\mathcal{T}} \subset V, \\ B(v_{\mathcal{T}}, z_{\mathcal{T}}) &= Q(v_{\mathcal{T}}), \quad \forall\, v_{\mathcal{T}} \in V_{\mathcal{T}} \subset V, \end{aligned} \tag{2.68}$$

respectively.

At this point, we can use the formulations given in problems (2.67) and (2.68), as well as linearity of the bilinear form to proceed. Using these, as well as orthogonality, and finally, the Cauchy-Schwarz inequality:

$$\begin{aligned} |Q(u) - Q(u_{\mathcal{T}})| &= |B(u, z) - B(u_{\mathcal{T}}, z)| \\ &= |B(u - u_{\mathcal{T}}, z)| \\ &= |B(u - u_{\mathcal{T}}, z - z_{\mathcal{T}})| \\ &\leqslant |\!|\!| u - u_{\mathcal{T}} |\!|\!| \, |\!|\!| z - z_{\mathcal{T}} |\!|\!|. \end{aligned} \tag{2.69}$$

Hence, if we can control the error for the primal and dual problems with indicators

$$|\!|\!| u - u_{\mathcal{T}} |\!|\!| < \mu, \quad |\!|\!| z - z_{\mathcal{T}} |\!|\!| < \eta,$$

then the error in the approximation of our quantity of interest $g(u)$ can be controlled by the product $\mu\eta$. The three estimators described in Section 2.4 are amongst the techniques that may be utilised in this framework, with the estimation for the dual problem being done in the same way as for the primal.

With the solution and estimation steps of our procedure complete, we have a global estimate for the error in the goal functional. However, if we wish to proceed with an adaptive algorithm, we still need a marked set that represents elements or edges that will be selected for refinement. In our case, we now have two different marked sets – one representing our primal problem,

and the other representing our dual. Suppose that our marked sets for the primal and dual problems are $\mathcal{M}_u$ and $\mathcal{M}_z$, respectively (for brevity, we omit the references to the number of iterations that we saw in Section 2.5). We require a rule in order to amalgamate these sets to obtain a single marked set $\mathcal{M}$. To this end, there are a few different sensible choices that we can make.

The first choice of combining rule is a fairly obvious choice: we simply define $\mathcal{M} := \mathcal{M}_u \cup \mathcal{M}_z$. This simple 'union' rule is intuitive, although it has the similarly intuitive disadvantage that there is the potential for over-refinement at each iteration in the adaptive algorithm.

Another potential choice of rule is the 'aggregation' rule defined in [85]. This rule is similar to the union rule above, although it truncates whichever of the two original marked sets is larger in order to mitigate the potential for unnecessary refinements. The procedure, in full, begins with us examining whichever of the two marked sets is smaller, defining:

$$\mathcal{M}_- = \begin{cases} \mathcal{M}_u, & |\mathcal{M}_u| \leqslant |\mathcal{M}_z| \\ \mathcal{M}_z, & |\mathcal{M}_u| > |\mathcal{M}_z|. \end{cases}$$

Let the remaining set be denoted as $\mathcal{M}_+$, and suppose that local error indicators are defined by η_E for each edge or element $E \in \mathcal{M}_u \cup \mathcal{M}_z$. We now describe a truncated set, $\mathcal{M}'_+$, that satisfies the following properties:

1. $\mathcal{M}'_+ \subset \mathcal{M}_+$,
2. $|\mathcal{M}'_+| = |\mathcal{M}_-|$,
3. $E \in \mathcal{M}'_+ \implies E_+ \in \mathcal{M}'_+, \quad \forall\, E, E_+ \in \mathcal{M}_+$ s.t. $\eta_{E_+} > \eta_E$.

We then define $\mathcal{M} := \mathcal{M}_- \cup \mathcal{M}'_+$.

A third possible choice would be a 'minimum cardinality' rule, suggested in [50], which reduces the size of the marked set even further than the aggregation rule. In this instance, we simply define $\mathcal{M} := \mathcal{M}_-$; that is, we take whichever of the two sets is smaller to be the marked set. While this requires disregarding either the primal or the dual problem over a single iteration, over many iterations of an adaptive algorithm, both the primal and dual problems will eventually be taken into account as required. It is expected that this may take more

iterations to accomplish, although the conservative refinement could lead to fewer degrees of freedom being required to attain a certain tolerance in some cases. After obtaining the final marked set $\mathcal{M}$, we can proceed to refine the elements within this set as needed, and our adaptive algorithm proceeds as before.

2.9 Dual-weighted residuals

An alternative to the goal-oriented approach based on the hierarchical products of estimates is a residual-based approach, as seen in [72, 86] and others. We recall the primal and dual formulations,

$$B(u, v) = F(v), \quad \forall\, v \in V,$$
$$B(v, z) = Q(v), \quad \forall\, v \in V,$$

and their respective discretisations,

$$B(u_{\mathcal{T}}, v_{\mathcal{T}}) = F(v_{\mathcal{T}}), \quad \forall\, v_{\mathcal{T}} \in V_{\mathcal{T}} \subseteq V,$$
$$B(v_{\mathcal{T}}, z_{\mathcal{T}}) = Q(v_{\mathcal{T}}), \quad \forall\, v_{\mathcal{T}} \in V_{\mathcal{T}} \subseteq V.$$

We now observe that for arbitrary $\psi_{\mathcal{T}} \in V_{\mathcal{T}}$, we can use orthogonality to obtain

$$\begin{aligned} Q(u - u_{\mathcal{T}}) &= B(u - u_{\mathcal{T}}, z) \\ &= B(u - u_{\mathcal{T}}, z - \psi_{\mathcal{T}}) \\ &= F(z - v_{\mathcal{T}}) - B(u_{\mathcal{T}}, z - \psi_{\mathcal{T}}) \\ &= : \mathfrak{R}(u_{\mathcal{T}})(z - \psi_{\mathcal{T}}). \end{aligned} \tag{2.70}$$

The residual $\mathfrak{R}\left[u_{\mathcal{T}}\right](\cdot)$ is a functional V, and can be estimated using a standard residual based approach (i.e., integrating by parts element-wise, in a similar manner to what is shown in Section 2.4.1). After doing this, we obtain:

$$\begin{aligned} Q(u - u_{\mathcal{T}}) &= \mathfrak{R}\left[u_{\mathcal{T}}\right](z - \psi_{\mathcal{T}}) \\ &= \sum_{K \in \mathcal{T}} \left(\int_K \left(f + \nabla \cdot (a \nabla u_{\mathcal{T}})\right)(z - \psi_{\mathcal{T}}) \, \mathrm{d}\boldsymbol{x} - \int_{\partial K \setminus \partial D} a \frac{\partial u_{\mathcal{T}}}{\partial \boldsymbol{n}} (z - \psi_{\mathcal{T}}) \, \mathrm{d}\boldsymbol{x} \right) \\ &= \sum_{K \in \mathcal{T}} \left(\int_K \left(f + \nabla \cdot (a \nabla u_{\mathcal{T}})\right)(z - \psi_{\mathcal{T}}) \, \mathrm{d}\boldsymbol{x} + \int_{\partial K \setminus \partial D} \frac{a}{2} \left[\!\!\left[\frac{\partial u_{\mathcal{T}}}{\partial \boldsymbol{n}} \right]\!\!\right] (z - \psi_{\mathcal{T}}) \, \mathrm{d}\boldsymbol{x} \right), \end{aligned} \tag{2.71}$$

where the jumps $[\![\cdot]\!]$ are defined as before. By defining appropriate element and jump residuals as we have done previously, we have:

$$r_K := f + \nabla \cdot (a \nabla u_{\mathcal{T}}),$$
$$j_K := \frac{a}{2} \left[\!\!\left[\frac{\partial u_{\mathcal{T}}}{\partial \boldsymbol{n}} \right]\!\!\right].$$

We can then use the Cauchy-Schwarz inequality to obtain

$$|Q(u - u_{\mathcal{T}})| \leqslant \sum_{K \in \mathcal{T}} \rho_K \omega_K,$$

where ρ_K are localised residuals, and ω_K are associated weights, which are given by:

$$\rho_K := \left(\|r\|_K^2 + h_K^{-1} \|j\|_{\partial K}^2 \right)^{1/2},$$
$$\omega_K := \left(\|z - \psi_{\mathcal{T}}\|_K^2 + h_K \|z - \psi_{\mathcal{T}}\|_{\partial K}^2 \right)^{1/2},$$

respectively, where the per-element diameters h_K represent scaling factors for the edge contributions.

There are two things for us to consider in the above calculation. Firstly, we do not know the true solution $z \in V$, and we must therefore find a way of approximating this. Secondly, as the function $\psi_{\mathcal{T}} \in V_{\mathcal{T}}$ is arbitrary, we must find a sensible choice for this function ourselves.

With respect to approximating $z \in V$, we cannot, for example, choose $z \approx z_{\mathcal{T}} \in V_{\mathcal{T}}$ due to orthogonality. Replacing z by $z_{\mathcal{T}}$ in equality chain (2.70) leads to the approximation of the error in the quantity of interest by the residual

$$\begin{aligned} \mathfrak{R}\left[u_{\mathcal{T}}\right](z_{\mathcal{T}} - \psi_{\mathcal{T}}) &= B(u - u_{\mathcal{T}}, z_{\mathcal{T}} - \psi_{\mathcal{T}}) \\ &= F(z_{\mathcal{T}} - \psi_{\mathcal{T}}) - F(z_{\mathcal{T}} - \psi_{\mathcal{T}}) = 0, \end{aligned} \tag{2.72}$$

rendering the approximation useless. Indeed, we cannot, in general, approximate z by any function in $V_{\mathcal{T}}$, and we must therefore consider some form of higher-order approximation. This can typically be a higher-order polynomial (for example, for linear finite elements, we need at least quadratic-based approximations), or a higher resolution mesh (that is, we can choose $z \approx \widehat{z}_{\mathcal{T}} \in \widehat{V}_{\mathcal{T}}$, recalling our work from Section 2.4.3).

Recalling our discussion of error estimation in the non-goal-oriented setting, we recognise that these approximations may be undesirable due to the high computational expense associated with them, although this additional cost can often be avoided. In the case where a refinement

of the spatial mesh is chosen for the purposes of computing an approximation to the dual solution, we observe that there are no restrictions on the choice of arbitrary function $\psi \in V_{\mathcal{T}}$. Hence, by making the obvious choice $\psi := z_{\mathcal{T}} \in V_{\mathcal{T}}$, we obtain weights, ω_K, that are in terms of differences $\widehat{z}_{\mathcal{T}} - z_{\mathcal{T}}$. From here, we can apply similar ideas to Sections 2.4.2 and 2.4.3 in order to generate more easily computable estimates and localised error indicators.

If higher-order polynomials are utilised for the approximation of $z \in V$ instead, there are also options for reducing computational expense. These include calculating a higher-order polynomial approximation on a coarser grid, $\check{\mathcal{T}}$, in comparison to the original triangulation, $\mathcal{T}$. For example, if our original dual solution approximation is piecewise linear, we can compute the residual using the function space $\mathcal{S}_0^2(\check{\mathcal{T}})$, and then perform a defect correction procedure, as suggested in [86].

With the above noted, we also briefly note the potential exploitation of symmetry while using this method. In particular, we have, for example:

$$\begin{aligned} Q(u - u_{\mathcal{T}}) &= B(u - u_{\mathcal{T}}, z) \\ &= B(u - u_{\mathcal{T}}, z - v_{\mathcal{T}}) \\ &= B(u, z - v_{\mathcal{T}}) \\ &= F(z - v_{\mathcal{T}}). \end{aligned} \tag{2.73}$$

We will discuss the implications of this style of approach, and its role in establishing results involving primal weights, in Chapter 5.

Finally, we describe the effects of this method on our adaptive algorithm. In practice, we want to maintain the sharpness of global error estimates if possible. We may therefore want to place additional consideration on how we proceed after calculation (2.71), and in particular, the difference between our approach at the global level and at the local level. During the marking stage, we do not, in general, have the same issues as those encountered with the norm-product based approach described in Section 2.8. In particular, we note that there are only one set of indicators, and as a result of this, only one marked set $\mathcal{M}$. We therefore do not require any additional procedure to combine marked sets with this method. The remainder of the adaptive algorithm proceeds as we have described previously.

Chapter 3

Parametric PDEs and Stochastic Collocation FEM

Previously, we restricted our attention to the deterministic setting in order to describe the fundamental ingredients of the finite element algorithms that we will be utilising throughout this work. As we alluded to in our introduction, however, contemporary mathematical descriptions of various phenomena often now attempt to model uncertainty in some way. There are multiple reasons for this becoming a topic of greater interest in recent years. On the theoretical side, a greater understanding of the surrounding subject matter naturally leads us to use more comprehensive models that incorporate intricacies associated with underlying phenomena that we are attempting to model. In addition, with improvements to computational power that have occurred in recent decades, our ability to perform complex calculations has also increased. As a result of this, practical impediments to solving problems that incorporate uncertainty have also been significantly reduced.

In order to introduce some form of uncertainty into our models, we are required to formulate our problem in terms of random variables defined on an appropriate probability space. We begin this chapter by recalling some of the required probabilistic concepts before introducing the formulation of our elliptic model problem. Different methods of solving this problem are then briefly discussed, before we proceed to set up the stochastic collocation FEM framework that will be of primary interest for the remainder of this thesis.

3.1 Parametric problem formulation

For the purposes of our work, we will extend problem (2.10) into a settings that admits parameter-dependent inputs. As before, we will assume $D \subset \mathbb{R}^2$ to be a connected, bounded, and Lipschitz domain, with boundary denoted ∂D. We now also introduce a random (or uncertain) parameter domain, which we define by a finite-dimension hypercube, $\Gamma = \prod_{m=1}^{M}[-1, 1]$.

Following from the standard theory as set out in [87] and others, we denote the probability space for our problem by $(\Gamma, \mathcal{B}(\Gamma), \rho)$. Here, $\rho : \Gamma \to [0, 1]$ represents a probability measure which is absolutely continuous with respect to the Lebesgue measure. We assume that $\rho(\boldsymbol{y}) = \prod_{m=1}^{M} \rho_m(y_m)$, where each ρ_m constitutes a Borel probability measure on Γ_m. Each of the random parameters, y_m, is an image of an independent random variable with cumulative distribution function $\rho_m(y_m)$ and probability density function $\varrho_m(y_m) = \mathrm{d}\rho_m(y_m)/\,\mathrm{d}\boldsymbol{y}_m$. The corresponding joint probability density function is denoted by $\varrho(\boldsymbol{y}) = \mathrm{d}\rho(\boldsymbol{y})/\,\mathrm{d}\boldsymbol{y}$.

The event space, $\mathcal{B}(\Gamma)$, is the Borel σ-algebra for our sample space, Γ; that is, $\mathcal{B}(\Gamma)$ is the collection of all sets that can be formed on Γ, such that $\mathcal{B}(\Gamma)$ remains closed under complements, countable unions and countable intersections. Practically speaking, this set represents the set of all possible events in our sample space, Γ, that we can assign a probability to.

We further define a (possibly non-affine) parameter-dependent diffusion coefficient $a(x, \boldsymbol{y})$ which is bounded away from zero and infinity. Specifically, we assume that this coefficient satisfies the condition that $\exists\, a_\flat, a_\sharp > 0$ such that for almost all $\boldsymbol{y} \in \Gamma$,

$$0 < a_\flat \leqslant \operatorname*{ess\,inf}_{x \in D} a(x, \boldsymbol{y}) \leqslant \operatorname*{ess\,sup}_{x \in D} a(x, \boldsymbol{y}) \leqslant a_\sharp < \infty, \tag{3.1}$$

Here, we denote $x := (x_1, x_2) \in D$ by convention (that is, we omit the reference to x being a vector) to emphasise the dependence of variables on different random parameter components. We will consider a parametric model problem which defines $u : D \times \Gamma \to \mathbb{R}$ as the unique solution to:

$$\begin{aligned} -\nabla \cdot (a(x, \boldsymbol{y}))\nabla u(x, \boldsymbol{y}) = f(x), \quad x \in D, \quad \boldsymbol{y} \in \Gamma, \\ u(x, \boldsymbol{y}) = 0, \quad x \in \partial D, \quad \boldsymbol{y} \in \Gamma. \end{aligned} \tag{3.2}$$

where $f \in H^{-1}(D)$ represents a scalar 'source' or 'forcing' term. This quantity will remain independent of random parameters as in the previous chapter (although problems involving

parameter-dependent source terms can be considered, as in [34, 88], amongst others). We also note that in this formulation, that $\nabla := \nabla_x$ (that is, that the differentiation associated with this operator is with respect to the spatial parameters only).

In order to turn this model problem into a weak formulation, we may pursue a similar approach to the deterministic case; however, problem (3.2) now involves a parameter-dependent domain that we must contend with. This naturally leads to the consideration of function spaces known as *Bochner spaces*.

In a similar manner to [89], we define the Bochner space $\mathcal{V} := L^p_\rho(\Gamma; V)$ for $1 \leq p < \infty$ by a set of equivalence classes of functions which satisfy:

$$\|u(x, \boldsymbol{y})\|_{L^p_\rho(\Gamma;V)} := \left(\int_\Gamma \|u(x, \boldsymbol{y})\|_V^p \, \mathrm{d}\rho(\boldsymbol{y}) \right)^{1/p} < \infty. \tag{3.3}$$

The Bochner norm defined above may also be denoted $\|\cdot\|_{\mathcal{V}}$, or by $\|\cdot\|_{\Gamma;V}$, with the latter convention carrying over to Bochner-type norms over spaces other than $\mathcal{V}$. We may also establish a 'sampled' weak formulation corresponding to problem (3.2) via integration over the spatial domain. By viewing the solution as a map $u : \Gamma \to H^1_0(D) =: V$, we have:

Find $u_{\bar{\boldsymbol{y}}}(x) := u(x, \bar{\boldsymbol{y}}) \in V$ *such that*

$$\int_D a(x, \bar{\boldsymbol{y}}) \nabla u_{\bar{\boldsymbol{y}}}(x) \cdot \nabla v(x) \, \mathrm{d}x = \int_D f(x) v(x) \, \mathrm{d}x, \quad \forall v \in V, \quad \rho\text{-a.e. in } \Gamma. \tag{3.4}$$

We can write this in the compact form

$$B_{\bar{\boldsymbol{y}}}(u_{\bar{\boldsymbol{y}}}, v) = F(v), \quad \forall v \in V, \quad \rho\text{-a.e. in } \Gamma \tag{3.5}$$

where:

$$\begin{aligned} B_{\bar{\boldsymbol{y}}}(u_{\bar{\boldsymbol{y}}}, v) &:= \int_D a(x, \bar{\boldsymbol{y}}) \nabla u_{\bar{\boldsymbol{y}}}(x) \cdot \nabla v(x) \, \mathrm{d}x, \\ F(v) &:= \int_D f(x) v(x) \, \mathrm{d}x, \end{aligned} \tag{3.6}$$

with the $\bar{\boldsymbol{y}}$ subscript on the bilinear form being used to denote the sampling of the diffusion coefficient. We denote the energy norm induced by the bilinear form as $|||\cdot|||_{\bar{\boldsymbol{y}}}$. Alternatively, we may integrate over both the spatial and parameter domains to obtain a 'complete' weak formulation. By considering our solution as an element of the Bochner space $L^2_\rho(\Gamma; V) =: \mathcal{V}$, we obtain:

Find $u \in \mathcal{V} := L^2_\rho(\Gamma; V)$ *such that*

$$\mathcal{B}(u, v) = \mathcal{F}(v), \quad \forall v \in \mathcal{V} \tag{3.7}$$

with:

$$\begin{aligned} \mathcal{B}(u, v) &:= \int_\Gamma \int_D (a(x, \boldsymbol{y}) \nabla u(x, \boldsymbol{y})) \cdot \nabla v(x, \boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}), \quad \forall v \in \mathcal{V}, \\ \mathcal{F}(v) &:= \int_\Gamma \int_D f(x) v(x, \boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}), \quad \forall v \in \mathcal{V}. \end{aligned} \tag{3.8}$$

The bilinear form $\mathcal{B}$ induces an energy norm, defined by $|||v||| := (\mathcal{B}(v, v))^{1/2}$. In practice, the question of which of the above formulations may be useful to us will depend on our strategy for solving the original problem.

3.2 Numerical methods

How we proceed from the above problem formulation, in terms of the discretisation and other solution aspects, will in general, depend on the method we choose to solve our model problem. We will briefly discuss three possibilities, outlining the key ideas and potential advantages and disadvantages for each.

3.2.1 Monte Carlo methods

Named after the famous casino in the Principality of Monaco, the first method we will describe is the widely used 'Monte Carlo' method, as given in [90] and others. The Monte Carlo method is a form of stochastic sampling method that relies on utilising the practical implementations for the underlying deterministic problems. As a result of being a sampling based method, our natural starting point is the sampled weak formulation given in problem (3.4). It works by selecting N points, $\{\boldsymbol{y}_n\}_{n=1}^N$, in order to sample the parameter space Γ, and then producing a number of simulated outcomes, $u_N \in V_N$, where the approximation space V_N will depend on the nature of the underlying deterministic problem.

Each simulated solution can be obtained using a standard adaptive finite element method, and the results can be averaged to obtain the Monte Carlo approximation. Specifically, we will

obtain an approximation of the form

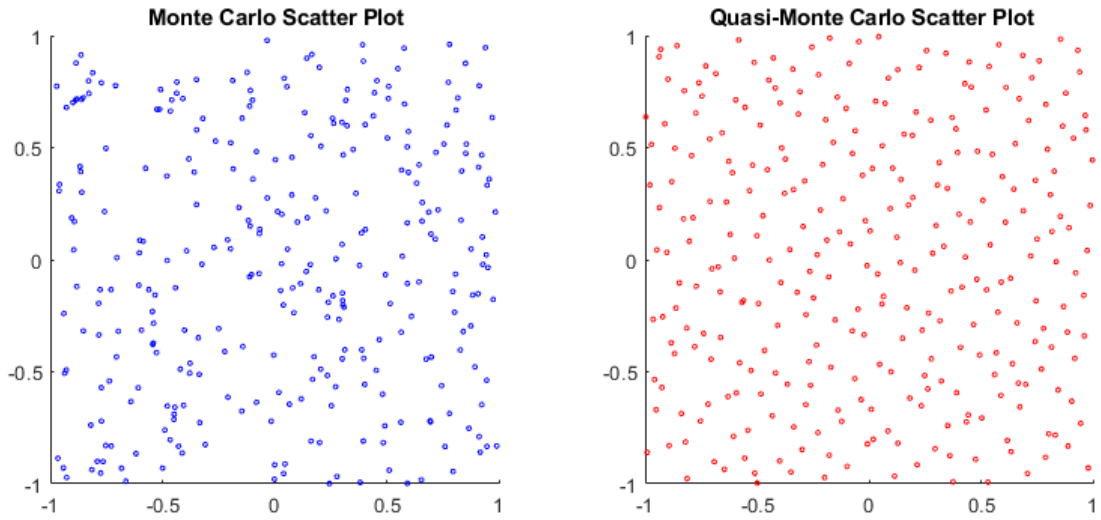
$$u_N\left(x, \{\boldsymbol{y}_n\}_{n=1}^N\right) = \frac{1}{N}\sum_{n=1}^{N} u_N(x, \boldsymbol{y}_n), \quad \forall\, x \in D. \tag{3.9}$$

An advantage of this method is how easy it is to adapt the underlying deterministic principles to this scenario. Notably, the underlying code needs little modification to be used for the required purposes (hence why Monte Carlo methods are often referred to as being *non-intrusive*). This advantage makes it both easy to implement, and to parallelise the computations. In addition, the performance of the Monte Carlo method itself is not dependent on the dimension of the problem being investigated, and there are no functional dependencies on parameters. This therefore allows a wide range of applications.

There are, however, some drawbacks to the method, one of which is the slow rate of convergence. As noted in [91], a typical Monte Carlo methods yield errors of order $\mathcal{O}(1/\sqrt{N})$, which is a consequence of the Central Limit Theorem. Hence, an exceptionally large number of sample points are required to be assured of accurate solutions. In addition, the information that is obtained from Monte Carlo methods is not always as useful as we would like to be. For example, the information that we do obtain is only in terms of statistical moments, such as an expectation – we do not obtain information about the explicit functional dependence on the random parameters. Lastly, if we want to estimate a solution given certain values for our parameters, and these parameter values were not sampled, then a certain amount of post-processing is required to obtain an estimate which takes time and produces another area for potential inaccuracies.

Different techniques have been introduced in order to mitigate some of the above issues. One such way of obtaining improvements over the standard approach is to modify how the sample points used for representation (3.9) are generated. In particular, we recall that the standard Monte Carlo method uses samples that are generated entirely randomly. This can be done in M-dimensions, for example, by taking a random sequence, and constructing vectors by taking M consecutive terms at a time. However, the randomness that we have by using this approach often leads to uneven distributions of sample points, as can be seen in Figure 3.1(a).

By contrast, 'Quasi-Monte Carlo' methods, as described in [92], use points from so-called *low-discrepancy* or *quasi-random* sequences. Samples generated from these sequences have the appearance of being random, but the sequences themselves are entirely deterministic, and



(a) (b)

Figure 3.1: A plot of $N = 300$ samples produced in the hypercube $\Gamma = [-1, 1]^M$, with $M = 2$ for (a) Monte Carlo, and (b) Quasi-Monte Carlo.

chosen to be far more evenly distributed than random sequences. Popular quasi-random sequences for generating samples include Sobol sequences [93] and Halton sequences [94], the latter of which has been used to generate the example shown in Figure 3.1(b). We note that unlike with random samples, the constructions of the quasi-random samples described here are explicitly dependent on the dimension of Γ. The improved distribution of samples means that the errors for Quasi-Monte Carlo simulations is typically of order $\mathcal{O}(\ln(N)^M/N)$.

The above convergence result immediately informs us of an important advantage, and a disadvantage associated with this method. In optimal situations, given the slow growth of the logarithm, we have an error of $\mathcal{O}(1/N^{1-\varepsilon})$ with $\varepsilon > 0$. However, the exponential dependence on the dimension means that for large M, this 'asymptotic' behaviour becomes insignificant and this method will struggle to outperform standard Monte Carlo procedures without an exceptionally large number of samples, N. That being said, if we want to refine an existing approximation by adding more samples, the use of the aforementioned quasi-random sequences allow us to maintain a favourable distribution across the parameter domain. This is something that is not achieved by, for example, a uniform lattice, which loses its even distribution on adding a small number of points, nor by use of random samples, which lack an even distribution to begin with.

Other variants and techniques for standard Monte Carlo methods worth mentioning include 'Multi-level Monte Carlo' methods (see, for example [27, 95]). These can be used to compute expectations by utilising random sampling combined with a series of successive discretisations of increasing precision. In particular, we can define a nested sequence of finite element spaces $V_0 \subset V_1 \subset ... \subset V_L \subset V$, with approximations $u_l \in V_l$ for each level of spatial discretisation, $l \in \{1, ..., L\}$. By defining $u_0 := 0$, we immediately obtain the telescoping sum:

$$u_L = \sum_{l=1}^{L} (u_l - u_{l-1}). \tag{3.10}$$

From the linearity of expectation, we have

$$\begin{aligned} \mathbb{E}[u_L] &= \mathbb{E}\left[\sum_{l=1}^{L} (u_l - u_{l-1})\right] \\ &= \sum_{l=1}^{L} \mathbb{E}[u_l - u_{l-1}]. \end{aligned} \tag{3.11}$$

Each of the terms in this sum can be estimated by a number of samples corresponding to each level. We can therefore estimate each of these expectations via

$$\mathbb{E}[u_l - u_{l-1}] \approx \frac{1}{N_l} \sum_{n=1}^{N_l} (u_l^{(l,n)} - u_{l-1}^{(l,n)}) \tag{3.12}$$

with the double superscript being used to denote the fact that independent samples are chosen for each of these levels of approximation. At each level, the number of samples used, N_l, is optimised in order to reduce the overall computational expense of the calculation in comparison to basic Monte Carlo methods. In particular, larger quantities of samples are utilised for coarser spatial meshes, while comparatively few samples are taken at the more precise levels where, for each sample, the resulting linear system is expensive to solve.

The above concept was originally used in [96] in the context of stochastic ordinary differential equations. Here, the number of samples at each level was chosen to minimise the variance of the multilevel estimator, which was then utilised to guarantee an upper bound on the mean square error on the estimator. In the context of finite element analysis, [95] balances the number of samples with the resolution of the spatial triangulation at each level in order to equilibrate the statistical and spatial discretisation errors, respectively. The result is that, for the multilevel Monte Carlo estimate given in that paper, the optimal number of samples at each level used in the computation of the estimate is proven to be inversely proportional to a power of the mesh width.

3.2.2 Stochastic Galerkin method

The stochastic Galerkin method takes a different approach to what we have just seen. In particular, rather than being a sampling based method, this is an *intrusive* method that utilises a discretisation associated with the complete weak formulation (3.7). In order to motivate the full discretisation, we note that a useful property of the Bochner space $L^2_\rho(\Gamma; H^1_0(D))$ is that it is isomorphic to the tensorised product $H^1_0(D) \otimes L^2_\rho(\Gamma)$.

For this method, the diffusion coefficient in the bilinear form associated with problem (3.7) is often taken to be linear, but can depend on an infinite number of parameters in the parameter domain (as in [54], for example). That is, we can define our parameter space by the infinite-dimension hypercube $\Gamma := [-1, 1]^\infty$, with the coefficient

$$a(x, \boldsymbol{y}) = a_0(x) + \sum_{m=1}^{\infty} y_m a_m(x). \tag{3.13}$$

By doing this, we are required to establish conditions that ensure the convergence of the above series. In particular, we modify condition (3.1) so that we assume that $\exists\, a_\flat, a_\sharp > 0$ such that

$$0 < a_\flat \leqslant a_0(x) \leqslant a_\sharp < \infty, \tag{3.14}$$

with

$$\frac{1}{a_\flat} \sum_{m=1}^{\infty} \|a_m\|_{L^\infty(D)} < 1. \tag{3.15}$$

Allowing for the possibility of an infinite number of random variables in representation (3.13) suggests a relaxing of the assumptions implied by the original problem formulation given in Section 3.1, despite the introduction of conditions (3.14) - (3.15). However, the use of an affine diffusion coefficient, and the assumption of a bounded parameter space, describe strong restrictions that preclude the use of, for example, log-normal distributions.

Using the weak formulation (3.7), we can imitate our original idea of constructing a discrete approximation space by defining this finite element space as $V_{\mathfrak{X}\mathfrak{P}} := \mathfrak{X} \otimes \mathfrak{P}$. In this case, $\mathfrak{X}$ corresponds to the spatial discretisation, and follows naturally from the discretisation we made when defining problem (2.15) in the purely deterministic setting. We also have an approximation space corresponding to a discretisation for the parameter space; this will be defined by $\mathfrak{P}$, which consists of a finite number of basis functions $\xi_j \in \mathfrak{P}$, each of which is a product of functions from an orthonormal basis of the space $L^2_\rho[-1, 1]$.

We can then obtain a fully discretised version of problem (3.7). We have the following formulation:

Find $u_{\mathfrak{X}\mathfrak{P}} \in V_{\mathfrak{X}\mathfrak{P}}$ *such that*

$$B(u_{\mathfrak{X}\mathfrak{P}}, v_{\mathfrak{X}\mathfrak{P}}) = F(v_{\mathfrak{X}\mathfrak{P}}) \quad \forall\, v_{\mathfrak{X}\mathfrak{P}} \in V_{\mathfrak{X}\mathfrak{P}} \subset V. \tag{3.16}$$

for which we will seek a solution of the form

$$u_{\mathfrak{X}\mathfrak{P}} = \sum_{i=1}^{N_{\mathfrak{X}}} \sum_{j=1}^{N_{\mathfrak{P}}} u_{ij} \varphi_i(x) \xi_j(\boldsymbol{y}). \tag{3.17}$$

The functions φ_i represent basis functions in $\mathfrak{X}$, and the functions ξ_j represent basis functions in $\mathfrak{P}$ with the total number of degrees of freedom in each case given by $N_{\mathfrak{X}} = |\mathfrak{X}|$ and $N_{\mathfrak{P}} = |\mathfrak{P}|$.

We note that the stochastic Galerkin approach allows us to obtain more quantitative and qualitative information about the solution to the original problem than Monte Carlo methods. In particular, the stochastic Galerkin method, by construction, reveals information about the solution's dependence on random parameters in a way that Monte Carlo methods do not. This allows us, for example, to evaluate quantities related to the solution which do not directly depend on statistical moments. Furthermore, if we want precise information about the solution for given parameter values, we can immediately evaluate the solution at those parameter values without incurring additional inaccuracies through post-processing.

Another advantage of the stochastic Galerkin framework is that, under certain assumptions with regards to the regularity of the problems we attempt to solve, we can see quicker convergence for this method than for Monte Carlo methods. On the other hand, due to the problem being reformulated as a full discretisation, legacy code for the deterministic case has to be redesigned for the purposes of this method.

Furthermore, it should be noted that because $\dim(V_{\mathfrak{X}\mathfrak{P}}) = \dim(\mathfrak{X}) \times \dim(\mathfrak{P})$, selecting a large number of random variables to represent the parameter domain Γ leads to this method becoming computationally expensive very quickly. Part of the reason for the computational expense associated with approximations via this method are the large sizes of coupled linear systems that need to be solved. The computational time associated with these linear systems can be reduced, for example, by use of carefully selected truncation pre-conditioners that solve the underlying systems more efficiently, as discussed in [97].

We also recall that we assumed a linear diffusion coefficient, as in representation (3.13). This is a somewhat restrictive condition; however, stochastic Galerkin FEM based approximations involving arbitrary diffusion coefficients are often very expensive to compute. That being said, some progress has been made for problems involving non-affine coefficients. For example, tensor decompositions have been successfully used in [98] to make problems involving log-normal diffusion coefficients tractable, while more general forms of the coefficient are considered in [99].

3.2.3 Stochastic Collocation method

The final method that we will review for the purposes of solving our original problem will be a stochastic collocation method, as previously described in [30, 29, 37]. Like the Monte Carlo method, this approach also proceeds along the lines of a spatial semi-discretisation, following from problem (3.4). However, we then use a grid of sample points to seek a solution $u_{\mathfrak{X}\mathfrak{P}} \in V_{\mathfrak{X}\mathfrak{P}} \subset \mathfrak{X} \otimes \mathfrak{P}$, that has explicit functional dependence on the random parameters.

In particular, let $\mathfrak{X}$ be a standard finite element approximation space, and let $\mathfrak{P} := \mathbb{P}(\Gamma)$, a polynomial-based subspace of $L^2_\rho(\Gamma)$. After a spatial semi-discretisation of problem (3.4), we are tasked with finding $u_\mathfrak{X} : \Gamma \to \mathfrak{X}$ such that

$$\int_D a(x, \boldsymbol{y}) \nabla u_\mathfrak{X}(x, \boldsymbol{y}) \cdot \nabla v_\mathfrak{X}(x) \ \mathrm{d}x = \int_D f(x) v_\mathfrak{X}(x) \ \mathrm{d}x, \quad \forall \, v_\mathfrak{X} \in \mathfrak{X} \tag{3.18}$$

almost everywhere in Γ. Note that unlike with the stochastic Galerkin method, we retain a truncated parameter space of the form $\Gamma = [-1, 1]^M$. We then consider a number of sample points (or collocation points) within the parameter space Γ, defined by the set $Y = \{\boldsymbol{y}_1, ..., \boldsymbol{y}_N\}$. This will enable us to construct an approximation of the form

$$u^{SC}(x, \boldsymbol{y}) := \sum_{\bar{\boldsymbol{y}} \in Y} u_{\bar{\boldsymbol{y}}} L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \tag{3.19}$$

with the $L_{\bar{\boldsymbol{y}}}$ representing elements of an appropriate set of polynomials, $\mathbb{P}(\Gamma) \subset L^2(\Gamma)$, that have the property that $u_\mathfrak{X}(\boldsymbol{y}_m) \in \mathfrak{X}$ solves problem (3.18) with $\boldsymbol{y} = \boldsymbol{y}_m$.

There are advantages associated with the stochastic collocation framework that distinguish it from both of the previous two methods. Firstly, this collocated-based approach is immediately amenable to the idea of obtaining full approximations to our original problem, with arbitrary

diffusion coefficients. The other methods we have described can also be used with non-affine diffusion coefficients; however, in the case of stochastic Galerkin methods, some novelty is typically required in order to make the resulting calculations tractable. Furthermore, with stochastic collocation FEM, we obtain an approximation that has explicit functional dependence on the random parameters involved in our problem, which is not the case with Monte Carlo methods.

Secondly, due to the fact that we are only required to solve decoupled problems that are deterministic in nature, we are able to heavily utilise deterministic solvers in a way that we could not for the stochastic Galerkin implementation. This can also be said of Monte Carlo methods; however, as noted in [37], the stochastic collocation method allows for the possibility of exploiting the regularity of the solution (with respect to random variables) to yield much faster convergence rates in certain cases.

However, there is some cause for concern with respect to our choice of samples within the parameter domain. In particular, we could choose a series of points on the interval $[-1, 1]$ and then tensorise this across a finite dimensional truncation of the parameter space Γ. The issue with this approach is the so-called *curse of dimensionality*. Simply taking a tensor grid in this manner leads to the number of sample points in the resulting grid increasing exponentially with the dimension, M, of the truncated parameter space. This then presents significant problems with respect to computational expense. More recently, alternative strategies for generating sample points are utilised, particularly those that incorporate a *sparse grid* structure, as introduced by [32]. In the sections that follow, we will describe this sparse grid, as well as the resulting interpolant that constitutes our approximation to the solution of problem (3.4).

3.3 Sparse grid interpolation

Having outlined three key approaches to solving our parametric PDE problem, we noted that in the case of the stochastic collocation method, we could exploit a sparse grid structure to construct an interpolant-based approximation. Of course, sparse grids have been utilised in various scenarios prior to their usage in the stochastic collocation FEM setting. For example, sparse tensor products have seen use in finite element methods for problems with multiple scales in [100], where simulation of the multiscale problem using a coarse finite element

approximation is used to avoid the requirement for analytical homogenisation. Nevertheless, in this instance, our objective will be to generate a stochastic collocation FEM approximation by constructing an interpolant which utilises the nodes associated with sparse grids. We proceed to outline the details of the construction here, along with some relevant concepts and results which will be important for the purposes of understanding both the sparse grid structure, and the resulting approximation.

3.3.1 The Smolyak sparse grid interpolant

We begin with some notation conventions for vectors in $\mathbb{R}^M$. Firstly, for two vectors, $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^M$, we denote:

$$\boldsymbol{u} \geqslant \boldsymbol{v} \iff u_k \geqslant v_k, \quad \forall k \in \{1, ..., M\},$$

and use the analogous convention for strict inequality. In addition, given $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w} \in \mathbb{N}^M$ we introduce the similar notation

$$\sum_{\boldsymbol{u}=\boldsymbol{v}}^{\boldsymbol{w}} := \sum_{u_1=v_1}^{w_1} \cdots \sum_{u_M=v_M}^{w_M}$$

with other operators and operations defined similarly. We also use the absolute value to denote the sum of components of a vector; that is, we set

$$|\boldsymbol{u}| = |u_1 + u_2 + ... + u_M|, \quad \forall\, u \in \mathbb{R}^M$$

Lastly, we say that a set, $I \subset \mathbb{R}^M$, is *monotonic*, or *downward-closed*, if

$$\boldsymbol{i} \in I \implies \boldsymbol{i} - \boldsymbol{e}_k \in I, \quad \forall\, k \in \{1, ..., M\} \text{ with } i_k > 1.$$

where $\boldsymbol{e}_k \in \mathbb{N}^M$ is the usual elementary basis vector corresponding to dimension $k \leqslant M$. Now, recall that our problem involves a parameter space described by the M-dimensional hyper-cube:

$$\Gamma^M := \bigotimes_{m=1}^{M} \Gamma^m = [-1, 1]^M.$$

In order to construct our interpolant in this parameter domain, we use *multi-index* vectors in $\mathbb{N}^M \subset \mathbb{R}^M$ to describe the data points that we use for the interpolation procedure. Define $\boldsymbol{i} = (i_1, ..., i_M) \in \mathbb{N}^M$ to be multi-indices. We note that i_k defines an index in each dimension, $k \in \{1, ..., M\}$.

The number of nodes, or collocation points, for each index i_k, is determined by pre-defined rule. Typically, this rule sets the number of nodes according to a strictly increasing function

$m_i = m(i) : \mathbb{N} \to \mathbb{N}$ which satisfies $m(0) = 0$. Using this rule, we note that for each direction, $k \in \{1, ..., M\}$, we can describe a set of points for interpolation using the set:

$$Y^{i_k} := \{y_1^{(i_k)}, ..., y_{m_{i_k}}^{(i_k)}\}.$$

This leads to a full tensor interpolation operator defined by:

$$(\mathcal{U}^{i_1} \otimes ... \otimes \mathcal{U}^{i_M})[u](\boldsymbol{y}) := \sum_{j_1=1}^{m_{i_1}} ... \sum_{j_M=1}^{m_{i_M}} u(y_{j_1}^{(i_1)}, ..., y_{j_M}^{(i_M)})(L_{j_1}^{(i_1)}(y_1) \times ... \times L_{j_M}^{(i_M)}(y_M)), \tag{3.20}$$

where $L_j^{(i)}$ is a Lagrange polynomial of degree $m_i - 1$ satisfying the usual Kronecker delta property $L_j^{(i)}(y_l^{(i)}) = \delta_{jl}$, $\forall\, j, l \in \{1, ...m_i\}$. Using our previously described conventions for collating indices, we can write the above in a more compact form:

$$\bigotimes_{k=1}^{M} \mathcal{U}^{i_k}[u](\boldsymbol{y}) = \sum_{\boldsymbol{j}=\boldsymbol{1}}^{m_{\boldsymbol{i}}} u(\boldsymbol{y}_{\boldsymbol{j}}^{(\boldsymbol{i})}) \left(\prod_{k=1}^{M} L_{j_k}^{i_k}(y_k) \right) \tag{3.21}$$

(In a similar fashion, $m_{\boldsymbol{i}}$ refers to a vector consisting of element-wise evaluations of the entries in $\boldsymbol{i} \in \mathbb{N}^M$.) It is clear that the above operator requires us to use the set of nodes $Y_M := \prod_{k=1}^{M} Y^{i_k}$ which leads to $|Y_M| = \prod_{k=1}^{M} m_{i_k}$ different function evaluations. It is clear that this set, and the corresponding number of function evaluations, will grow rapidly with the number of dimensions, M. Hence, due to the associated computational expense, we want to avoid calculations involving this set of nodes.

An alternative to the above approach is Smolyak sparse grid interpolation [32], which uses a linear combination of products, but considerably fewer points for interpolation. This is achieved by describing a framework of *hierarchical surplus* operators constructed from one-dimensional Lagrange basis functions. For a given index in a single dimension, $i \in \mathbb{N}$, we define:

$$\Delta^i = \mathcal{U}^i - \mathcal{U}^{i-1}, \quad \forall\, i \in \mathbb{N} \tag{3.22}$$

with

$$\mathcal{U}^i := \begin{cases} \sum_{j=1}^{m_i} u(y_j^{(i)}) L_j^{(i)}(y), & i \in \mathbb{N} \\ 0, & i = 0. \end{cases} \tag{3.23}$$

Then, with an index set $I \in \mathbb{N}^M$, containing indices $\boldsymbol{i} = (i_1, ..., i_M)$, we may define the operator $\Delta^{\boldsymbol{i}}$ by a tensor product of the one dimensional operators above. Doing this, we have:

$$\Delta^{\boldsymbol{i}} := \bigotimes_{k=1}^{M} \Delta^{i_k} = \bigotimes_{k=1}^{M} (\mathcal{U}^{i_k} - \mathcal{U}^{i_k - 1}).$$

The above operator has a number of properties, one of which we demonstrate with our next result:

Lemma 3.1. *The operator $\Delta^{\boldsymbol{i}}$ can be re-written as:*

$$\begin{aligned}\Delta^{\boldsymbol{i}} &:= \Delta^{i_1} \otimes ... \otimes \Delta^{i_M} \\ &= \sum_{\boldsymbol{\alpha} \in \{0,1\}^M} (-1)^{\boldsymbol{\alpha}} (\mathcal{U}^{i_1 - \alpha_1} \otimes ... \otimes \mathcal{U}^{i_M - \alpha_M}).\end{aligned} \tag{3.24}$$

Proof. This can be demonstrated by induction on the dimension, M. Of course, the case of one dimension is trivial, but beyond this, we will want to start expanding the surplus operators. For $M = 2$, it is clear that

$$\begin{aligned}\Delta^{\boldsymbol{i}} &= \Delta^{i_1} \otimes \Delta^{i_2} \\ &= (\mathcal{U}^{i_1} - \mathcal{U}^{i_1 - 1}) \otimes (\mathcal{U}^{i_2} - \mathcal{U}^{i_2 - 1}) \\ &= (\mathcal{U}^{i_1} \otimes \mathcal{U}^{i_2}) - (\mathcal{U}^{i_1} \otimes \mathcal{U}^{i_2 - 1}) - (\mathcal{U}^{i_1 - 1} \otimes \mathcal{U}^{i_2}) + (\mathcal{U}^{i_1 - 1} \otimes \mathcal{U}^{i_2 - 1}) \\ &= \sum_{\boldsymbol{\alpha} \in \{0,1\}^2} (-1)^{|\boldsymbol{\alpha}|} \, (\mathcal{U}^{i_1 - \alpha_1} \otimes \mathcal{U}^{i_2 - \alpha_2}).\end{aligned}$$

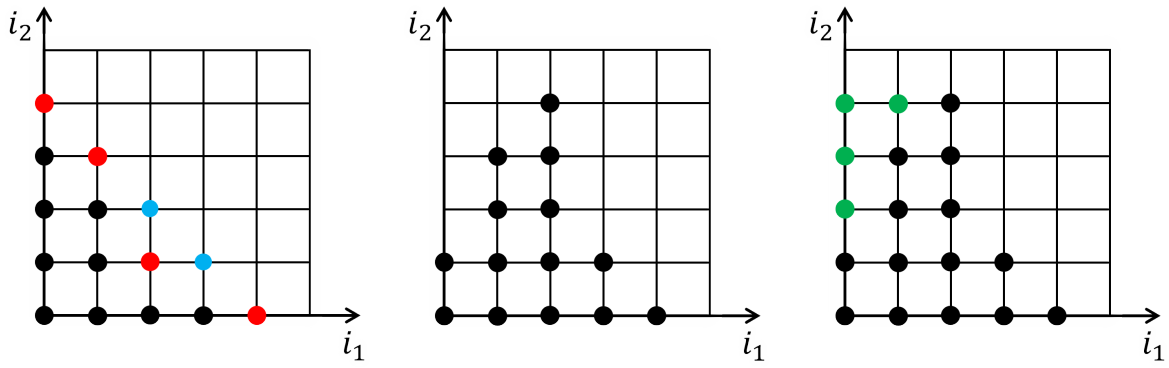
Under the assumption that equality (3.24) holds for M, our inductive step proceeds by calculating:

$$\begin{aligned}&\Delta^{i_1} \otimes ... \otimes \Delta^{i_{M+1}} \\ &= \big((\mathcal{U}^{i_1} - \mathcal{U}^{i_1 - 1}) \otimes ... \otimes (\mathcal{U}^{i_M} - \mathcal{U}^{i_M - 1})\big) \otimes (\mathcal{U}^{i_{M+1}} - \mathcal{U}^{i_{M+1} - 1}) \\ &= \sum_{\boldsymbol{\alpha} \in \{0,1\}^M} (-1)^{|\boldsymbol{\alpha}|} (\mathcal{U}^{i_1 - \alpha_1} \otimes ... \otimes \mathcal{U}^{i_M - \alpha_M}) \otimes (\mathcal{U}^{i_{M+1}} - \mathcal{U}^{i_{M+1} - 1}) \\ &= \sum_{\boldsymbol{\alpha} \in \{0,1\}^M} (-1)^{|\boldsymbol{\alpha}|} (\mathcal{U}^{i_1 - \alpha_1} \otimes ... \otimes \mathcal{U}^{i_M - \alpha_M} \otimes (-1)^0 \, \mathcal{U}^{i_{M+1}}) \\ &\qquad + \sum_{\boldsymbol{\alpha} \in \{0,1\}^M} (-1)^{|\boldsymbol{\alpha}|} (\mathcal{U}^{i_1 - \alpha_1} \otimes ... \otimes \mathcal{U}^{i_M - \alpha_M} \otimes (-1)^1 \, \mathcal{U}^{i_{M+1} - 1}) \\ &= \sum_{\boldsymbol{\alpha} \in \{0,1\}^{M+1}} (-1)^{|\boldsymbol{\alpha}|} (\mathcal{U}^{i_1 - \alpha_1} \otimes ... \otimes \mathcal{U}^{i_{M+1} - \alpha_{M+1}}).\end{aligned}$$

This concludes the proof. □

Using the above operator and the index set I, we define the Smolyak sparse grid interpolant by:

$$S_I[u](\boldsymbol{y}) := \sum_{\boldsymbol{i} \in I} \Delta^{\boldsymbol{i}}[u](\boldsymbol{y}). \tag{3.25}$$



(a) (b) (c)

Figure 3.2: Representations of index sets $I \subset \mathbb{N}^2$. In (a), we see I, $\mathcal{R}_I$ (red) and $\mathcal{M}_I$ (blue and red). The index set in (b) is non-monotonic. The index set in (c) represents the nodes that have to be added to the set in order to make it monotonic.

The set of collocation points associated with the above operator is given by:

$$Y := \bigcup_{\boldsymbol{i} \in I} (Y^{i_1} \times ... \times Y^{i_M}) \tag{3.26}$$

We note here that many sources, such as [101], define a sparse grid setup explicitly using the level of refinement associated with the index sets. This is a somewhat useful tool for proving core properties of the index sets and related quantities; however, this is not required for the purposes of understanding the functioning of the sparse grid. Furthermore, representing the theory in this way does not help us with the intricacies of adaptive refinement that we will consider later.

Another index set that we will introduce here, related to $\breve{I}$, is the *reduced margin* set. For a monotonic index set, I, we define its *margin*, $\mathcal{M}_I$, by

$$\mathcal{M}_I := \{\boldsymbol{i} \in \mathbb{N}^M \setminus I : \exists\, k \in \{1, ..., M\} \text{ s.t. } \boldsymbol{i} - \boldsymbol{e}_k \in I\}, \tag{3.27}$$

and the *reduced margin* $\mathcal{R}_I$ by:

$$\mathcal{R}_I := \{\boldsymbol{i} \in \mathbb{N}^M \setminus I : \boldsymbol{i} - \boldsymbol{e}_k \in I,\ \forall\, k \in \{1, ..., M\}\}. \tag{3.28}$$

We give an example of the above definitions in Figure 3.2.

The form of the sparse grid interpolant given with definition (3.25) is a compact representation of the operator. However, we would prefer this to be written in a more useful form that is

typical for interpolation operators, and that is easier to construct for practical computations. The theorem below addresses this subject.

Theorem 3.2. *If the multi-index set I is monotonic, then the Smolyak Sparse grid interpolant S_I, as defined in 3.25, may be written as*

$$S_I[u](\boldsymbol{y}) := \sum_{\boldsymbol{m}\in\breve{I}} u(\boldsymbol{y_m}) L_{\boldsymbol{m}}(\boldsymbol{y}), \tag{3.29}$$

where

$$\begin{aligned} L_{\boldsymbol{m}}(\boldsymbol{y}) &:= \sum_{\substack{\boldsymbol{i}\in I \\ m(\boldsymbol{i})\geqslant \boldsymbol{m}}} c_{\boldsymbol{i}} \left(L_{m_1}^{(i_1)}(y_1) \times ... \times L_{m_M}^{(i_M)}(y_M) \right) \\ &= \sum_{\substack{\boldsymbol{i}\in I \\ m(\boldsymbol{i})\geqslant \boldsymbol{m}}} c_{\boldsymbol{i}} \left(\prod_{k=1}^{M} L_{m_k}^{(i_k)}(y_k) \right), \end{aligned}$$

with

$$\breve{I} := \{\boldsymbol{m} \in \mathbb{N}^M : \exists\, \boldsymbol{i} \in I \quad s.t. \quad \boldsymbol{m} \leqslant m(\boldsymbol{i})\},$$

and

$$c_{\boldsymbol{i}} := \sum_{\substack{\boldsymbol{\alpha}\in\{0,1\}^M \\ \boldsymbol{\alpha}+\boldsymbol{i}\in I}} (-1)^{|\boldsymbol{\alpha}|}.$$

Proof. By definition of our hierarchical surplus operator, we have

$$S_I[u](\boldsymbol{y}) := \sum_{\boldsymbol{i}\in I} \Delta^{\boldsymbol{i}}[u](\boldsymbol{y}).$$

Having established an alternative representation for the expansion of the surplus operator in Lemma 3.1, we may write our sparse grid interpolant as:

$$S_I[u](\boldsymbol{y}) := \sum_{\boldsymbol{i}\in I} \sum_{\boldsymbol{\alpha}\in\{0,1\}^M} (-1)^{|\boldsymbol{\alpha}|} \left(\mathcal{U}^{i_1-\alpha_1} \otimes ... \otimes \mathcal{U}^{i_M-\alpha_M} \right)[u](\boldsymbol{y}).$$

We now seek to re-write our double summation. Firstly, we define $\boldsymbol{j} := \boldsymbol{i} - \boldsymbol{\alpha}$ with $\boldsymbol{i} \in I$ and $\boldsymbol{\alpha} \in \{0,1\}^M$. It is at this point that we use the fact that the multi-index set I is monotonic. In particular, we observe that, provided that $\boldsymbol{i} - \boldsymbol{\alpha} \geqslant \mathbf{1}$, we have $\boldsymbol{i} \in I \implies \boldsymbol{j} = \boldsymbol{i} - \boldsymbol{\alpha} \in I$. This yields the result:

$$S_I[u](\boldsymbol{y}) := \sum_{\boldsymbol{j}\in I} \sum_{\substack{\boldsymbol{\alpha}\in\{0,1\}^M \\ \boldsymbol{\alpha}+\boldsymbol{j}\in I}} (-1)^{|\boldsymbol{\alpha}|} \left(\mathcal{U}^{j_1} \otimes ... \otimes \mathcal{U}^{j_M} \right)[u](\boldsymbol{y}).$$

Using the form of the full tensor interpolation operator given in definition (3.21), the sparse grid interpolant is:

$$\begin{aligned} S_I[u](\boldsymbol{y}) &:= \sum_{\boldsymbol{j}\in I} \sum_{\substack{\boldsymbol{\alpha}\in\{0,1\}^M \\ \boldsymbol{\alpha}+\boldsymbol{j}\in I}} (-1)^{|\boldsymbol{\alpha}|} \sum_{\boldsymbol{m}=\boldsymbol{1}}^{m_{\boldsymbol{j}}} u(\boldsymbol{y}_{\boldsymbol{m}}^{(\boldsymbol{j})}) \left(\prod_{k=1}^{M} L_{m_k}^{(j_k)}(y_k) \right) \\ &= \sum_{\boldsymbol{j}\in I} \sum_{\boldsymbol{m}=\boldsymbol{1}}^{m_{\boldsymbol{j}}} \sum_{\substack{\boldsymbol{\alpha}\in\{0,1\}^M \\ \boldsymbol{\alpha}+\boldsymbol{j}\in I}} (-1)^{|\boldsymbol{\alpha}|} u(\boldsymbol{y}_{\boldsymbol{m}}^{(\boldsymbol{j})}) \left(\prod_{k=1}^{M} L_{m_k}^{(j_k)}(y_k) \right) \\ &= \sum_{\boldsymbol{j}\in I} \sum_{\boldsymbol{m}=\boldsymbol{1}}^{m_{\boldsymbol{j}}} c_{\boldsymbol{j}} u(\boldsymbol{y}_{\boldsymbol{m}}^{(\boldsymbol{j})}) \left(\prod_{k=1}^{M} L_{m_k}^{(j_k)}(y_k) \right), \end{aligned}$$

with

$$c_{\boldsymbol{j}} := \sum_{\substack{\boldsymbol{\alpha}\in\{0,1\}^M \\ \boldsymbol{\alpha}+\boldsymbol{j}\in I}} (-1)^{|\boldsymbol{\alpha}|}.$$

Finally, we swap the order of the sums above:

$$S_I[u](\boldsymbol{y}) := \sum_{\boldsymbol{m}\in\breve{I}} \sum_{\substack{\boldsymbol{i}\in I \\ m(\boldsymbol{i})\geqslant\boldsymbol{m}}} c_{\boldsymbol{i}} \left(\prod_{k=1}^{M} L_{m_k}^{(i_k)}(y_k) \right) u(\boldsymbol{y}_{\boldsymbol{m}}^{(\boldsymbol{i})}),$$

with

$$\breve{I} := \{\boldsymbol{m} \in \mathbb{N}^M : \exists\, \boldsymbol{i} \in I \quad \text{s.t.} \quad \boldsymbol{m} \leqslant m(\boldsymbol{i})\}.$$

The result follows. □

One corollary of the above theorem is that the functions $L_{\boldsymbol{m}} \in \mathbb{P}(\Gamma) \subset L^2(\Gamma)$ are not only linear combinations of Lagrange basis functions, but also Lagrange basis functions themselves, in the sense that they satisfy the Kronecker delta property across collocation points. In particular, we note that the multi-indices $\boldsymbol{i} \in I$ can correspond to multiple collocation points $\bar{\boldsymbol{y}} \in Y$; however, the multi-index set $\breve{I}$ is isomorphic to the collocation point set Y (as we will see when discussing different types of collocation point rule later in this section). The issue of exactness is discussed in [101] and others, while we have

$$S_I[L_{\boldsymbol{n}}](\boldsymbol{y}) = \sum_{\boldsymbol{m}\in\breve{I}} L_{\boldsymbol{n}}(\boldsymbol{y}_{\boldsymbol{m}}) L_{\boldsymbol{m}}(\boldsymbol{y}) \tag{3.30}$$

and the trivial result

$$L_{\boldsymbol{n}}(\boldsymbol{y}) = \sum_{\boldsymbol{m}\in\breve{I}} \delta_{\boldsymbol{m}\boldsymbol{n}} L_{\boldsymbol{m}}(\boldsymbol{y}) \tag{3.31}$$

from which the required result follows.

Another consequence is the use of the interpolant for the purposes of quadrature formulae. Define $\mathfrak{Q}^i$ as quadrature formulae derived from integrating the 1D Lagrange interpolant – that is,

$$\mathfrak{Q}^i[u](\boldsymbol{y}) := \begin{cases} \sum_{j=1}^{m_i} u(y_j^{(i)}) \int_{-1}^{1} L_j^{(i)}(y)\, \mathrm{d}\rho_1(y), & i \in \mathbb{N} \\ 0, & i = 0. \end{cases} \tag{3.32}$$

The Smolyak quadrature formula can be given analogously to formula (3.25), so that

$$\mathfrak{Q}[u](\boldsymbol{y}) := \sum_{\boldsymbol{i} \in I} \Delta_{\mathfrak{Q}}^{\boldsymbol{i}}[u](\boldsymbol{y}) \tag{3.33}$$

where, in a similar manner to the interpolant, we define the quadrature surplus operator,

$$\Delta_{\mathfrak{Q}}^i = \mathfrak{Q}^i - \mathfrak{Q}^{i-1}, \quad \forall\, i \in \mathbb{N}. \tag{3.34}$$

An alternative to the above would simply be to integrate representation (3.29) to obtain the equivalent formula

$$\mathfrak{Q}[u](\boldsymbol{y}) := \sum_{\boldsymbol{m} \in \breve{I}} u(\boldsymbol{y}_{\boldsymbol{m}}) \int_{\Gamma} L_{\boldsymbol{m}}(\boldsymbol{y})\, \mathrm{d}\rho(\boldsymbol{y}). \tag{3.35}$$

Quadrature-like calculations are useful for calculating moments of the stochastic collocation FEM approximation. Furthermore, we can use quadrature formulae to approximate other difficult to compute quantities, which we will see more of in Chapters 4 and 5.

3.3.2 Collocation point indexing rules

Of course, we need to relate the above multi-indices, $\boldsymbol{i} \in I$ and $\boldsymbol{m} \in \breve{I}$, to the locations of collocation points in our M-dimensional hypercube. We recall that in order to do this, we must introduce a pre-defined rule that defines a strictly increasing function $m_i = m(i) : \mathbb{N} \to \mathbb{N}$. Suppose that $i \in \mathbb{N}$ is one of the indices contained in one the multi-indices from the set I.

One possible choice is the rule first described by Leja in [102]. For this rule, we set

$$m_i := i \tag{3.36}$$

and then, determine the values of nodes using the expression

$$\arg\max_{y \in [-1,1]} \prod_{k=1}^{j-1} \left| y - y_k^{(i)} \right|. \tag{3.37}$$

Sequences of Leja points described using this expression are not, in general, unique. The maximiser in the above definition does not yield unique values, and the initial starting value, $y_1^{(1)}$, can be set arbitrarily on the interval $[-1, 1]$. Hence, we specify an initial value and take the minimum possible value across all maximisers in order to create a unique sequence. This leads to the recursive definition

$$y_j^{(i)} := \begin{cases} 1, & j = 1 \\ \min\left\{ \underset{y\in[-1,1]}{\arg\max} \prod_{k=1}^{j-1} \left| y - y_k^{(i)} \right| \right\}, & j \in \{2, ..., m_i\}. \end{cases} \tag{3.38}$$

With quantities uniquely defined by the above definition, we generate a sequence of nested points with values (to 4 decimal places where appropriate):

$$\begin{aligned} y_1^{(1)} &= 1 \\ y_j^{(2)} &= \{1, -1\} \\ y_j^{(3)} &= \{1, -1, 0\} \\ \dots \quad & \dots \\ y_j^{(9)} &= \{1, -1, 0, -0.5774, 0.6587, -0.8393, 0.8700, 0.3056, -0.3217\} \\ \dots \quad & \dots \end{aligned}$$

We also note that the sequence we have described lacks symmetry on the interval $[-1, 1]$, although the nodes appear to be relatively well-distributed. A version of this scheme that introduces symmetric points have been described in [103]. In this version, the maximiser given by expression (3.37) is ignored when k is even, and instead, we set $y_k^{(i)} := -y_{k-1}^{(i)}$. This, of course, does not comport with definition (3.38), nor is it a Leja sequence more generally.

A key feature of this rule is that we have a nested sequence of points that can be refined by only adding one node at a time to our interpolant. This can be useful for the purposes of reducing the number of function evaluations necessary in our sparse grid to achieve a prescribed error tolerance. However, in comparison to Gaussian-type rules, they do not tend to be particularly efficient. Work has also been done here to establish results for Leja points with weightings applied to the maximiser function; that is, the maximiser given by expression (3.37) becomes:

$$\underset{y\in[-1,1]}{\arg\max} \sqrt{\mathfrak{v}(z)} \prod_{k=1}^{j-1} \left| y - y_k^{(i)} \right| \tag{3.39}$$

for some continuous, positive weight determined by the function, $\mathfrak{v} : [-1, 1] \to \mathbb{R}^+$. The results of doing this are discussed in [104].

Another popular choice for the collocation point indexing rule is that which was proposed by Clenshaw and Curtis [105]. This 'Clenshaw–Curtis' rule defines collocation points according to the roots of Chebyshev polynomials. For this rule, the quantity of collocation points per-index, $m_i \in \mathbb{N}$, is given by:

$$m_i := \begin{cases} i, & i = 0, 1, \\ 2^{i-1} + 1, & i \in \mathbb{N} \setminus \{1\}. \end{cases} \tag{3.40}$$

The locations of the points are then defined by:

$$y_j^{(i)} := \begin{cases} 0, & m_i = 1, \quad j = 1 \\ -\cos\left\{\dfrac{\pi(j-1)}{m_i - 1}\right\}, & m_i > 1, \quad j \in \{1, ..., m_i\}. \end{cases} \tag{3.41}$$

Using definition (3.41), we can evaluate the positions of the resulting nodes:

$$\begin{aligned} y_1^{(1)} &= 0 \\ y_j^{(2)} &= \{0, -1, 1\} \\ y_j^{(3)} &= \{0, -1, 1, -0.7071, 0.7071\} \\ y_j^{(4)} &= \{0, -1, 1, -0.7071, 0.7071, -0.9239, -0.3827, 0.3827, 0.9239\} \\ &... \qquad ... \end{aligned}$$

As with Leja points, the Clenshaw–Curtis scheme produces a nested set of points. This scheme also provides a clear closed-form expression for the positions of nodes, as opposed to the iterative formula outlined previously for the Leja points.

The disadvantage of the Clenshaw–Curtis rule is that, as described with definition (3.40), we are required to double the number of nodes at each level of refinement, which in theory, could make it harder to fine-tune the number of function evaluations required for a certain error tolerance. Despite this lack of *granularity*, however, it has been suggested in [103], as well as [106], that numerical results for Clenshaw–Curtis points appear to be comparable to those of Leja points, with slight performance improvements for quadrature-based calculations, and slight performance degradation for interpolation–based calculations.

3.3.3 An Illustrative Example

We now demonstrate some of the aforementioned concepts with an illustrative example. Specifically, to connect the ideas involved with the collocation point rules to Theorem 3.2, we

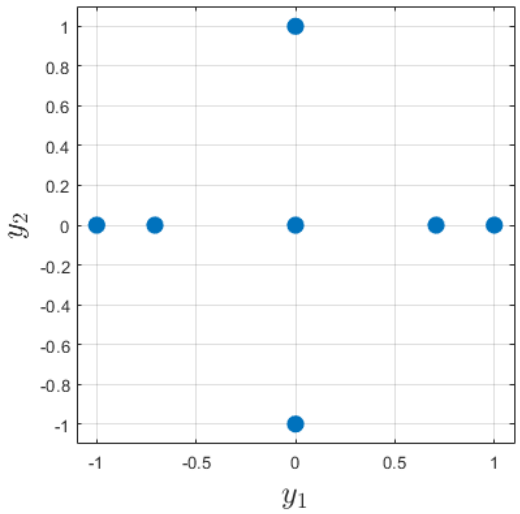


Figure 3.3: The locations of the collocation points in this example.

evaluate the Lagrange polynomials $L_{\boldsymbol{m}}$ for a given index set I, with the positions of nodes determined by the Clenshaw–Curtis rule.

In 2 dimensions (that is, $\Gamma = [-1, 1]^2$), we take the multi-index set given by:

$$I = \{(1, 1), (1, 2), (2, 1), (3, 1)\}. \tag{3.42}$$

From the definition of the Clenshaw–Curtis scheme, this leads to a set of 7 collocation points (recalling that for each index, i, in the multi-index, the number of points in that direction is $m_i = 2^{i-1} + 1$ whenever $i > 1$). In particular, using the framework of the representation given in Theorem 3.2, we calculate that with $\boldsymbol{i} \in I$ defined as above, we have the set:

$$\breve{I} = \{(1, 1), (1, 2), (1, 3), (2, 1), (3, 1), (4, 1), (5, 1)\}, \tag{3.43}$$

which corresponds to the collocation point set:

$$Y = \left\{(0, 0), (0, -1), (0, 1), (-1, 0), (1, 0), \left(-\frac{1}{\sqrt{2}}, 0\right), \left(\frac{1}{\sqrt{2}}, 0\right)\right\}. \tag{3.44}$$

We plot these collocation points in Figure 3.3. Applying to the definition of the Lagrange basis functions, $L_{\boldsymbol{m}}$, we calculate each of these via:

$$L_{\boldsymbol{m}}(\boldsymbol{y}) = \sum_{\substack{\boldsymbol{i} \in I \\ m(\boldsymbol{i}) \geqslant \boldsymbol{m}}} c_{\boldsymbol{i}} \left(L^{(i_1)}_{m_1}(y_1) \times L^{(i_2)}_{m_2}(y_2)\right). \tag{3.45}$$

To work out the relevant terms in the summation for each collocation point multi-index, $\boldsymbol{m} \in \breve{I}$, we construct the following table:

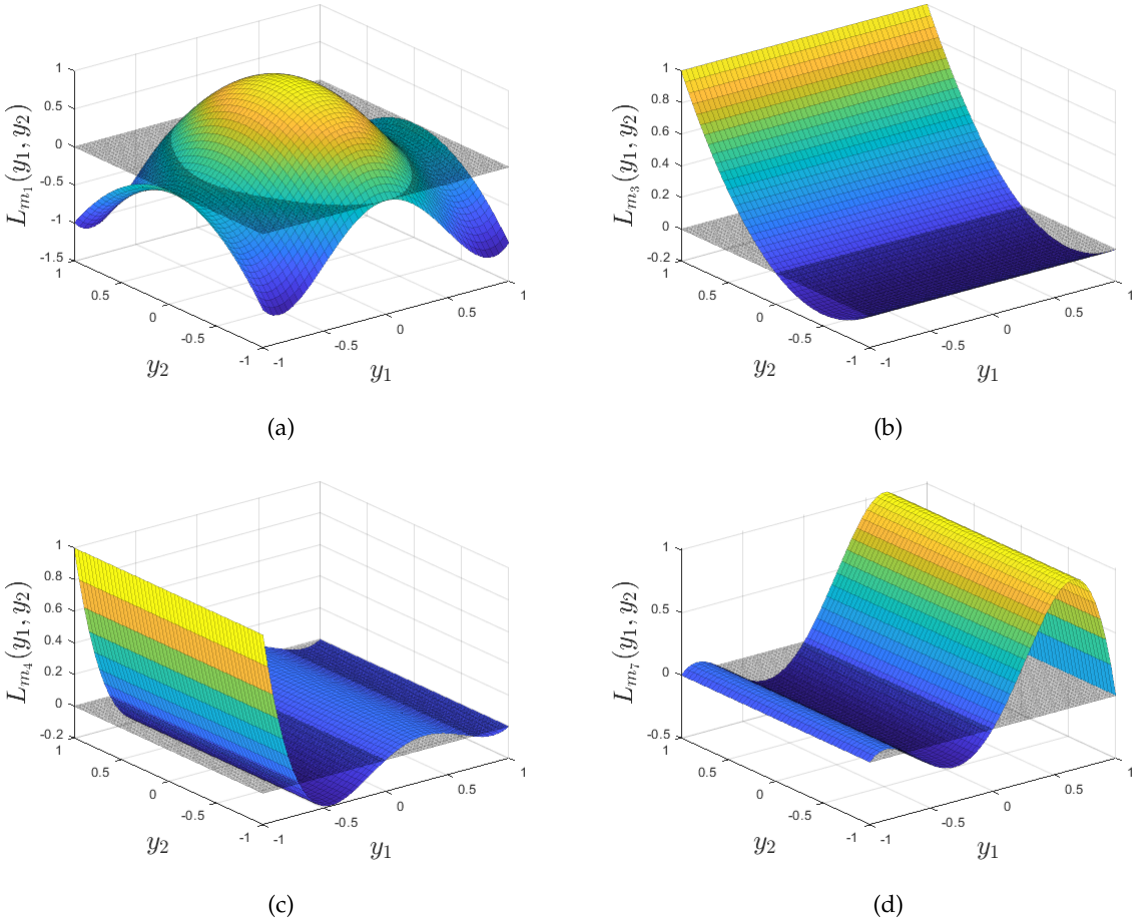
$\boldsymbol{m}_k \in \breve{I}$	$\{\boldsymbol{i} \in I : m(\boldsymbol{i}) \geqslant \boldsymbol{m}_k\}$	$\bar{\boldsymbol{y}} \in Y$
$\boldsymbol{m}_1 = (1,1)$	$\{(1,1),(1,2),(2,1),(3,1)\}$	$(0,0)$
$\boldsymbol{m}_2 = (1,2)$	$\{(1,2)\}$	$(0,-1)$
$\boldsymbol{m}_3 = (1,3)$	$\{(1,2)\}$	$(0,1)$
$\boldsymbol{m}_4 = (2,1)$	$\{(2,1),(3,1)\}$	$(-1,0)$
$\boldsymbol{m}_5 = (3,1)$	$\{(2,1),(3,1)\}$	$(1,0)$
$\boldsymbol{m}_6 = (4,1)$	$\{(3,1)\}$	$\left(-\frac{1}{\sqrt{2}},0\right)$
$\boldsymbol{m}_7 = (5,1)$	$\{(3,1)\}$	$\left(\frac{1}{\sqrt{2}},0\right)$

For each of the $\boldsymbol{i} \in I$, we also require the coefficients $c_{\boldsymbol{i}}$, which again, can be calculated using the definition provided in Theorem 3.2:

$\boldsymbol{i} \in I$	$\sum(-1)^{\lvert\boldsymbol{\alpha}\rvert}$	$c_{\boldsymbol{i}}$
$\boldsymbol{i} = (1,1)$	$(-1)^{\lvert(0,0)\rvert} + (-1)^{\lvert(0,1)\rvert} + (-1)^{\lvert(1,0)\rvert}$	-1
$\boldsymbol{i} = (1,2)$	$(-1)^{\lvert(0,0)\rvert}$	1
$\boldsymbol{i} = (2,1)$	$(-1)^{\lvert(0,0)\rvert} + (-1)^{\lvert(1,0)\rvert}$	0
$\boldsymbol{i} = (3,1)$	$(-1)^{\lvert(0,0)\rvert}$	1

These coefficients multiply the appropriate products of 1D Lagrange polynomials, $L_j^{(i)}$, which are given by:

Basis function $L_j^{(i)}(y)$	Resulting Polynomial	Position y_j
$L_1^{(1)}(y)$	1	0
$L_1^{(2)}(y)$	$1 - y^2$	0
$L_2^{(2)}(y)$	$y^2/2 - y/2$	-1
$L_3^{(2)}(y)$	$y^2/2 + y/2$	1
$L_1^{(3)}(y)$	$2y^4 - 3y^2 + 1$	0
$L_2^{(3)}(y)$	$y^4 - y^3 - y^2/2 + y/2$	-1
$L_3^{(3)}(y)$	$y^4 + y^3 - y^2/2 - y/2$	1
$L_4^{(3)}(y)$	$-2y^4 + \sqrt{2}y^3 + 2y^2 - \sqrt{2}y$	$-1/\sqrt{2}$
$L_5^{(3)}(y)$	$-2y^4 - \sqrt{2}y^3 + 2y^2 + \sqrt{2}y$	$1/\sqrt{2}$



(a) (b) (c) (d)

Figure 3.4: Representations of some of the calculated basis functions. In (a), (b), (c) and (d), we have $L_{\boldsymbol{m}_1}$, $L_{\boldsymbol{m}_3}$, $L_{\boldsymbol{m}_4}$ and $L_{\boldsymbol{m}_7}$, respectively.

Applying all of what we have above, we deduce that:

$$\begin{aligned}
L_{\boldsymbol{m}_1}(\boldsymbol{y}) &= c_{(1,1)}\left(L_1^{(1)}(y_1)\times L_1^{(1)}(y_2)\right) + c_{(1,2)}\left(L_1^{(1)}(y_1)\times L_1^{(2)}(y_2)\right) \\
&\quad + c_{(2,1)}\left(L_1^{(2)}(y_1)\times L_1^{(1)}(y_2)\right) + c_{(3,1)}\left(L_1^{(3)}(y_1)\times L_1^{(1)}(y_2)\right) \\
&= -1 + (1-y_2^2) + (2y_1^4 - 3y_1^2 + 1), \\
&= 2y_1^4 - 3y_1^2 - y_2^2 + 1 \\
L_{\boldsymbol{m}_2}(\boldsymbol{y}) &= c_{(1,2)}\left(L_1^{(1)}(y_1)\times L_2^{(2)}(y_2)\right) \\
&= y_2^2/2 - y_2/2, \\
L_{\boldsymbol{m}_3}(\boldsymbol{y}) &= c_{(1,2)}\left(L_1^{(1)}(y_1)\times L_3^{(2)}(y_2)\right) \\
&= y_2^2/2 + y_2/2, \\
L_{\boldsymbol{m}_4}(\boldsymbol{y}) &= c_{(2,1)}\left(L_2^{(2)}(y_1)\times L_1^{(1)}(y_2)\right) + c_{(3,1)}\left(L_2^{(3)}(y_1)\times L_1^{(1)}(y_2)\right) \\
&= y_1^4 - y_1^3 - y_1^2/2 + y_1/2, \\
L_{\boldsymbol{m}_5}(\boldsymbol{y}) &= c_{(2,1)}\left(L_3^{(2)}(y_1)\times L_1^{(1)}(y_2)\right) + c_{(3,1)}\left(L_3^{(3)}(y_1)\times L_1^{(1)}(y_2)\right) \\
&= y_1^4 + y_1^3 - y_1^2/2 - y_1/2, \\
L_{\boldsymbol{m}_6}(\boldsymbol{y}) &= c_{(3,1)}\left(L_4^{(3)}(y_1)\times L_1^{(1)}(y_2)\right) \\
&= -2y_1^4 + \sqrt{2}y_1^3 + 2y_1^2 - \sqrt{2}y_1, \\
L_{\boldsymbol{m}_7}(\boldsymbol{y}) &= c_{(3,1)}\left(L_5^{(3)}(y_1)\times L_1^{(1)}(y_2)\right) \\
&= -2y_1^4 - \sqrt{2}y_1^3 + 2y_1^2 + \sqrt{2}y_1.
\end{aligned}$$

By inspection, one can readily see that each of these modified Lagrange basis functions does indeed satisfy the delta property over the 2D collocation point set (3.44).

3.4 Stochastic Collocation FEM

Having discussed the setup of our sparse grid, as well as its corresponding interpolant and associated basis functions, we now move to describing the broader stochastic collocation framework. We recall our original formulation given in problem (3.2), as well as the 'sampled' weak formulation described by problem (3.4), which introduced a solution as a mapping $u : \Gamma \to H_0^1(D) =: V$ which satisfies

$$\int_D a(x,\bar{\boldsymbol{y}})\nabla u_{\bar{\boldsymbol{y}}}(x)\cdot\nabla v(x)\,\mathrm{d}x = \int_D f(x)v(x)\,\mathrm{d}x, \quad v\in V, \quad \rho\text{-a.e. in } \Gamma.$$

We know from [30] that the above problem is well-posed and therefore admits a unique solution for each $\boldsymbol{y} \in \Gamma$. In particular, this unique solution exists for a prescribed set of collocation points $Y \subset \Gamma$. Hence, across our set of collocation points, this represents a series of distinct, deterministic and decoupled problems that we can discretise and solve in a similar manner to what was discussed in Chapter 2.

At this stage, we will now explicitly note the dependence of discrete quantities on the different levels of refinement that are achieved at different iterations within the context of an adaptive algorithm. In particular, we use the bullet subscript ('$\bullet$') as a placeholder which denotes the fact that this dependence exists, but that the specific iteration does not affect our calculation.

In doing all of the above, we obtain sampled approximations, denoted by $u_{\bullet\bar{\boldsymbol{y}}} \in V_{\bullet\bar{\boldsymbol{y}}} \subset V$, which for each collocation point $\bar{\boldsymbol{y}} \in Y_\bullet$, solve the deterministic problems:

$$\int_D a(x,\bar{\boldsymbol{y}})\nabla u_{\bullet\bar{\boldsymbol{y}}}(x)\cdot\nabla v(x)\,\mathrm{d}x = \int_D f(x)v(x)\,\mathrm{d}x, \quad v \in V_{\bullet\bar{\boldsymbol{y}}}. \tag{3.46}$$

Here, the spaces $V_{\bullet\bar{\boldsymbol{y}}}$ represent finite element spaces associated with the spatial mesh for each collocation point. In a similar manner to definition (2.14), we set $V_{\bullet\bar{\boldsymbol{y}}} := \mathcal{S}_0^p(\mathcal{T}_{\bullet\bar{\boldsymbol{y}}})$ for each $\bar{\boldsymbol{y}} \in Y_\bullet$. For the purposes of this work, we will typically consider the case $p = 1$ for practical purposes (that is, piecewise linear finite element approximations); however, many of the ideas discussed are also implementable for polynomials of higher degree.

Another assumption that we will make is that at a given level of refinement, we will assume the same triangulation for each collocation point within our set. In other words, we define $\mathcal{T}_\bullet := \mathcal{T}_{\bullet\bar{\boldsymbol{y}}}$ and $V_\bullet := V_{\bullet\bar{\boldsymbol{y}}}, \forall\, \bar{\boldsymbol{y}} \in Y_\bullet$. Approximations that are based on calculations involving this assumption are typically called *single-level* approximations. Again, this assumption is not strictly necessary within the context of stochastic collocation, and the topic of *multi-level* collocation has been studied in [42]; however, for reasons that will be alluded to in Chapter 6, we do not deem this to be necessary for our work here.

With the above assumption, we can, for each $\boldsymbol{y} \in Y_\bullet$, represent the *semi-discrete* formulation given by Problem 3.46 in the compact form:

$$B_{\bar{\boldsymbol{y}}}(u_{\bullet\bar{\boldsymbol{y}}}, v_\bullet) = F(v_\bullet), \quad \forall\, v_\bullet \in V_\bullet, \tag{3.47}$$

where

$$B_{\bar{\boldsymbol{y}}}(u_{\bullet\bar{\boldsymbol{y}}}, v_\bullet) = \int_D a(x,\bar{\boldsymbol{y}})\nabla u_{\bullet\bar{\boldsymbol{y}}}(x)\cdot\nabla v_\bullet(x)\,\mathrm{d}x,$$
$$F(v_\bullet) = \int_D f(x)v_\bullet(x)\,\mathrm{d}x.$$

Upon solving each of these problems as we did in Chapter 2, we can write our full stochastic collocation FEM approximation as

$$u_\bullet^{SC}(x,\boldsymbol{y}) = \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \tag{3.48}$$

where, following our discussion of Theorem 3.2, $L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \in \mathbb{P}(\Gamma)$ are multivariate Lagrange basis functions, which satisfy the delta property for our set of collocation points, $Y_\bullet$. That is, we have $L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') = \delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'}, \forall\, \bar{\boldsymbol{y}}, \bar{\boldsymbol{y}}' \in Y_\bullet$, and

$$u_\bullet^{SC}(x,\bar{\boldsymbol{y}}') = \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') = \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)\delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'} = u_{\bullet\bar{\boldsymbol{y}}'}(x).$$

We note the omission of $\bullet$ in our description of the Lagrange basis functions. For the purposes of our work, this can be done without ambiguity, and it is further understood that these functions will be dependent on (and often appear in sums over) the underlying collocation point set, $Y_\bullet$, so that the dependence is described implicitly in any case.

In addition to calculating the solution approximation (3.48), we can also calculate quantities that are derived from this. The expected value (or mean) of $u_\bullet^{SC}$ is given by

$$\begin{aligned}\mathbb{E}\left[u_\bullet^{SC}(x,\boldsymbol{y})\right] &= \int_\Gamma \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y}) \\ &= \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)\int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y}).\end{aligned} \tag{3.49}$$

The variance of the approximation is given by

$$\begin{aligned}\mathbb{V}\left[u_\bullet^{SC}\right] &:= \mathbb{E}\left[\left(u_\bullet^{SC} - \mathbb{E}[u_\bullet^{SC}]\right)^2\right] \\ &= \mathbb{E}\left[\left(u_\bullet^{SC}\right)^2\right] - \mathbb{E}\left[u_\bullet^{SC}\right]^2\end{aligned} \tag{3.50}$$

which can be calculated using

$$\begin{aligned}\mathbb{V}\left[u_\bullet^{SC}\right] &= \int_\Gamma \left(\sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\right)^2 \mathrm{d}\rho(\boldsymbol{y}) - \left(\sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)\int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y})\right)^2 \\ &= \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)u_{\bullet\bar{\boldsymbol{y}}'}(x)\mathbb{V}_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'}\end{aligned}$$

where

$$\mathbb{V}_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'} = \int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}) - \int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}) \int_\Gamma L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}).$$

We can also consider more exotic quantities, such as the expectation of some other feature of the solution, rather than just the solution itself. This is, of course, a core topic of this thesis, and we will see much more of this in Chapters 4 and 5.

3.5 Adaptive Algorithms

Naturally, we wish to estimate the error associated with our stochastic collocation FEM approximation, and use our estimates to drive an adaptive algorithm. This task is now made more difficult by the fact that we have approximation errors associated with both the spatial discretisation and the sampling of our parameter domain. If one of these components is of too great a magnitude, then reducing the errors in the other component will be of little use to us for the purposes of reducing the total error.

We wish to use our usual adaptive loop in order to drive the errors associated with our approximation below a prescribed tolerance, $\texttt{tol} = \varepsilon$.

3.5.1 Error Estimation

Having previously discussed aspects of the stochastic collocation FEM approximation, we begin by discussing approaches to error estimation.

One possible strategy is a residual-based technique, which has been described in [37]. In Section 2.4.1, we saw how this could be implemented in the deterministic setting. The hierarchical approach proceeds in a similar manner here, with the main theoretical additions including the interpolation property associated with the sparse grid interpolant and the decomposition of integrals into sums over collocation points or indices. The core result that is established as a result of this is stated below, although it specifically applies to affine diffusion coefficients (as given in representation (3.13)), rather than the more general diffusion coefficients that we wish to consider in this framework.

Theorem 3.3. *Let $u : \Gamma \to V$ satisfy the sampled weak formulation (3.4), and let $u_\bullet^{SC} \in V_{\bullet\bar{\boldsymbol{y}}} \otimes \mathbb{P}(\Gamma)$ be the stochastic collocation FEM approximation formed by discrete approximations $u_{\bullet\bar{\boldsymbol{y}}} : \Gamma \to V_{\bullet\bar{\boldsymbol{y}}}$ which solve problem (3.47) for each $\boldsymbol{y} \in Y_\bullet$, with $Y_\bullet$ computed using the index set $I_\bullet$. Define element-based and edge-based residuals by:*

$$r_K(u_{\bullet\bar{\boldsymbol{y}}}(x)) := f(x) + \nabla \cdot (a(x, \bar{\boldsymbol{y}})\nabla u_{\bullet\bar{\boldsymbol{y}}}(x))$$
$$j_K(u_{\bullet\bar{\boldsymbol{y}}}(x)) := \frac{a(x, \bar{\boldsymbol{y}})}{2} \left[\!\left[\frac{\partial u_{\bullet\bar{\boldsymbol{y}}}(x)}{\partial \boldsymbol{n}} \right]\!\right]$$

Then we have that $\exists C^\sharp, C^\flat > 0$, dependent only on the shape regularity of the triangulation $\mathcal{T}_\bullet$, such that for any $p \in \mathbb{N} \cup \{\infty\}$,

$$\left\| u - u_\bullet^{SC} \right\|_{\mathcal{V}} \leqslant \frac{1}{C_\flat^2}(C^\sharp \mu_\bullet + \tau_\bullet), \tag{3.51}$$

where

$$\mu_\bullet := \sum_{\boldsymbol{y} \in Y_\bullet} \mu_{\bullet\bar{\boldsymbol{y}}} \|L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\|_\Gamma, \quad \mu_{\bullet\bar{\boldsymbol{y}}} := \left(\sum_{K \in \mathcal{T}_\bullet} \mu_{\bullet\bar{\boldsymbol{y}},K}^2 \right)^{\frac{1}{2}} \tag{3.52}$$

with

$$\mu_{\bullet\bar{\boldsymbol{y}},K} := h_K^2 \|r_K\|_K^2 + \sum_{E \subset \partial K} h_E \|j_K\|_E, \tag{3.53}$$

and

$$\tau_\bullet := \sum_{\boldsymbol{i} \in \mathcal{M}_{I_\bullet}} \left\| \Delta^{m(\boldsymbol{i})}(a \nabla u_\bullet^{SC}) \right\|_{\Gamma;D}. \tag{3.54}$$

There are a few observations to be made about the above. Each of the $\mu_{\bullet\bar{\boldsymbol{y}}}$ defined in definition (3.52) are entirely deterministic and adopt the rule of controlling the spatial error for each collocation point $\bar{\boldsymbol{y}} \in Y_\bullet$, where the norms of the Lagrange basis functions $\|L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\|_\Gamma$ act as weights for these spatial errors. As a result of this, the estimate $\mu_\bullet$ corresponds to an estimate for the error in the finite element approximations – that is, it is a measure of the 'spatial' error. By contrast, the $\tau_\bullet$ corresponds to the error in the stochastic collocation procedure and therefore corresponds to the error in the parameter domain.

We also note that the components in the expressions for $\mu_\bullet, \tau_\bullet$ can easily be used as localised indicators for the purposes of driving adaptive refinement. In addition, as $\mu_{\bullet\bar{\boldsymbol{y}},K}$ is specifically a per-element and per-collocation point quantity, this approach can easily be extended to the setting of *multi–level* adaptivity, where different spatial meshes can be used for each collocation point.

Finally, in definition (3.54), we observe that the summation over multi-indices only occurs over the margin, $\mathcal{M}_{I_\bullet}$, of the index set $I_\bullet$. This is only made possible by the fact that the diffusion coefficient has an affine dependence on the random parameters, and represents a significant drawback with this estimation result. Whether this approach can be adapted to the case of non-affine diffusion coefficients or not remains an open question.

Another possible approach to error estimation in this framework is a hierarchical framework, as discussed by [41]. In order to pursue this type of strategy, we want to begin by introducing some form of enhanced, or refined, solution of our approximation. At this point, however, there are two different ways we can enhance our solution - we can refine the spatial mesh, as we did before, or we can refine our sparse grid.

We begin by defining new formulations, we find solutions $\hat{u}_{\bullet\bar{\boldsymbol{y}}} \in \widehat{V}_\bullet$ and $\tilde{u}_{\bullet\bar{\boldsymbol{y}}} \in V_\bullet$ which satisfy the relations:

$$B_{\bar{\boldsymbol{y}}}(\hat{u}_{\bullet\bar{\boldsymbol{y}}}, v_\bullet) = F(v_\bullet), \quad \forall\, v_\bullet \in \widehat{V}_\bullet \supset V_\bullet \tag{3.55}$$

and

$$B_{\bar{\boldsymbol{y}}}(\tilde{u}_{\bullet\bar{\boldsymbol{y}}}, v_\bullet) = F(v_\bullet), \quad \forall\, v_\bullet \in V_\bullet, \tag{3.56}$$

respectively. In a similar manner as before, equations (3.55) and (3.56) are solved as separate problems for all collocation points $\bar{\boldsymbol{y}} \in Y_\bullet$ and $\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet \supset Y_\bullet$, respectively. The refined collocation point set, $\widetilde{Y}_\bullet$ should be determined by a monotone index set $\widetilde{I}_\bullet \supset I_\bullet$, for example, $\widetilde{I}_\bullet := I_\bullet \cap \mathcal{R}_{I_\bullet}$. These formulations lend themselves to the refined stochastic collocation FEM approximations:

$$\hat{u}_\bullet^{SC} := \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \hat{u}_{\bullet\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}), \tag{3.57}$$

and

$$\tilde{u}_\bullet^{SC} := \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet} \tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x) \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}), \quad \text{with} \quad \tilde{u}_{\bullet\bar{\boldsymbol{y}}} = u_{\bullet\bar{\boldsymbol{y}}},\ \forall\, \bar{\boldsymbol{y}} \in Y_\bullet \subset \widetilde{Y}_\bullet. \tag{3.58}$$

where $\widetilde{L}_{\bar{\boldsymbol{y}}}$ are the Lagrange basis functions corresponding to the new set $\widetilde{Y}_\bullet$. Note that, in general $L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \neq \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y})$, even for the original set of collocation points $Y_\bullet \subset \widetilde{Y}_\bullet$. We want a way of accurately representing the difference between discretisation errors for a given spatial mesh and for the sparse grid utilised for the collocation procedure. We therefore define an enhanced solution:

$$u_\star^{SC} := \hat{u}_\bullet^{SC} + \left(\tilde{u}_\bullet^{SC} - u_\bullet^{SC}\right). \tag{3.59}$$

This is immediately useful to us, as it yields the trivial result:

$$u_\star^{SC} - u_\bullet^{SC} = \left(\hat{u}_\bullet^{SC} - u_\bullet^{SC}\right) + \left(\tilde{u}_\bullet^{SC} - u_\bullet^{SC}\right), \tag{3.60}$$

with the first of these brackets representing the a deterministic error component, and the second bracket representing a stochastic error component. Then, provided that the saturation assumption,

$$\left\|u - u_\star^{SC}\right\| \leqslant \varkappa \left\|u - u_\bullet^{SC}\right\|, \tag{3.61}$$

holds for some $\varkappa \in (0,1)$, we have that

$$\begin{aligned} \left\|u - u_\bullet^{SC}\right\|_{\mathcal{V}} &\leqslant \frac{1}{1-\varkappa}\left\|u_\star^{SC} - u_\bullet^{SC}\right\|_{\mathcal{V}} \\ &\leqslant \frac{1}{1-\varkappa}\left(\left\|\hat{u}_\bullet^{SC} - u_\bullet^{SC}\right\|_{\mathcal{V}} + \left\|\tilde{u}_\bullet^{SC} - u_\bullet^{SC}\right\|_{\mathcal{V}}\right). \end{aligned} \tag{3.62}$$

From here, we can then derive estimates for each of the two norms $\left\|\hat{u}_\bullet^{SC} - u_\bullet^{SC}\right\|_{\mathcal{V}}$ and $\left\|\tilde{u}_\bullet^{SC} - u_\bullet^{SC}\right\|_{\mathcal{V}}$. These norms represent estimates for the spatial and parametric components of the total discretisation error, respectively. Analysis of these norms yields the upper bounds provided in the theorem below.

Theorem 3.4. *Suppose that the stochastic collocation FEM approximations $u_\bullet^{SC}$ and $\hat{u}_\bullet^{SC}$ are given as in definitions (3.48) and (3.57), respectively. Then we have that $\exists\, C > 0$, depending only on the regularity of the triangulation $\mathcal{T}_\bullet$, such that*

$$\mu_\bullet := \left\|\hat{u}_\bullet^{SC} - u_\bullet^{SC}\right\|_{\mathcal{V}} \leqslant \frac{C}{a_\flat} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \mu_{\bullet\bar{\boldsymbol{y}}} \left\|L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\right\|_\Gamma \tag{3.63}$$

with

$$\mu_{\bullet\bar{\boldsymbol{y}}} := \left(\sum_{K \in \mathcal{T}_\bullet} \mu_{\bullet\bar{\boldsymbol{y}},K}^2\right)^{1/2}$$

where $a_\flat > 0$ is the constant from condition (3.1), $\mu_{\bullet\bar{\boldsymbol{y}},K}$ are per-collocation point, per-element two-level error indicators analogous to those discussed in Section 2.4.3.

Further, suppose that $\tilde{u}_\bullet^{SC}$ is given by definition (3.58). Then we have

$$\tau_\bullet := \left\|\tilde{u}_\bullet^{SC} - u_\bullet^{SC}\right\|_{\mathcal{V}} \leqslant \left\| \sum_{\boldsymbol{i} \in \widetilde{I}_\bullet \setminus I_\bullet} \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_{\bullet\boldsymbol{i}}} (u_{\bullet\bar{\boldsymbol{y}}}(x) - u_\bullet^{SC}(x,\bar{\boldsymbol{y}}))\widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \right\|_{\mathcal{V}} \tag{3.64}$$

where the set $\widetilde{Y}_{\bullet\boldsymbol{i}}$ is the set of collocation points that are (uniquely) generated by an individual multi-index $\boldsymbol{i} \in \widetilde{I}_\bullet \setminus I_\bullet$.

Firstly, we briefly note that other types of hierarchical estimation strategies exist and were discussed in the same paper from which the above result is taken. The two-level error estimation strategy is of particular significance, however, due to the lack of a requirement of

solving linear systems resulting from the strategy. We will briefly provide some numerical comparisons of these different strategies in Section 3.7.

We also observe that these estimates lend themselves to natural indicators that can be used for local refinement. In the case of the contribution to the error from the parameter domain, we specifically re-wrote a set of collocation points $\widetilde{Y}_\bullet$ in terms of the underlying indices which induce them. The associated summands can then be used as per-index indicators. The sets $\widetilde{Y}_{\bullet \boldsymbol{i}}$ satisfy

$$\widetilde{Y}_\bullet \setminus Y_\bullet = \bigcup_{\boldsymbol{i} \in \widetilde{I}_\bullet \setminus I_\bullet} \widetilde{Y}_{\bullet \boldsymbol{i}}, \quad \text{and} \quad \widetilde{Y}_{\bullet \boldsymbol{i}} \cap \widetilde{Y}_{\bullet \boldsymbol{i}'} = \emptyset, \quad \forall\, \boldsymbol{i}, \boldsymbol{i}' \in \widetilde{Y}_\bullet \setminus Y_\bullet \quad \text{s.t.} \quad \boldsymbol{i} \neq \boldsymbol{i}', \tag{3.65}$$

and we will see these again in subsequent chapters of this thesis.

Secondly, we note that the results above, in contrast to the residual estimate given in Theorem 3.3, apply to non-affine representations of the diffusion coefficient. Furthermore, as discussed in Section 2.4, hierarchical approaches (including the two-level estimator) have innate relevance to the concept of reductions in errors yielded by specific local refinements. This usefulness carries over to the stochastic collocation FEM setting. One potential issue with this approach that carries over from the deterministic framework is the use of the saturation assumption (3.61), which is not guaranteed to be satisfied.

With respect to error estimation in general, we also observe, as we did in Chapter 2, that it is entirely possible to use the ideas of this section to generate indicators associated with both elements, $K \in \mathcal{T}_\bullet$, as well as edges, $E \in \mathcal{E}_\bullet$. That being said, we will typically continue to refer to indicators and marked sets associated with elements for the theory described in this chapter, as well as subsequent chapters.

One more thing that we will note is that instead of using the approximation defined by representation (3.56), we can also define

$$u_\star^{SC} := \hat{u}_\bullet^{SC} - (\tilde{u}_{\bullet \mathbf{0}}^{SC} - u_{\bullet \mathbf{0}}^{SC}), \tag{3.66}$$

where

$$u_{\bullet \mathbf{0}}^{SC} := \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} u_{\mathbf{0}\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}), \tag{3.67}$$

and

$$\tilde{u}_{\bullet \mathbf{0}}^{SC} := \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet} \tilde{u}_{\mathbf{0}\bar{\boldsymbol{y}}}(x) \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}), \quad \text{with} \quad \tilde{u}_{\mathbf{0}\bar{\boldsymbol{y}}} = u_{\mathbf{0}\bar{\boldsymbol{y}}},\ \forall\, \bar{\boldsymbol{y}} \in Y_\bullet \subset \widetilde{Y}_\bullet, \tag{3.68}$$

so that we are using the coarse mesh $\mathcal{T}_0$, rather than $\mathcal{T}_\bullet$, to perform these calculations. The constant, $C > 0$, in Theorem 3.4, will then depend on the mesh $\mathcal{T}_0$ instead. This modification is easy to implement, and the use of a coarse has the obvious advantage of reducing the computational expense associated with the calculation.

3.5.2 Marking and Refinement

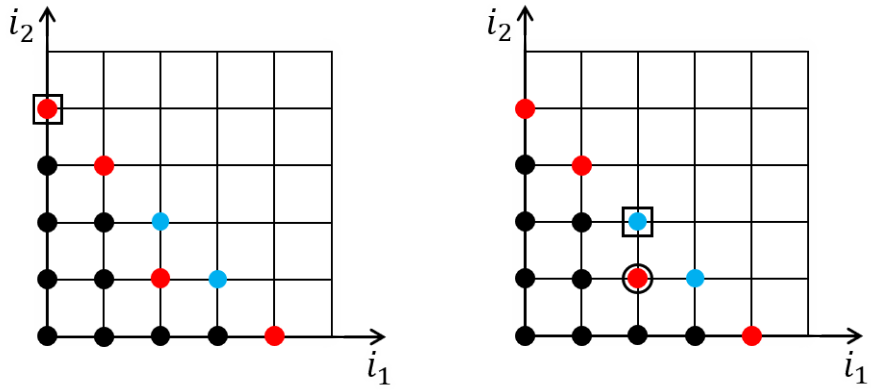
We now move to the subject of obtaining a marked set for the purpose of refinement. Before we decide how to generate marked sets, we first need to decide what type of refinement we wish to pursue; the generation of a marked set of elements, is not necessary, for example, if we only decide to perform parametric refinement. To this end, we can compare the spatial and parametric estimates $\mu_\bullet$ and $\tau_\bullet$, and choose to perform spatial refinement if $\mu_\bullet \geqslant \tau_\bullet$, and parametric refinement if this is not the case.

We note that computing these estimates at each iteration will quickly become computationally expensive, and we may therefore choose some amalgamation of the local indicators their place to determine whether spatial or parametric refinement is necessary. In particular, for the two-level estimate discussed in Theorem 3.4, the upper bounds given by inequalities (3.63) and (3.64) would be appropriate.

We typically refer to these amalgamations of indicators as 'indirect', or 'cumulative' error estimates, to be denoted by $\bar{\mu}_\bullet$ and $\bar{\tau}_\bullet$. These estimates are less computationally expensive than the corresponding 'direct' estimates, $\mu_\bullet$ and $\tau_\bullet$, although they represent coarser bounds, and are therefore not particularly desirable with respect to being used as part of a termination criterion.

Having selected whether to pursue spatial or parametric refinement, and after generating localised per-element spatial indicators $\mu_{\bullet \bar{\boldsymbol{y}},K}$, and per-index parametric error indicators $\tau_{\bullet \boldsymbol{i}}$, we now must discuss how the generation of the marked sets will function.

In the case of spatial refinement, the per-element indicators $\mu_{\bullet \bar{\boldsymbol{y}},K}$ can be used to produce marked sets for each collocation point. we can use the Dörfler marking strategy, or a similar strategy described in Section 2.5, in order to accomplish this. These sets can then be combined into a single marked set, $\widehat{\mathcal{M}}_\bullet$, by taking a simple union of all of the marked sets.



(a) (b)

Figure 3.5: The procedure for refinement of an index set, I using the margin $\mathcal{M}_I$. The boxed indices from the sets $\mathcal{R}_I$ (red) and $\mathcal{M}_I \setminus \mathcal{R}_I$ (blue) are selected for refinement in (a) and (b), respectively. Circled indices are not marked, but must be added to obtain the monotonic set $\widetilde{I}_\bullet$.

In the case of parametric refinement, we observe that for both the residual and two-level approaches, we wrote upper bounds for our parametric errors contributions in terms of indices from a new index set. Doing this, rather than simply writing these contributions in terms of collocation points, $\bar{\boldsymbol{y}} \in Y_\bullet$, is critical for the purposes of refinement, as our refinement procedure does not necessarily add individual collocation points at a time. Instead, this procedure will consist of adding new multi-indices $\boldsymbol{i} \in \mathbb{N}^M \setminus I_\bullet$ to our index set to generate a new monotone index set, $\widetilde{I}_\bullet$, for the next iteration of the adaptive algorithm. We can use the same strategy that we did for spatial refinement in order to achieve this.

With respect to the practicalities of the refinement step, the refinement of the spatial mesh was something we discussed in Chapter 2. In particular, we can use longest edge bisection, or similar variants of newest edge bisection routines, in order to obtain new triangulations, $\widehat{\mathcal{T}}_\bullet := \texttt{refine}\,(\mathcal{T}_\bullet, \widehat{\mathcal{M}}_\bullet)$. The key observation to make here is that a specific refinement rule is required in order to obtain a refined spatial mesh. As with the deterministic case, this means that the refined triangulation will typically contain more elements and/or edges than those that we marked in the previous step of the algorithm.

On the other hand, if we consider indices from the reduced margin, $\mathcal{R}_{I_\bullet}$ for marking, the enrichment of our index set induces no such complications. We can add marked indices from our index set individually, and the resulting set, $\widetilde{I}_\bullet$, will itself be monotonic. If we were to use

the full margin, $\mathcal{M}_{I_\bullet}$, as is required by the residual-based approach, then we need to introduce an additional procedure for whenever indices selected for refinement are in the set $\mathcal{M}_{I_\bullet} \setminus \mathcal{R}_{I_\bullet}$ in order to ensure that the new set, $\widetilde{I}_\bullet := \texttt{refine}\,(I_\bullet, \widetilde{\mathcal{M}}_\bullet)$, remains monotonic. We illustrate this idea in Figure 3.5.

On refining either the spatial or parametric domains, we can move to the next iteration of the adaptive algorithm. In the case where we have just performed spatial refinement, we are required to re-solve problems on the new spatial mesh and proceed from there. Where we have completed parametric refinement, our next stochastic collocation FEM approximation will resemble definition (3.58). In particular, as the sets of collocation points that will be utilised throughout the algorithm are nested, many of the required sampled solutions have already been computed in the previous iteration. Hence, we are only required to solve problems for the new set of collocation points $\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet \setminus Y_\bullet$.

We continue through successive iterations, until we arrive at an iteration where we have calculated a direct error estimate, and this estimate is lower than our stopping criterion. At this point, we can terminate the adaptive loop and proceed to post-processing without any further refinements. It is at this stage that we have a final approximation u_*^{SC}, as well as a corresponding error estimate that is smaller than the error tolerance that was originally set for the adaptive process. The key points of the algorithm are summarised in Algorithm 3.1.

3.6 Implementation via a Stochastic Collocation FEM toolbox

We now turn to aspects of implementation via MATLAB. For this, we use the code that was previously utilised in order to generate results in [41], but has since been modified to create the Adaptive ML-SCFEM toolbox. Running the software requires us to call the main driver, `singlelevelSC`, which contains all of the components in our adaptive loop, and other aspects of processing that are required to allow our algorithm to function.

We begin our algorithm with an initialisation stage. This is largely accomplished with the script `stochcol_adaptive_global_settings`, which allows choices for the type of diffusion coefficient, random variable, estimation strategy, as well as other features. We set a unit right-hand side for the purposes of simplicity, although the associated quantity can be modified

Algorithm 3.1: Stochastic Collocation FEM algorithm

Data: Initial spatial mesh, $\mathcal{T}_0$, initial index set $I_0 = \{\mathbf{1}\}$; error tolerance $\texttt{tol} = \varepsilon$; problem parameters, including marking criteria and a refinement procedure.

Result: Stochastic collocation FEM approximation u_*^{SC}; corresponding error estimates satisfying $\mu_* + \tau_* \leqslant \varepsilon$.

Set iteration counter $k = 0$; iteration factor $l \in \mathbb{N}$.

while $\mu_l + \tau_l \geqslant \varepsilon$ **do**

Obtain the approximations, $u_{k\bar{\boldsymbol{y}}} \in V_k$ by solving problem (3.56) for each $\bar{\boldsymbol{y}} \in Y_k$.

For each $\bar{\boldsymbol{y}} \in Y_k, K \in \mathcal{T}_k$, compute spatial error indicators $\mu_{k\bar{\boldsymbol{y}},K}$, and for each $\boldsymbol{i} \in \widetilde{I}_k \setminus I_k$, compute parametric error indicators $\tau_{k\,\boldsymbol{i}}$.

Use the indicators to determine indirect estimators $\bar{\mu_k}$ and $\bar{\tau_k}$.

For each $\bar{\boldsymbol{y}} \in Y_k$, determine the marked spatial set $\widehat{\mathcal{M}}_k := \bigcup_{\bar{\boldsymbol{y}} \in Y_k} \widehat{\mathcal{M}}_{k,\bar{\boldsymbol{y}}}$ and the marked index set $\widetilde{\mathcal{M}}_k$.

if $k = jl,\ j \in \mathbb{N}$ **then**

Compute the direct error estimates μ_l, τ_l.

end

if $\mu_l + \tau_l \geqslant \varepsilon$ **then**

if $\bar{\mu_k} < \bar{\tau_k}$ **then**

Define $I_{k+1} := \texttt{refine}\ (I_k, \widetilde{\mathcal{M}}_k)$, and set $\mathcal{T}_{k+1} := \mathcal{T}_k$.

else

Define $\mathcal{T}_{k+1} := \texttt{refine}\ (\mathcal{T}_k, \widehat{\mathcal{M}}_k)$ and set $I_{k+1} := I_k$.

end

Set $k \to k + 1$.;

end

end

Set $u_*^{SC} := u_l^{SC}, \mu_* := \mu_l, \tau_* := \tau_l$.

if necessary. After setting the initial mesh and retrieving the initial index set, we proceed to our first major step of the adaptive loop.

For the solution step, the fact that we are solving distinct, decoupled problems at each collocation point implies that the solver we used in the deterministic setting can, in no small part, be repurposed here. Further, the collocation approach lends itself to the parallelisation of parts of the procedure across the set of collocation points. For the first iteration, this obviously makes no difference to our calculation, as I_0 consists of a single collocation point, but at latter iterations, it is known that the use of `parfor`-based statements will yield reductions in CPU time associated with computationally intensive tasks.

Hence, we run a parallelised code which calls the function `stochcol_fem_solver` to solve problem (3.47) at each collocation point, $\boldsymbol{y} \in Y_\bullet$. In the same loop, we use the function `stochcol_fem_estimator` to provide local estimation of errors associated with each of the finite element approximations, and generate per-collocation point marked sets associated with the spatial mesh.

As we do not wish to be restricted to the use of affine diffusion coefficients, Theorem 3.3 is of limited use to us for the purposes of error estimation, and we instead prefer to proceed with hierarchical strategies. In particular, three different choices of hierarchical estimation strategy are given in our estimation function, with the implementations for each of these being described for the deterministic setting in Chapter 2. These strategies, when applied in the stochastic collocation FEM setting, use the same functions that were utilised in the T-IFISS toolbox discussed in that chapter. The two-level estimator, which we recall is implemented via the `diffpost_p1_with_p1_2level` function, is part of Theorem 3.4, while the other implemented estimators are also discussed in [41]. These constitute components of the indirect spatial estimate, which is assembled in the main driver.

The per-index parametric estimates are calculated using information about estimates for finite element solutions at prospective nodes corresponding to the reduced margin, $\mathcal{R}_I$. This information is computed using a variety of functions, including the solver, and a function for calculating vectors of multivariate Lagrange polynomials, $\widetilde{L}_{\bar{\boldsymbol{y}}}$, while the indirect estimate itself is computed in the function `stochcol_est_parametric`.

The last key estimation function is `stochcol_direct_estimator`, which is used to compute

the estimates that will be used to determine whether the stopping criterion has been met. If it has not, then a refinement of the spatial or parametric domain is computed as required, using `stochcol_mesh_refine` or `stochcol_get_grid`, respectively, in order to achieve this. Then the adaptive loop continues until the stopping criterion has been met, at which point, a series of post-processing steps are completed, including the calculation and plotting of the mean and variance of the final approximation.

While not part of the main driver, the script `reference_SC` can also be run in order to compute a 'reference' solution that can be used for additional analysis of the performance of our estimation strategy. We will describe the details of this in the next section.

3.7 Experiments

We conclude this chapter with some illustrative examples that will demonstrate the concepts that we have described in previous sections of this chapter.

We begin with a simple experiment involving the computing of a stochastic collocation FEM approximation on a unit square domain, $D = (0,1)^2$. We set $f(x) = 1$ in equation (3.4), and let the diffusion coefficient represent a truncated affine expansion of $M = 4$ random parameters, so that

$$a(x, \boldsymbol{y}) = a_0(x) + \sum_{m=1}^{M} y_m a_m(x). \tag{3.69}$$

We must also define the different expansion coefficients, a_m. Following [107] (and others), we use coefficients that describe planar Fourier modes of increasing total order, so that $a_0(x) := 1$, and

$$a_m(x) := Am^{-\sigma_a} \cos\left(2\pi\beta_1(m)x_1\right) \cos\left(2\pi\beta_2(m)x_2\right), \quad m = 1, ..., M. \tag{3.70}$$

Here, we take

$$\beta_1(m) = m - \frac{k(m)(k(m)+1)}{2}, \quad \beta_2(m) = k(m) - \beta_1(m), \tag{3.71}$$

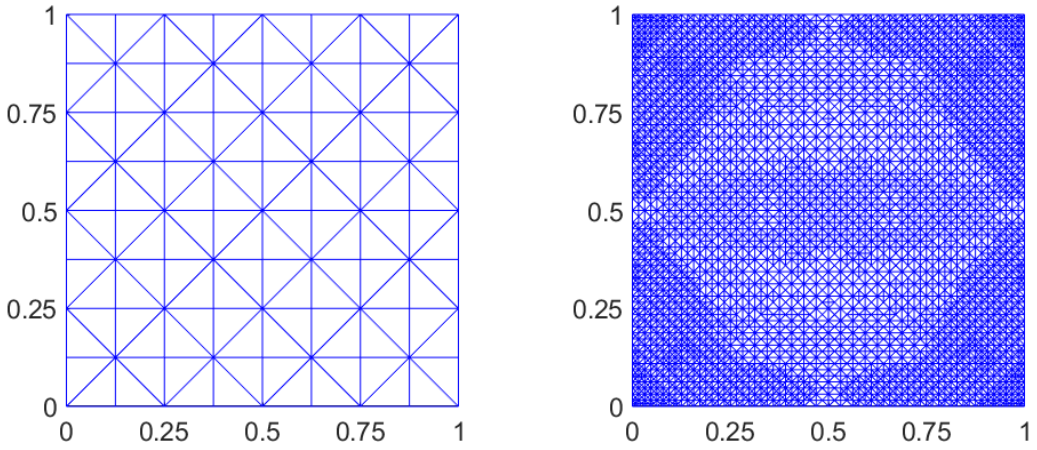
with $k(m) = \lfloor -1/2 + \sqrt{1/4 + 2m} \rfloor$. The parameters of this expansion can be varied; however, for the purposes of this experiment, we set $\sigma_a = 2$, and $A = 0.547$.

The values above indicate a slow decay of the mode amplitudes, and are common choices, not least because they can also be used in the stochastic Galerkin FEM setting. Indeed, with these

values, we know that

$$A\,\zeta(\sigma_a) = 0.547\frac{\pi^2}{6} < 1,$$

where ζ is the Riemann zeta function. This is sufficient to ensure that even condition (3.14) is satisfied, and that the equivalent problem in the stochastic Galerkin FEM setting is well-posed with a convergent diffusion coefficient expansion. For this example, the random variables themselves are assumed to be uniformly distributed, with probability density function $\varrho_m(y_m) = 1/2$, for each $m = 1, ..., 4$.



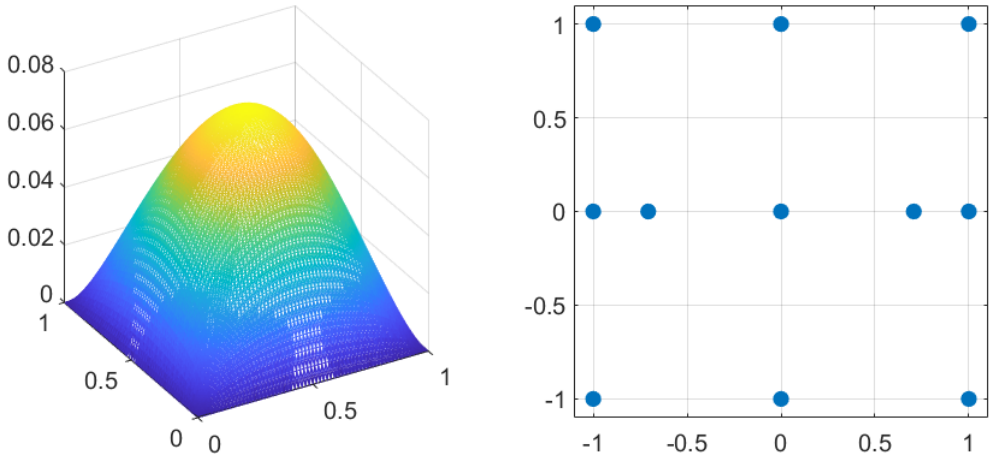
(a) (b)

Figure 3.6: Diagrams pertaining to the spatial domain involved with the first test problem: the initial mesh $\mathcal{T}_0$ in (a), and the mesh at $\mathcal{T}_{20}$ in (b).

Our spatial error estimation will be done initially with the element-wise residual variant, and it is from this test that we produce the associated plots in Figures 3.6 and 3.7. The initial spatial mesh is given in the first of these, consisting of a total of 128 elements, arranged into 16 blocks of 8 elements each. For the parameter domain, we begin with the collocation point set induced by the mutli-index set $I_0 = \{\mathbf{1}\}$; that is, Y_0 consists of a single collocation point at the origin. For index sets more generally, we will use the Clenshaw–Curtis rule to determine the number and location of the resulting collocation points.

We will also adopt Dörfler marking techniques with associated spatial and parametric marking parameters of $\theta_x = 0.3$ and $\theta_{\boldsymbol{y}} = 0.3$, respectively, with edge-based marking (rather than element-based marking) for the spatial mesh. We take the iteration factor $l = 1$ in Algorithm 3.1 to recover more data about the performance of our estimation routine at each iteration, and

will continue to do so during other experiments.

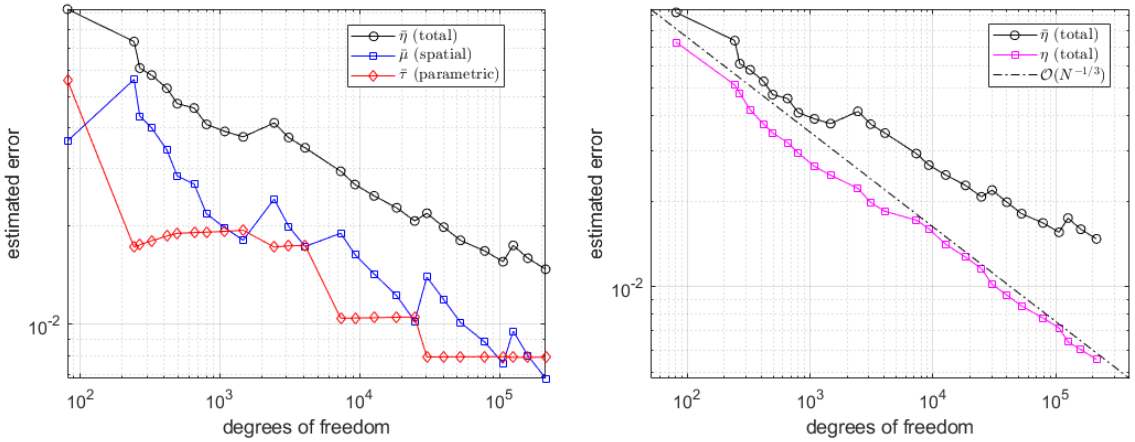


(a) (b)

Figure 3.7: The mean of the solution for our first test problem in (a), and the final set of collocation points in the first two dimensions in (b).

We run our algorithm to a tolerance $\texttt{tol} = \texttt{6e} - \texttt{3}$, and obtain an approximation with estimated error lower than this required tolerance after 25 iterations, at which point, we have $N = 214,149$ degrees of freedom resulting from $32,456$ elements and 13 collocation points. The adaptive process correctly identifies weak singularities in the corners of the domain, hence the refinement pattern in Figure 3.6 (b). Furthermore, the final index set, given by $I_{25} = \{(1,1,1,1),(1,1,2,1),(1,2,1,1),(2,1,1,1),(2,2,1,1),(3,1,1,1)\}$, is indicative of greater refinement in earlier parameters than in the latter ones (with no refinement at all in parameter 4). Given the form of our diffusion coefficient in representation (3.70), this is entirely expected.

We now turn to the estimates of the errors admitted by our solution. With Figure 3.8 (a), we highlight the interplay between spatial and parametric refinement. The total indirect error is in black, with the spatial contribution in blue and the parametric contribution in red. On each of the five occasions where the parametric contribution to the indirect estimate is greater than the direct estimate, a single index has been added to our index set, which allows for the associated error to be decreased further. Meanwhile, Figure 3.8 (b) reveals the importance of estimating the total error directly, rather than through an indirect procedure involving summation of a set of indicators. In particular, we see that whenever a parametric refinement occurs, the total indirect error estimate actually increases at the following iteration. This causes a deterioration



(a) (b)

Figure 3.8: Error estimates associated with the first test problem: spatial and parametric contributions to the indirect estimate in (a), and a comparison between direct and indirect estimates in (b).

in the rate of convergence, and leaves us with a more coarse estimate as a result. The magenta line that represents the direct error estimate, by contrast, experiences no such deterioration, reproducing the converge rate of around $\mathcal{O}(N^{-1/3})$ found in [41].

From here, we now seek to examine the performance of the estimator itself, and compare this with the performance of the two other hierarchical estimation implementations described in Section 2.7, and later alluded to in Section 3.6. We conduct further experiments with the same selection of parameters, to the same tolerance.

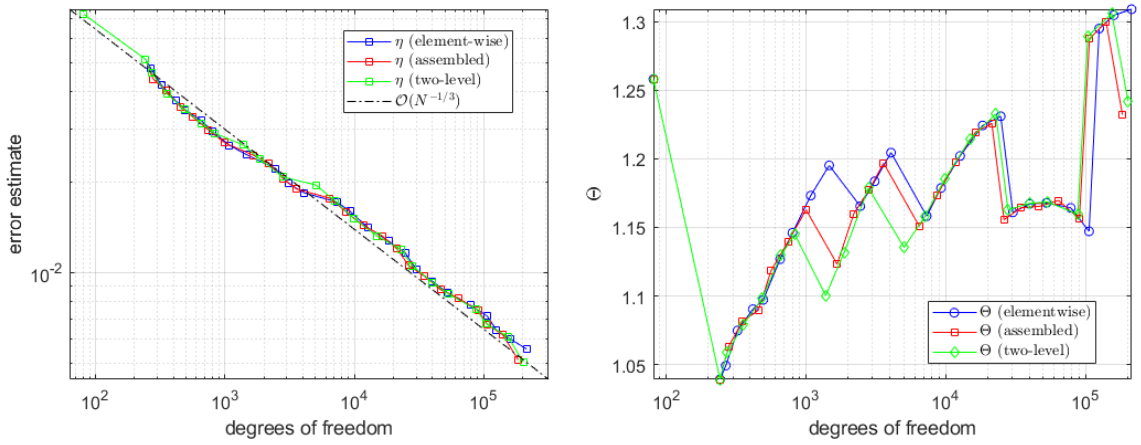
On comparing the three types of hierarchical spatial estimation, we find that they are comparable in terms of their performance, although the element-wise residual-based approach performed more slowly, and required additional degrees of freedom to achieve the required tolerance, as shown in Table 3.1. The two-level error estimator occasionally requires fewer iterations to complete, and the lack of a requirement for solving linear systems also assists us in terms of computational expense.

While requiring fewer iterations could provide advantages in terms of computational expense over the entire adaptive process, this could also lead to unfavourable consequences. In particular, if the penultimate step of the adaptive progress leads to an estimate that is just above the set error tolerance, this could increase the likelihood of more degrees of freedom

Method	Element-wise	Assembled	Two-level
Error Estimate (Direct)	`5.5536e-03`	`5.1185e-03`	`5.0220e-03`
Degrees of Freedom	214, 149	184, 059	202, 096
Collocation Points	13	17	17
Iterations	25	24	21
Parametric Refinements	5	6	6
Time Elapsed	103.45 sec	83.85 sec	83.98 sec

Table 3.1: Data corresponding to different types of hierarchical estimator after running the first test problem to a tolerance of $\texttt{tol} = \texttt{6e} - 3$.

being required in a solution that does achieve the required tolerance, when compared with other strategies. Further experiments reveal that this is something that does occasionally arise in minor modifications of the setup for this test problem (for example, by adjusting the error tolerance, or aspects of the diffusion coefficient). The error decays associated with this particular setup are displayed in Figure 3.9 (a).



(a) (b)

Figure 3.9: A comparison of different hierarchical estimation strategies, with estimated errors in (a) and effectivity indices in (b).

In each case, we also compare the effectivity indices associated with this method. This requires a comparison between our error estimates and the corresponding 'true' error in the Bochner norm, $\left\|u - u_{\bullet}^{SC}\right\|_{\mathcal{V}}$. In practical applications, we would not compute this 'true' error; however,

we do this here for the purposes of verifying the quality of our estimate. Of course, this is not something we can calculate per-se, as we do not know the true solution, $u \in \mathcal{V}$.

As a substitute for the above, we calculate a proxy of the true error. This is generated using a reference solution $u_{ref} \approx u$, which is computed on a finite space that constitutes a uniform refinement of both the spatial and parametric domains at the final iteration of our adaptive algorithm. In this case, we utilise a piecewise-quadratic approximation on a uniformly refined spatial domain, and we add the entire reduced margin to the current index set to get the reference index set. From this, we obtain a significantly more accurate surrogate approximation for the true solution such that the relative error admitted by the approximation

$$\left\|u - u_{\bullet}^{SC}\right\|_{\mathcal{V}} \approx \left\|u_{ref} - u_{\bullet}^{SC}\right\|_{\mathcal{V}}$$

is sufficiently low at each iteration.

On completing this approximation, and comparing this to our error estimate, we can generate the effectivity indices given in Figure 3.9 (b). We can see that the effectivity indices produced by each error estimation strategy are similar, in each case remaining between 1 and 1.5 for the entirety of the adaptive process. These results appear to be representative of the behaviour of these estimation strategies, and while minor modifications to this test problem can adjust the values of the effectivity indices, these modifications not appear to significantly alter the conclusion that the estimation strategies behave similarly.

For our second experiment, we will look at an aspect of the parametric contribution to our algorithm, namely the rule for determining collocation points. For this setup, we will introduce a second test problem, where we maintain our definition of the right-hand side functional from the previous example, as well as the unit square domain, $D = (0, 1)^2$.

However, we modify our definition of the diffusion coefficient, so that it has a non-affine dependence on random parameters. In particular, with a_m defined as in equation (3.70), we define a quadratic expansion

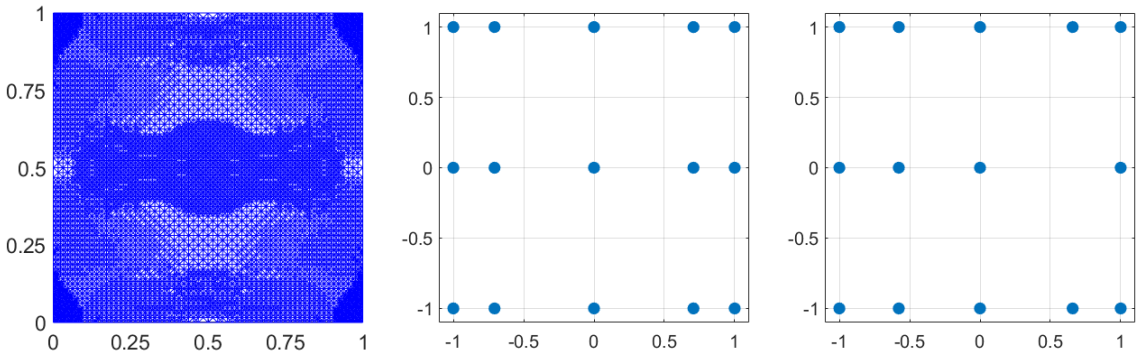
$$a(x, \boldsymbol{y}) = \left(a_0(x) + \sum_{m=1}^{M} y_m a_m(x)\right)^2, \tag{3.72}$$

where we set $M = 4$, so that our expansion consists of four random variables, as in the previous example. We now assume that these random variables are distributed according to a truncated

normal (or Gaussian) distribution, with probability density function

$$\varrho_m(y_m) := \frac{\sigma_\rho}{\sqrt{2\pi}} \operatorname{erf}\left(\frac{1}{\sqrt{2}\sigma_\rho}\right) e^{-\frac{1}{2}\left(\frac{y_m}{\sigma_\rho}\right)^2} \tag{3.73}$$

for each $m = 1, ..., M$. For this example, we set the standard deviation, $\sigma_\rho = 1$.



(a) (b) (c)

Figure 3.10: Diagrams computed for the second test problem: the mesh at $\mathcal{T}_{20}$ in (a), and the final set of collocation points in the first two dimensions for Clenshaw–Curtis points in (b), and for Leja points in (c).

We mark edges in the spatial setting, using a two-level error estimation strategy. For marking in the spatial and parametric domains, we retain our Dörfler marking strategy with threshold parameters $\theta_x = 0.3$ and $\theta_{\boldsymbol{y}} = 0.3$, respectively. We also keep the same initial mesh $\mathcal{T}_0$, and the same initial index set, I_0, from our previous example. Our adaptive algorithm was run to the same tolerance as the first test problem, i.e., $\texttt{tol} = \texttt{6e} - \texttt{3}$.

The mean of the solution to this problem, which we do not plot here, is very similar to our first problem. We do plot the mesh $\mathcal{T}_{20}$, which is taken from running the adaptive algorithm with the Clenshaw–Curtis rule being used to determine the quantity and locations of collocation points. We also plot the final sets of collocation points in the first two dimensions, along with the set that is obtained after using the Leja rule.

For the Clenshaw–Curtis rule, our adaptive process completed in 154.3 seconds, and yielded an approximation involving $464,535$ degrees of freedom. By contrast, when using Leja points, the number of degrees of freedom for the final approximation was significantly higher, at $1,534,000$, while the process as a whole took 466.0 seconds to run. Both approximations, as with the previous test problem, exhibited considerable degrees of anisotropy with respect to

Method	Leja	Clenshaw–Curtis
Error Estimate (Direct)	`5.9093e-03`	`5.8042e-03`
Degrees of Freedom	$1,534,000$	$464,535$
Collocation Points	50	27
Iterations	34	21
Parametric Refinements	22	9
Time Elapsed	465.99 sec	154.30 sec

Table 3.2: Data corresponding to different collocation point rules after running the second test problem to a tolerance of $\texttt{tol} = \texttt{6e} - \texttt{3}$.

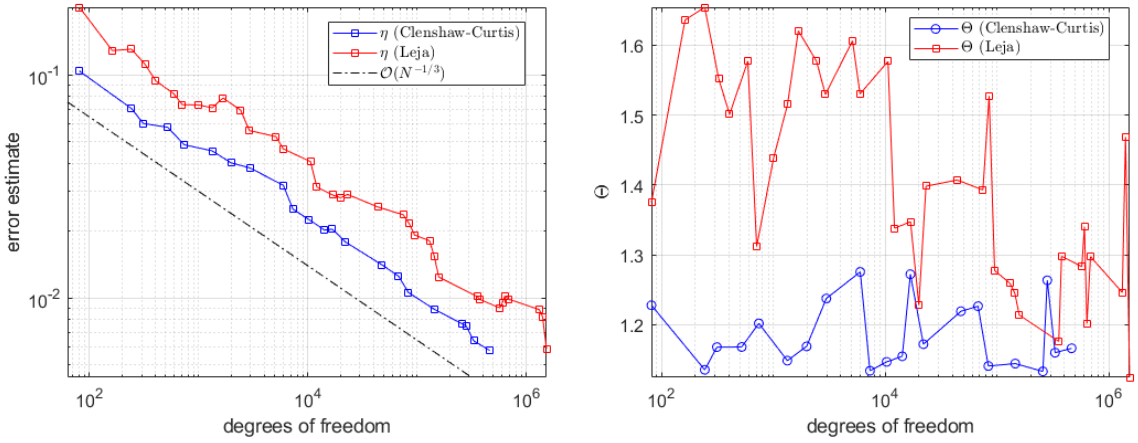
the final index sets, with earlier dimensions containing more collocation points than later ones. We also include additional data concerning the results for our second test problem in Table 3.2.

There are a few observations that we can immediately make about our results. Firstly, we observe that our adaptive algorithm required additional parametric refinements when our collocation points were determined via the Leja rule, which is unsurprising as only a single collocation point is added with each index. The ancillary effect of this is excess computations are performed with Leja abscissas that in order to motivate the inclusion of additional quantities of collocation points that would already be generated (albeit in slightly different locations) by the Clenshaw–Curtis rule. Indeed, running the same experiment for our first test problem yielded 17 collocation points after 6 parametric refinements in the case of Clenshaw–Curtis nodes, but in the case of Leja points, 11 parametric refinements (some involving the addition of multiple indices from the reduced margin simultaneously) were required to achieve a total of 18 collocation points.

That being said, this is not the only phenomenon that is occurring here, as is demonstrated by the results corresponding to this second test problem. In particular, for the approximation that used Leja points, the associated adaptive process not only required more parametric refinements, but it also resulted in more collocation points in the approximation, despite the fact that the Clenshaw–Curtis rule, on average, added more collocation points for each parametric refinement.

Hence, our main conclusion is that, at least for our purposes, increased granularity of Leja

points did not lead to a decrease in the total degrees of freedom required in an approximation with an estimated error that was below our tolerance threshold. In fact, surprisingly, the total degrees of freedom associated with our final approximation was significantly higher than the equivalent approximation using Clenshaw–Curtis nodes. This is a trend that is replicated with both of the test problems described, and with minor modifications to each of these.



(a) (b)

Figure 3.11: A comparison of Clenshaw–Curtis nodes and Leja nodes, with estimated errors in (a) and effectivity indices in (b).

In order to investigate why this appears to be the case, we compare the convergence of the error estimates using Clenshaw–Curtis and Leja rules. Furthermore, we also compare the effectivity indices for each of these rules using a reference solution, as previously described for the first test problem.

Our calculations for the error estimates, given in Figure 3.11 (a), reveal that both rules lead to the expected convergence rate of $\mathcal{O}(N^{-1/3})$, so that, at the very least, the use of Leja points does not lead to a deterioration in the order of convergence. However, the error decay associated with the approximation using Clenshaw–Curtis nodes appears to be more stable, and the error estimate itself is consistently lower throughout the entire adaptive procedure. Furthermore, despite the fact that effectivity indices were higher for our approximation using Leja points, our proxy for the true error associated with the final iteration was 0.005255 to 6 decimal places, compared with 0.004977 for the Clenshaw–Curtis rule. In other words, the use of Leja points for the duration of the algorithm did not produce a more accurate numerical approximation, despite the additional computational expense induced by this choice.

Finally, we perform some experiments in which we compare the use of the reduced margin with the full margin to describe the enhanced set of collocation points associated with parametric marking and refinement. As alluded to in Section 3.6, we typically use the reduced margin for this purpose, and have done so in our previous experiments; however, we use this opportunity to investigate whether having access to more collocation points at each refinement step yields any additional benefit to us. For this, we introduce a third test problem, with an L-shaped domain, $D = (-1, 1)^2 \setminus (-1, 0]^2$.

We retain the unit right-hand side from our previous test problems, and choose another non-affine diffusion coefficient. Specifically, we choose the exponential diffusion coefficient

$$a(x, \boldsymbol{y}) = \exp\left(a_0(x) + \sum_{m=1}^{M} y_m a_m(x)\right), \tag{3.74}$$

with, again, $M = 4$, and the individual coefficient components a_m defined as in the previous two examples. We also take the underlying random variables to have uniform probability distributions.

For the purposes of the adaptive algorithm, we continue with our use of two-level spatial error estimation, with edge-based marking. We also continue our use of a Dörfler marking strategy with parameters $\theta_x = 0.3$ and $\theta_{\boldsymbol{y}} = 0.3$. Further, the Clenshaw–Curtis rule will be used with respect to determining our collocation nodes.

As with all of our previous experiments, we take an initial index set $I_0 = \{\mathbf{1}\}$, and our initial spatial mesh is a mesh consisting of 96 elements, arranged in blocks of 8 in the same way as the initial mesh we saw in the previous test problem. In addition to this, we also plot the spatial mesh, $\mathcal{T}_{20}$, associated with our test involving the reduced margin. For the test involving the reduced margin, after running our adaptive algorithm with a tolerance $\texttt{tol} = \texttt{6e} - \texttt{3}$, we end up with the solution plotted in Figure 3.12 (a), which has $137,790$ degrees of freedom associated with it. By contrast, our approximation with the full margin being used for our enhanced index set yields a final approximation involving $220,968$ degrees of freedom.

We first note the features associated with our approximation. In particular, most of the heaviest areas of refinement are located in the corners of the domain, including re-entrant corner, at the origin. These heavier levels of refinement are entirely expected and are consistent with deterministic setting, where use of this domain induces similar geometric singularities.

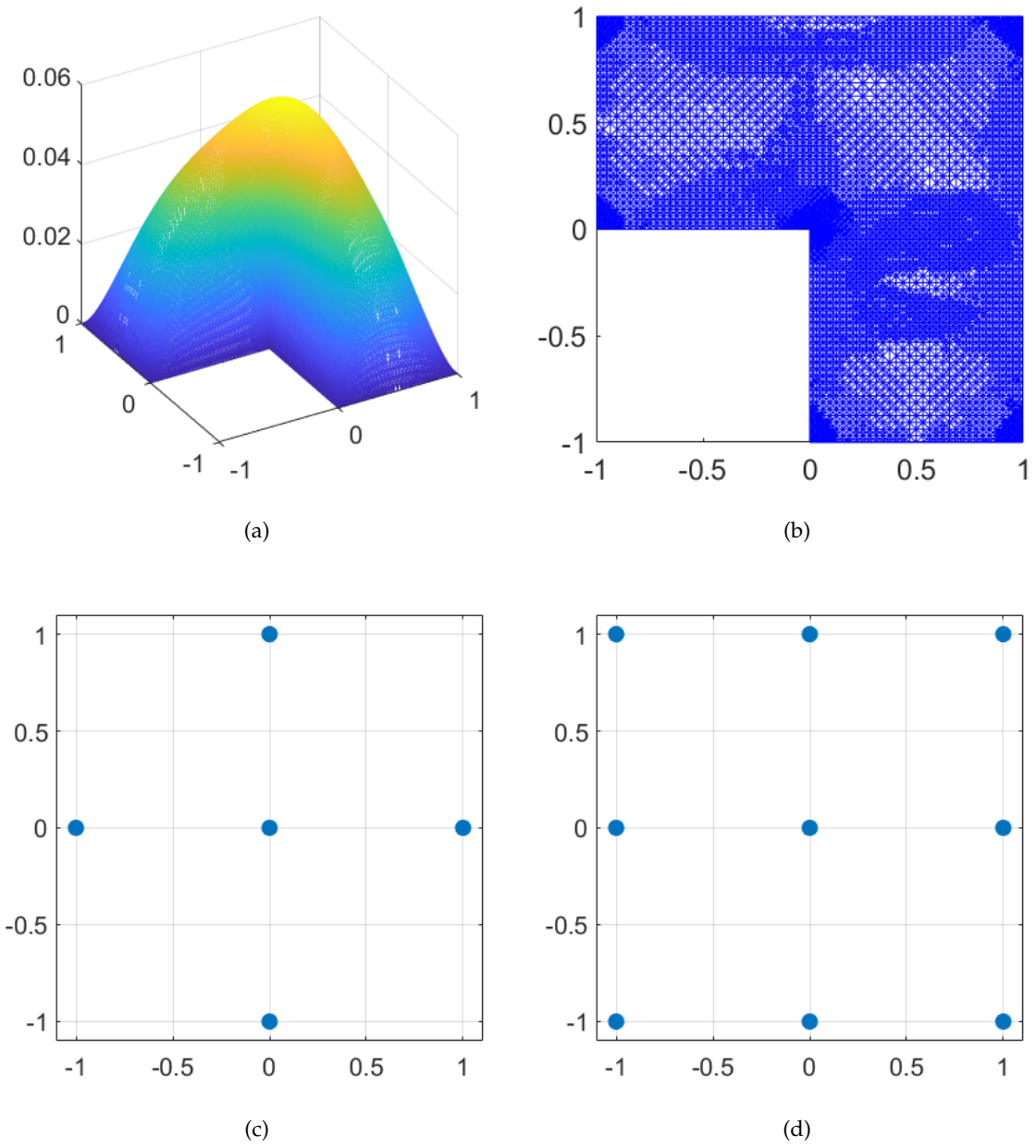


Figure 3.12: Diagrams pertaining to the third test problem: the solution in (a), the spatial mesh at $\mathcal{T}_{20}$ in (b), and the final set of collocation points in the first two dimensions for Clenshaw–Curtis points in (c), and for Leja points in (d).

Method	Reduced Margin	Full Margin
Error Estimate (Direct)	`5.9783e-03`	`5.9144e-03`
Degrees of Freedom	$137,790$	$220,968$
Elements	$23,386$	$16,049$
Collocation Points	5	27
Iterations	23	20
Parametric Refinements	2	3
Time Elapsed	69.89 sec	135.45 sec

Table 3.3: Data corresponding to different collocation point rules after running the third test problem to a tolerance of $\texttt{tol} = \texttt{6e} - \texttt{3}$.

The refinement of the parameter domain, however, is where our results become interesting. We find that use of the reduced margin results for this problem leads to an index set that is not particularly developed, with 2 parametric refinements across the adaptive procedure leading to the final approximation using just 5 collocation points. The approximation involving the full margin requires a similar number of parametric refinements, but ends up with 27 collocation points.

We immediately hypothesise that the reason for this is that during some of the marking steps, we end up marking indices from the margin, which forces us to add additional indices to ensure that the resulting index set is monotonic. For this setup, this does indeed occur at iteration 19, where the indices $(2, 2, 2, 1)$ and $(2, 1, 2, 1)$ are added together.

Further experimenting shows that running this comparison with our first setup (that is, with a square domain and an affine coefficient) yields an even more egregious example, as indices from the margin are sometimes chosen in such a way that multiple additional indices from the reduced margin must be added in order to create a monotonic index set at the next step of the adaptive algorithm. In that specific case, the adaptive algorithm that uses the reduced margin completes in around 88 seconds, with the final approximation using 17 collocation points. By contrast, the full margin requires 580 seconds to complete, and the resulting approximation ends up utilising 135 collocation points. The data associated with this additional sub-experiment is given in Table 3.4.

Method	Reduced Margin	Full Margin
Error Estimate (Direct)	`5.0220e-03`	`5.8588e-03`
Degrees of Freedom	202,096	504,900
Elements	23,386	7,266
Collocation Points	17	135
Iterations	15	21
Parametric Refinements	6	5
Time Elapsed	88.33 sec	580.83 sec

Table 3.4: Data corresponding to different collocation point rules after running the first test problem to a tolerance of $\texttt{tol} = \texttt{6e} - 3$.

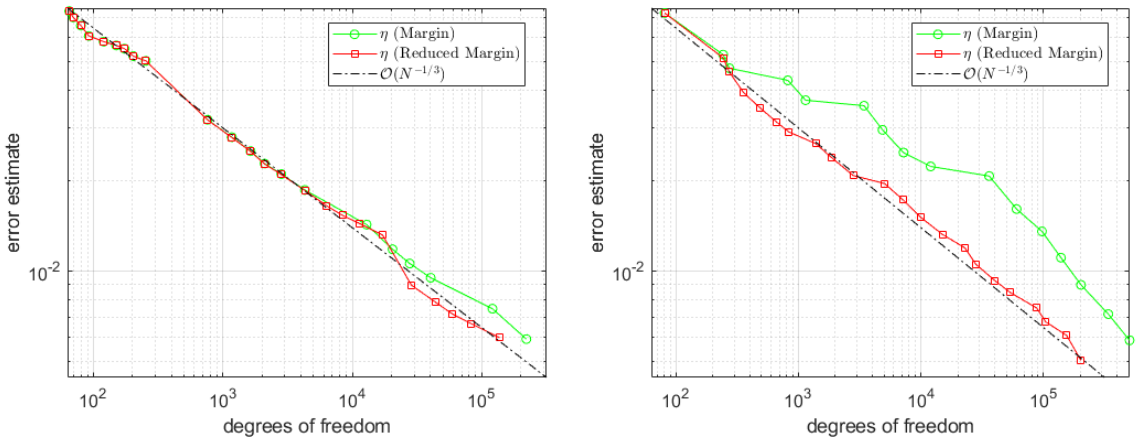


Figure 3.13: Error decays corresponding to two test problems: the third test problem, with an exponential coefficient and an L-shape domain in (a), and the first test problem, with an affine coefficient and a unit square domain in (b).

Part of the reason for the additional computational expense of running the adaptive procedure can be seen when looking at the decay of the estimated error. In Figure 3.13, we plot the estimated errors associated with the two setups that we have discussed and provided data for. For both of these cases, and for others, we see evidence of sub-optimal parametric over-refinement, which leads to a temporary, but significant deterioration in the rate of convergence. This, in turn, leads to the associated error estimates being much larger for the full margin, which then clearly has ancillary affects on both the time required to run the algorithm, and total degrees of freedom in the final approximation.

In this instance, we have revealed the necessity of part of the adaptive strategy described for the residual estimate in [37]. Where additional collocation points must be added, the adaptive algorithm as described in this work does not factor in the additional cost of adding further indices to the approximation. This is not as likely to be an issue when considering only the reduced margin, as comparatively few indices will be added at a time. This is particularly true when considering either Leja points, or earlier levels of refinement for Clenshaw–Curtis points.

In the case of the full margin, however, a multi-index outside of the reduced margin may have a larger contribution to the error estimate, but many collocation points may need to be added to generate a new monotonic index set. The result of this is that particular indices may not provide improvements to the approximations that are significant enough, relative to the cost of adding them, for their addition to be worthwhile. We therefore emphasise that without some consideration of the per-collocation point profits associated with adding indices, the full margin should not be used for the purposes of adaptively refining index sets.

Chapter 4

Goal-Oriented Adaptivity

In the previous chapter, we provided an explanation of the stochastic collocation FEM process, including the problem setup, the form of the solution, and the associated algorithm. Furthermore, we provided novel interpretations and proofs of existing results, as well as examples and experiments which demonstrated the underlying concepts which are utilised in this process. In Section 2.8, however, we discussed the fact that we may not just be interested in the underlying solution, but instead, in some underlying quantity of interest, or goal functional, related to that solution. We then gave an account of the goal-oriented framework in the deterministic setting. This chapter will be focused on extending goal-oriented error estimates to the stochastic collocation FEM setting.

We begin by setting out the framework of our problem in the goal-oriented setting. We then proceed to examine potential approaches to estimation using an enhanced solution splitting with a view to recovering the product-based estimate from Section 2.8. Ancillary results are proved in Section 4.2, along with an estimation result that, for reasons we will explain, should be considered sub-optimal for the purposes of our work. After this, we will introduce the correction term, as well as the resulting estimation strategy, which will be the main feature of this chapter.

4.1 Problem formulation

Naturally, we recognise that we must begin by introducing a goal functional. We assume that instead of the solution $u \in \mathcal{V}$, we seek a quantity dependent on a linear goal functional $\mathcal{Q} : \mathcal{V} \to \mathbb{R}$. In particular, we assume that like with the deterministic setting, we have a continuous goal functional $Q : V \to \mathbb{R}$, and we then define $\mathcal{Q}$ to be the mean value:

$$\mathcal{Q}(u) := \int_\Gamma Q(u(x, \boldsymbol{y}))\, \mathrm{d}\rho(\boldsymbol{y}). \tag{4.1}$$

Again, the underlying functional, Q, can be range of quantities, for example, the average of the solution across the sub-domain, or a flux-based quantity.

Mirroring the complete primal problem (3.7), the linear goal functional $\mathcal{Q}$ lends itself to a complete dual problem, which requires us to find $z \in \mathcal{V}$ such that:

$$\mathcal{B}(v, z) = \mathcal{Q}(v), \quad \forall v \in \mathcal{V}, \tag{4.2}$$

where, we recall that

$$\mathcal{B}(v, z) = \int_\Gamma \int_D (a(x, \boldsymbol{y}) \nabla v(x, \boldsymbol{y})) \cdot \nabla z(x, \boldsymbol{y})\, \mathrm{d}x\, \mathrm{d}\rho(\boldsymbol{y})$$

We also have an equivalent sampled dual formulation, corresponding to (3.4), that can be written in compact form. Given $\bar{\boldsymbol{y}} \in \Gamma$, we have:

$$B_{\bar{\boldsymbol{y}}}(v, z_{\bar{\boldsymbol{y}}}) = Q(v), \quad v \in V, \quad \rho\text{-a.e. in } \Gamma \tag{4.3}$$

where:

$$B_{\bar{\boldsymbol{y}}}(u_{\bar{\boldsymbol{y}}}, v) := \int_D a(x, \bar{\boldsymbol{y}}) \nabla u_{\bar{\boldsymbol{y}}}(x) \cdot \nabla v(x)\, \mathrm{d}x. \tag{4.4}$$

We note that, in a similar manner to the 'sampled' primal problem (3.4), we can write

$$Q(v) := \int_D q(x) v(x)\, \mathrm{d}x \tag{4.5}$$

as we did with F in definition (3.6); however, this is not typically necessary for the development of our theory, and we may desire ulterior representations of the goal functional Q.

We can then perform a spatial discretisation of problem (4.3). The result of this is that we obtain the 'semi-discrete' form:

$$B_{\bar{\boldsymbol{y}}}(v_\bullet, z_{\bullet\bar{\boldsymbol{y}}}) = Q(v_\bullet), \quad \forall\, v_\bullet \in V_\bullet, \tag{4.6}$$

which can be solved for each collocation point $\bar{\boldsymbol{y}} \in Y_\bullet$. With our dual problems defined, we can generate the stochastic collocation FEM approximation associated with our dual formulation in the same way as for our primal approximation, given with representation (3.48). Doing this yields

$$z_\bullet^{SC}(x, \boldsymbol{y}) = \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} z_{\bullet\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}). \tag{4.7}$$

Our objective will then be to estimate the error in our approximations to the quantity of interest, $\mathcal{Q}(u)$.

4.2 Explicit hierarchical error estimation

With the formulations (3.7) and (4.2), we want to estimate the quantity $\mathcal{Q}(u) - \mathcal{Q}(u_\bullet^{SC})$. The goal-oriented approach in the deterministic setting would typically involve us exploiting orthogonality, so our calculation would proceed as follows:

$$\begin{aligned}
\mathcal{Q}(u) - \mathcal{Q}(u_\bullet^{SC}) &= \mathcal{Q}(u - u_\bullet^{SC}) \\
&= \mathcal{B}(u - u_\bullet^{SC}, z) \\
&= \mathcal{B}(u - u_\bullet^{SC}, z - z_\bullet^{SC}) \\
&\leqslant |||u - u_\bullet^{SC}||| \cdot |||z - z_\bullet^{SC}|||.
\end{aligned}$$

There is a significant problem with this argument, however. We recall that our method is a sampling based method, and the final approximation that we generate is an interpolant, not a projection onto some combined finite dimensional space. As a result of this, and in contrast to the stochastic Galerkin framework, the stochastic collocation framework setting lacks global orthogonality. Hence, in general, we have $\mathcal{B}(u - u_\bullet^{SC}, z) \neq \mathcal{B}(u - u_\bullet^{SC}, z - z_\bullet^{SC})$, so that the above argument does not hold. We must therefore seek alternative ways of estimating the error in the quantity of interest.

To gain information about the spatial and parametric components of the error, we decide to assume the splitting that we described in Section 3.5. With the enhanced approximation as defined by 3.59, we recall that this immediately yields:

$$u_\star^{SC} - u_\bullet^{SC} = (\hat{u}_\bullet^{SC} - u_\bullet^{SC}) + (\tilde{u}_\bullet^{SC} - u_\bullet^{SC}) \tag{4.8}$$

We can use the linearity of the goal functional Q to write

$$Q(u_\star^{SC} - u_\bullet^{SC}) = Q(\hat{u}_\bullet^{SC} - u_\bullet^{SC}) + Q(\tilde{u}_\bullet^{SC} - u_\bullet^{SC}) \tag{4.9}$$

so that we have contributions to the error in the goal functional from distinct spatial and parametric components. The first term can be dealt with by observing

$$\begin{aligned}\hat{u}_\bullet^{SC} - u_\bullet^{SC} &= \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \hat{u}_{\bullet\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) - \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \\ &= \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} (\hat{u}_{\bullet\bar{\boldsymbol{y}}}(x) - u_{\bullet\bar{\boldsymbol{y}}}(x)) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}),\end{aligned} \tag{4.10}$$

so that

$$Q(\hat{u}_\bullet^{SC} - u_\bullet^{SC}) = \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \left\{Q(\hat{u}_{\bullet\bar{\boldsymbol{y}}}(x)) - Q(u_{\bullet\bar{\boldsymbol{y}}}(x))\right\} L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}). \tag{4.11}$$

We must now seek an alternative way of dealing with the second term in equation (4.9). Our strategy for doing this involves rewriting the parametric contribution to the total error. After this, we can apply the properties of our goal functional in order to rewrite the parametric contribution so that it is in a form which is conducive to our goal of generating estimates and localised indicators. We address the first of these two tasks in the following lemma.

Lemma 4.1. *Suppose that $u_\bullet^{SC}, \tilde{u}_\bullet^{SC}$ are the approximations given by representations (3.48) and (3.58), respectively. Then we have:*

$$\tilde{u}_\bullet^{SC} - u_\bullet^{SC} = \sum_{\bar{\boldsymbol{y}}\in \widetilde{Y}_\bullet \setminus Y_\bullet} (u_{\bullet\bar{\boldsymbol{y}}}(x) - u_\bullet^{SC}(x,\bar{\boldsymbol{y}})) \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}). \tag{4.12}$$

Proof. We begin by utilising representations (3.58) and (3.48), as well as the fact that that $\widetilde{Y}_\bullet \supset Y_\bullet$ as a result of the underlying multi-index sets $I_\bullet$ and $\widetilde{I}_\bullet$ are monotone. This yields:

$$\begin{aligned}\tilde{u}_\bullet^{SC} - u_\bullet^{SC} &= \sum_{\bar{\boldsymbol{y}}\in \widetilde{Y}_\bullet} \tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x) \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) - \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \\ &= \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x) \Big(\widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) - L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \Big) - \sum_{\bar{\boldsymbol{y}}\in \widetilde{Y}_\bullet \setminus Y_\bullet} \tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}).\end{aligned} \tag{4.13}$$

We now recall that the relevant Lagrange basis functions above satisfy:

$$\begin{aligned} L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') &= \delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'}, \quad \bar{\boldsymbol{y}}, \bar{\boldsymbol{y}}' \in Y_\bullet, \\ \widetilde{L}_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') &= \delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'}, \quad \bar{\boldsymbol{y}}, \bar{\boldsymbol{y}}' \in \widetilde{Y}_\bullet \supset Y_\bullet. \end{aligned}$$

These imply that given any collocation point $\boldsymbol{y}' \in \widetilde{Y}_\bullet$, we have:

$$\begin{aligned}
&\tilde{u}_\bullet^{SC}(x,\bar{\boldsymbol{y}}') - u_\bullet^{SC}(x,\bar{\boldsymbol{y}}') \\
&\quad= \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)\left(\widetilde{L}_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') - L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}')\right) + \sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet} \tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x)\widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \\
&\quad= \begin{cases} \displaystyle\sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)\left(\delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'} - L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}')\right) + \displaystyle\sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet} \tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x)\delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'}, & \bar{\boldsymbol{y}}' \in \widetilde{Y}_\bullet \setminus Y_\bullet \\ \displaystyle\sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)(\delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'} - \delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'}) + \displaystyle\sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet} \tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x)\delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'}, & \bar{\boldsymbol{y}}' \in Y_\bullet \end{cases} \\
&\quad= \begin{cases} -\displaystyle\sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') + \displaystyle\sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet} \tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x)\delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'}, & \bar{\boldsymbol{y}}' \in \widetilde{Y}_\bullet \setminus Y_\bullet \\ 0, & \bar{\boldsymbol{y}}' \in Y_\bullet \end{cases} \\
&\quad= \begin{cases} -u_\bullet^{SC}(x,\bar{\boldsymbol{y}}') + u_{\bullet\bar{\boldsymbol{y}}'}(x), & \bar{\boldsymbol{y}}' \in \widetilde{Y}_\bullet \setminus Y_\bullet \\ 0, & \bar{\boldsymbol{y}}' \in Y_\bullet \end{cases}
\end{aligned}$$

On the other hand, by substituting this back into the form implied by equation (4.13) yields representation (4.12). □

This result simplifies the parametric component of the error for the stochastic collocation FEM approximation into a single sum over the set of new collocation points $\widetilde{Y}_\bullet \setminus Y_\bullet$. While we are not directly interested in the error of the approximation, there is an immediate consequence for the parametric contribution to the error in the goal functional that follows from this.

Corollary 4.2. *With the linear goal functional $Q : V \to \mathbb{R}$, as well as the solutions $u_\bullet^{SC}, \tilde{u}_\bullet^{SC}$ as defined previously, we have:*

$$Q(\tilde{u}_\bullet^{SC} - u_\bullet^{SC}) = \sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet} \left\{Q(u_{\bullet\bar{\boldsymbol{y}}}(x)) - Q(u_\bullet^{SC}(x,\bar{\boldsymbol{y}}))\right\}\widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \tag{4.14}$$

Proof. Via Lemma 4.1, we have:

$$Q(\tilde{u}_\bullet^{SC} - u_\bullet^{SC}) = Q\left(\sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet} (u_{\bullet\bar{\boldsymbol{y}}}(x) - u_\bullet^{SC}(x,\bar{\boldsymbol{y}}))\widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\right).$$

From here, we use the linearity of the goal functional, and the fact that Q is deterministic so that it is independent of the random parameter vector $\boldsymbol{y} \in \Gamma$, and the result follows. □

Theorem 4.3. *Let $\mathcal{Q} : \mathcal{V} \to \mathbb{R}$ be a linear goal functional as previously described, and let $u_\bullet^{SC}$ be the original stochastic collocation FEM approximation given in definition (3.48). Suppose that $\hat{u}_\bullet^{SC}, \tilde{u}_\bullet^{SC}, u_\star^{SC}$ are the enhanced approximations as given in definitions (3.57) – (3.59), and that for some $\varkappa \in (0,1)$, we have*

$$\left|\mathcal{Q}(u) - \mathcal{Q}(u_\star^{SC})\right| \leqslant \varkappa \left|\mathcal{Q}(u) - \mathcal{Q}(u_\bullet^{SC})\right|. \tag{4.15}$$

Then

$$\left|\mathcal{Q}(u) - \mathcal{Q}(u_\bullet^{SC})\right| \leqslant \frac{1}{1-\varkappa}(\mu_\bullet + \tau_\bullet), \tag{4.16}$$

where

$$\mu_\bullet := \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \left\{Q(\hat{u}_{\bullet\bar{\boldsymbol{y}}}(x)) - Q(u_{\bullet\bar{\boldsymbol{y}}}(x))\right\} \int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\, d\rho(\boldsymbol{y}), \tag{4.17}$$

and

$$\tau_\bullet := \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet \setminus Y_\bullet} \left\{Q(u_{\bullet\bar{\boldsymbol{y}}}(x)) - Q(u_\bullet^{SC}(x,\bar{\boldsymbol{y}}))\right\} \int_\Gamma \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\, d\rho(\boldsymbol{y}). \tag{4.18}$$

Proof. We have, via the triangle inequality

$$\left|\mathcal{Q}(u) - \mathcal{Q}(u_\bullet^{SC})\right| \leqslant \left|\mathcal{Q}(u) - \mathcal{Q}(u_\star^{SC})\right| + \left|\mathcal{Q}(u_\star^{SC}) - \mathcal{Q}(u_\bullet^{SC})\right| \tag{4.19}$$

so that using inequality (4.15),

$$\left|\mathcal{Q}(u) - \mathcal{Q}(u_\bullet^{SC})\right| = \frac{1}{1-\varkappa}\left|\mathcal{Q}(u_\star^{SC}) - \mathcal{Q}(u_\bullet^{SC})\right|. \tag{4.20}$$

Then, using the triangle inequality again,

$$\left|\mathcal{Q}(u_\star^{SC}) - \mathcal{Q}(u_\bullet^{SC})\right| \leqslant \left|\mathcal{Q}(\hat{u}_\bullet^{SC}) - \mathcal{Q}(u_\bullet^{SC})\right| + \left|\mathcal{Q}(\tilde{u}_\bullet^{SC}) - \mathcal{Q}(u_\bullet^{SC})\right| \tag{4.21}$$

Using representation (4.11), and then, by observing that the solution samples are deterministic, we have

$$\begin{aligned}
\mathcal{Q}(\hat{u}_\bullet^{SC} - u_\bullet^{SC}) &= \int_\Gamma Q(\hat{u}_\bullet^{SC} - u_\bullet^{SC})\, \mathrm{d}\rho(\boldsymbol{y}) \\
&= \int_\Gamma \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \left\{Q(\hat{u}_{\bullet\bar{\boldsymbol{y}}}(x)) - Q(u_{\bullet\bar{\boldsymbol{y}}}(x))\right\} L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\, \mathrm{d}\rho(\boldsymbol{y}) \\
&= \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \left\{Q(\hat{u}_{\bullet\bar{\boldsymbol{y}}}(x)) - Q(u_{\bullet\bar{\boldsymbol{y}}}(x))\right\} \int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\, \mathrm{d}\rho(\boldsymbol{y})
\end{aligned}$$

We also have via Corollary (4.2),

$$\begin{aligned}
\mathcal{Q}(\tilde{u}_\bullet^{SC} - u_\bullet^{SC}) &= \int_\Gamma Q(\tilde{u}_\bullet^{SC} - u_\bullet^{SC})\, \mathrm{d}\rho(\boldsymbol{y}) \\
&= \int_\Gamma \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet \setminus Y_\bullet} \left\{Q(u_{\bullet\bar{\boldsymbol{y}}}(x)) - Q(u_\bullet^{SC}(x,\bar{\boldsymbol{y}}))\right\} \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\, \mathrm{d}\rho(\boldsymbol{y}) \\
&= \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet \setminus Y_\bullet} \left\{Q(u_{\bullet\bar{\boldsymbol{y}}}(x)) - Q(u_\bullet^{SC}(x,\bar{\boldsymbol{y}}))\right\} \int_\Gamma \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\, \mathrm{d}\rho(\boldsymbol{y})
\end{aligned}$$

Combining these with inequality (4.20) yields the required result. □

We can also produce localised indicators associated with the above results as components of $\mu_\bullet, \tau_\bullet$ that can be calculated quickly. Recalling our problem formulations, it is clear that we can utilise local Galerkin orthogonality, followed by the Cauchy–Schwarz inequality in order to obtain:

$$\begin{aligned} Q(\hat{u}_{\bullet\bar{\boldsymbol{y}}}) - Q(u_{\bullet\bar{\boldsymbol{y}}}) &= B_{\bar{\boldsymbol{y}}}\left(\hat{u}_{\bullet\bar{\boldsymbol{y}}}, \hat{z}_{\bullet\bar{\boldsymbol{y}}}\right) - B_{\bar{\boldsymbol{y}}}\left(u_{\bullet\bar{\boldsymbol{y}}}, \hat{z}_{\bullet\bar{\boldsymbol{y}}}\right) \\ &= B_{\bar{\boldsymbol{y}}}\left(\hat{u}_{\bullet\bar{\boldsymbol{y}}} - u_{\bullet\bar{\boldsymbol{y}}}, \hat{z}_{\bullet\bar{\boldsymbol{y}}}\right) \\ &= B_{\bar{\boldsymbol{y}}}\left(\hat{u}_{\bullet\bar{\boldsymbol{y}}} - u_{\bullet\bar{\boldsymbol{y}}}, \hat{z}_{\bullet\bar{\boldsymbol{y}}} - z_{\bullet\bar{\boldsymbol{y}}}\right) \\ &\leqslant \left|\!\left|\!\left|\hat{u}_{\bullet\bar{\boldsymbol{y}}} - u_{\bullet\bar{\boldsymbol{y}}}\right|\!\right|\!\right|_{\bar{\boldsymbol{y}}} \left|\!\left|\!\left|\hat{z}_{\bullet\bar{\boldsymbol{y}}} - z_{\bullet\bar{\boldsymbol{y}}}\right|\!\right|\!\right|_{\bar{\boldsymbol{y}}} \end{aligned} \tag{4.22}$$

which can then be bounded above by estimates for these quantities. We also note that using the Cauchy–Schwarz inequality, we could also bound the integrals of the Lagrange basis functions $L_{\bar{\boldsymbol{y}}}$, using the elementary result

$$\begin{aligned} \left|\int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y})\right| &\leqslant \left(\int_\Gamma L^2_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y})\right)^{1/2}\left(\int_\Gamma \mathrm{d}\rho(\boldsymbol{y})\right)^{1/2} \\ &= \left(\int_\Gamma L^2_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y})\right)^{1/2} \\ &= \left\|L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\right\|_\Gamma \end{aligned} \tag{4.23}$$

although this isn't strictly necessary. If we utilise all of these results together, however, our outcome is an upper bound of the form

$$\mathcal{Q}(\hat{u}^{SC}_\bullet - u^{SC}_\bullet) \leqslant \sum_{\bar{\boldsymbol{y}}\in Y_\bullet}\left(\sum_{K\in\mathcal{T}_\bullet}\mu^2_{\bullet\bar{\boldsymbol{y}},K}\|L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\|_\Gamma\right)^{1/2}\left(\sum_{K\in\mathcal{T}_\bullet}\mu^2_{\bullet\bar{\boldsymbol{y}},K}\|L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\|_\Gamma\right)^{1/2} \tag{4.24}$$

where $\mu_{\bullet\bar{\boldsymbol{y}},K}, \mu_{\bullet\bar{\boldsymbol{y}},K}$ are local error indicators (for example, two-level error indicators) that can be used for single-level or multi-level routines. For the parametric component, we have, via Corollary 4.2:

$$\begin{aligned} Q(\tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x)) - Q(u^{SC}_\bullet(x,\bar{\boldsymbol{y}})) &= Q(\tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x) - u^{SC}_\bullet(x,\bar{\boldsymbol{y}})) \\ &= B_{\bar{\boldsymbol{y}}}(\tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x) - u^{SC}_\bullet(x,\bar{\boldsymbol{y}}), \tilde{z}_{\bullet\bar{\boldsymbol{y}}}(x)) \\ &\leqslant \left|\!\left|\!\left|\tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x) - u^{SC}_\bullet(x,\bar{\boldsymbol{y}})\right|\!\right|\!\right|_{\bar{\boldsymbol{y}}} \left|\!\left|\!\left|\tilde{z}_{\bullet\bar{\boldsymbol{y}}}(x)\right|\!\right|\!\right|_{\bar{\boldsymbol{y}}} \end{aligned} \tag{4.25}$$

so that

$$\mathcal{Q}(\tilde{u}^{SC}_\bullet - u^{SC}_\bullet) \leqslant \sum_{\boldsymbol{i}\in\widetilde{I}_\bullet\setminus I_\bullet}\sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_{\bullet\boldsymbol{i}}}\tau_{\bullet\bar{\boldsymbol{y}}}\sigma_{\bullet\bar{\boldsymbol{y}}} \tag{4.26}$$

with

$$\tau_{\bullet\bar{\boldsymbol{y}}} := \left|\!\left|\!\left| \tilde{u}_{\bullet\bar{\boldsymbol{y}}}(x) - u_{\bullet}^{SC}(x,\bar{\boldsymbol{y}}) \right|\!\right|\!\right|_{\bar{\boldsymbol{y}}} \left\| \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \right\|_{\Gamma}^{1/2} \tag{4.27}$$

and

$$\sigma_{\bullet\bar{\boldsymbol{y}}} := \left|\!\left|\!\left| \tilde{z}_{\bullet\bar{\boldsymbol{y}}}(x) \right|\!\right|\!\right|_{\bar{\boldsymbol{y}}} \left\| \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \right\|_{\Gamma}^{1/2} \tag{4.28}$$

and we are therefore able to retrieve local indicators associated with multi-indices that contribute to the set of collocation points in our parameter domain.

Despite this apparent success, there are a few significant problems with the results given above. Firstly, while Theorem 4.3 reveals a certain amount of information about the 'error reduction', it does not yield any further information unless inequality (4.15) holds. This can be thought of as a modified saturation assumption for the goal functional, and is likely to be even more contentious than the saturation assumption in the standard setting.

Furthermore, the crude upper-bound embedded within inequality (4.25) has an even more undesirable effect that will inevitably appear in the final result. The underlying idea of the deterministic goal-oriented setting is that there is a 'symmetry' in the final result with respect to the primal and dual problems. In particular, we ended up with a product of two norms that are dependent on some form of solution differences. As each of these norms converge at a certain rate, the rate of convergence for the quantity of interest is then at twice the order of the solution itself.

The parametric component of this estimate does not have this symmetry property, and it instead includes the norm of an enhanced dual solution by itself, which does not provide any benefits for the purposes of convergence rates. This result motivates the use of an alternative approach in an attempt to induce some form of symmetry in the final estimates.

4.3 The corrected goal functional

If we attempt to proceed using the goal-oriented approach in the deterministic setting, we use the weak formulation (3.7) and the linearity of $\mathcal{B}$ to obtain

$$\begin{aligned}\mathcal{Q}(u) - \mathcal{Q}(u_\bullet^{SC}) &= \mathcal{Q}(u - u_\bullet^{SC}) \\ &= \mathcal{B}(u - u_\bullet^{SC}, z) \\ &= \mathcal{B}(u - u_\bullet^{SC}, z - z_\bullet^{SC}) + \mathcal{B}(u - u_\bullet^{SC}, z_\bullet^{SC}).\end{aligned}$$

Now, while a lack of global Galerkin orthogonality means that $\mathcal{B}(u - u_\bullet^{SC}, z_\bullet^{SC}) \neq 0$, we can at least find alternative ways of dealing with this bilinear form. We proceed by incorporating this bilinear form into the quantity of interest as a 'correction term' which compensates for the lack of orthogonality in the stochastic collocation setting. A similar approach based on the same principle has been previously been utilised in [58] for the purposes of estimating integral functionals in the deterministic setting. We define a new 'corrected' quantity of interest:

$$\bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) := \mathcal{Q}(u_\bullet^{SC}) + \mathcal{B}(u - u_\bullet^{SC}, z_\bullet^{SC}), \tag{4.29}$$

Re-writing this using the linearity of $\mathcal{B}$, as well as problem (3.7), we have

$$\begin{aligned}\bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) &:= \mathcal{Q}(u_\bullet^{SC}) + \mathcal{B}(u, z_\bullet^{SC}) - \mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}) \\ &= \mathcal{Q}(u_\bullet^{SC}) + \mathcal{F}(z_\bullet^{SC}) - \mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}).\end{aligned} \tag{4.30}$$

Note that with the above simplifications, this 'corrected' goal functional is computable. In fact, we can simplify this even further using a result that we give below.

Lemma 4.4. *Let $\mathcal{F}, \mathcal{Q} : \mathcal{V} \to \mathbb{R}$ be the right-hand side and goal functional quantities given in problems (3.7) and (4.2), with $u_\bullet^{SC}, z_\bullet^{SC}$ being the stochastic collocation FEM approximations to each of these problems, respectively. Then $\mathcal{Q}(u_\bullet^{SC}) = \mathcal{F}(z_\bullet^{SC})$.*

Proof. Using the dual and primal formulations (4.6) and (3.47), we deduce that

$$Q(u_{\bullet\bar{\boldsymbol{y}}}) = B_{\bar{\boldsymbol{y}}}(u_{\bullet\bar{\boldsymbol{y}}}, z_{\bullet\bar{\boldsymbol{y}}}) = F(z_{\bullet\bar{\boldsymbol{y}}}). \tag{4.31}$$

We then use the definitions of $u_\bullet^{SC}, z_\bullet^{SC}$, as well as the fact that using the fact that F, Q are

deterministic to deduce

$$\begin{aligned}\mathcal{Q}(u_\bullet^{SC}) &= \int_\Gamma Q(u_\bullet^{SC})\,\mathrm{d}\rho(\boldsymbol{y}) \\ &= \int_\Gamma Q\left(\sum_{\bar{\boldsymbol{y}}\in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x)L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\right)\,\mathrm{d}\rho(\boldsymbol{y}) \\ &= \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} Q(u_{\bullet\bar{\boldsymbol{y}}}(x))\int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y})\end{aligned}$$

and

$$\begin{aligned}\mathcal{F}(z_\bullet^{SC}) &= \int_\Gamma F(z_\bullet^{SC})\,\mathrm{d}\rho(\boldsymbol{y}) \\ &= \int_\Gamma F\left(\sum_{\bar{\boldsymbol{y}}\in Y_\bullet} z_{\bullet\bar{\boldsymbol{y}}}(x)L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\right)\,\mathrm{d}\rho(\boldsymbol{y}) \\ &= \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} F(z_{\bullet\bar{\boldsymbol{y}}}(x))\int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y})\end{aligned}$$

Using equation (4.31), we conclude

$$\mathcal{Q}(u_\bullet^{SC}) = \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} B_{\bar{\boldsymbol{y}}}(u_{\bullet\bar{\boldsymbol{y}}}(x), z_{\bullet\bar{\boldsymbol{y}}}(x))\int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y}) = \mathcal{F}(z_\bullet^{SC}) \tag{4.32}$$

as required. □

With regards to the above lemma, we remark that the lack of orthogonality means that the expression given in equation (4.32) is not equal to $\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC})$. If it were, the above calculation would both be elementary, and in any case, make the correction term redundant, a fact which will become relevant when discussing how to calculate the correction term. Our 'corrected' goal functional can now be represented by

$$\bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) = 2\mathcal{Q}(u_\bullet^{SC}) - \mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}). \tag{4.33}$$

In addition to this, we have

$$\begin{aligned}\mathcal{Q}(u) - \bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) &= \mathcal{B}(u - u_\bullet^{SC}, z - z_\bullet^{SC}) \\ &\leqslant |||u - u_\bullet^{SC}||| \cdot |||z - z_\bullet^{SC}||| \\ &\lesssim \|u - u_\bullet^{SC}\|_{\mathcal{V}} \cdot \|z - z_\bullet^{SC}\|_{\mathcal{V}} \\ &\lesssim (\mu_\bullet + \tau_\bullet)(\eta_\bullet + \sigma_\bullet)\end{aligned} \tag{4.34}$$

for some error estimates $\mu_\bullet, \tau_\bullet, \eta_\bullet, \sigma_\bullet$ that we recall from the standard stochastic collocation FEM framework (such as Theorem 3.4, with $\eta_\bullet, \sigma_\bullet$ representing spatial and parametric error estimates obtained from applying the same result to the dual problem).

We should emphasise that now, with this approach, the quantity $\mathcal{Q}(u_\bullet^{SC})$ is no longer the estimate that we will use for the quantity of the interest. Instead, what we have deduced is that the approximation of the quantity of interest $\mathcal{Q}(u)$, is best done using the corrected goal functional $\bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC})$, which depends on both the primal and dual responses by virtue of the bi-linear form, $\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC})$. Further, we note that in the context of an adaptive algorithm, where we have continual refinement of the finite dimensional spaces, $u \to u_\bullet^{SC}$, so that $\mathcal{B}(u - u_\bullet^{SC}, z_\bullet^{SC}) \to 0$. This being said, we do not know how fast this convergence occurs in comparison to the rate at which $\bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC})$ converges to $\mathcal{Q}(u)$.

4.4 Calculation of the correction term

In calculating the corrected goal functional, there are now two components that we are required to compute – namely, the goal functional $\mathcal{Q}(u_\bullet^{SC})$, and the bilinear form $\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC})$. The calculation of the goal functional is performed easily enough using equation (4.32). In particular, this functional has been split into a sum of products of deterministic bilinear forms and integrals of Lagrange basis functions. The deterministic bilinear forms can be evaluated using the stiffness matrices associated with the sampled bi-linear forms at each $\boldsymbol{y} \in Y_\bullet$, while the stochastic integrals can easily be directly calculated after using the underlying setup described in Theorem 3.2 by by evaluating products of appropriate 1D Lagrange basis functions.

Calculating the bilinear form is more difficult, as it does not necessarily involve a sum of separable products. Indeed, by expanding the stochastic collocation FEM approximations, we obtain:

$$\begin{aligned}\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}) &= \int_\Gamma \int_D a(x,\boldsymbol{y}) \nabla u_\bullet^{SC}(x,\boldsymbol{y}) \cdot \nabla z_\bullet^{SC}(x,\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \\ &= \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}' \in Y_\bullet} \int_\Gamma \int_D a(x,\boldsymbol{y}) \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot \nabla z_{\bullet\bar{\boldsymbol{y}}'}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}).\end{aligned}$$

The diffusion coefficient's dependence on both spatial and random parameters proves to be a prohibitive obstacle, and it transpires that writing each term in the above sum as a product of spatial and parametric contributions is only possible for specific cases of the diffusion coefficient.

Suppose that our diffusion coefficient $a(x,\boldsymbol{y})$ consists of an affine expansion of random

parameters, as in equation (3.69). Using this expansion of the diffusion coefficient, and by expanding the stochastic collocation FEM approximations, we obtain

$$\begin{aligned}\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}) &= \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'\in Y_\bullet} \int_\Gamma \int_D a(x,\boldsymbol{y}) \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot \nabla z_{\bullet\bar{\boldsymbol{y}}'}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \\ &= \sum_{m=0}^{M} \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'\in Y_\bullet} \int_\Gamma \int_D y_m a_m(x) \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot \nabla z_{\bullet\bar{\boldsymbol{y}}'}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \\ &= \sum_{m=0}^{M} \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'\in Y_\bullet} \left(\int_D a_m(x) u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot z_{\bullet\bar{\boldsymbol{y}}'}(x) \,\mathrm{d}x \right) \left(\int_\Gamma y_m L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}) \right).\end{aligned}$$

where we define $y_0 := 1$. From here, we can calculate the spatial integrals using the standard procedure involving stiffness matrices, as we did for $\mathcal{Q}(u_\bullet^{SC})$. The parametric integrals, while a little more complicated than for the goal functional, are still computable using pre-computed tensors involving weighted products of integrals involving Lagrange basis functions in a single dimension.

In a similar manner to the linear case, we can assume a quadratic diffusion coefficient of the form given in representation (3.72), which we can expand out as (again, defining $y_0 := 1$):

$$a(x,\boldsymbol{y}) = \sum_{m,l=0}^{M} y_m y_l a_m(x) a_l(x), \tag{4.35}$$

From this, we may establish that

$$\begin{aligned}\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}) &= \sum_{m,l=0}^{M} \int_\Gamma \int_D y_m y_l a_m(x) a_l(x) \nabla u_\bullet^{SC}(x,\boldsymbol{y}) \cdot \nabla z_\bullet^{SC}(x,\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \\ &= \sum_{m,l=0}^{M} \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'\in Y_\bullet} \int_\Gamma \int_D y_m y_l a_m(x) a_l(x) \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot z_{\bullet\bar{\boldsymbol{y}}'}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \\ &= \sum_{m,l=0}^{M} \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'\in Y_\bullet} B_{m,l}(u_{\bullet\bar{\boldsymbol{y}}}, z_{\bullet\bar{\boldsymbol{y}}'}) \left(\int_\Gamma y_m y_l L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}) \right)\end{aligned}$$

where

$$B_{m,l}(u_{\bullet\bar{\boldsymbol{y}}}, z_{\bullet\bar{\boldsymbol{y}}'}) := \int_D a_m(x) a_l(x) \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot \nabla z_{\bullet\bar{\boldsymbol{y}}'}(x) \,\mathrm{d}x \tag{4.36}$$

We also note that for practical purposes, rather than using equation (4.35), we may calculate the above using the diffusion coefficient in the form

$$a(x,\boldsymbol{y}) = \sum_{m=1}^{M} y_m a_m(x) \left(y_m a_m(x) + 2 \sum_{l=0}^{m-1} y_l a_l(x) \right) \tag{4.37}$$

in order to offer a small optimisation in terms of computational expense. Theoretically, exact representations of the element residual norm exist for any diffusion coefficient that has

polynomial expansion in terms of random parameters given by:

$$a(x,\boldsymbol{y}) = \left(a_0(x) + \sum_{m=1}^{M} y_m a_m(x)\right)^n, \quad n \in \mathbb{N}$$

or any similar variation thereof. In practice, however, calculating these quantities becomes computationally expensive very quickly, and so, even for higher order polynomial expansions, it may be prudent to seek an alternative, inexact representation of our required bilinear form. Nevertheless, we note that the above result is useful as it gives us a non-affine example that can be computed exactly. The result of this can then be used for comparison purposes, with regards to both computational expense and accuracy.

In addition to higher order polynomials, we may wish to consider other non-affine expansions of the diffusion coefficient. For these purposes we consider a quadrature rule. We begin by observing that, using a quadrature rule with $Y_\bullet$, we obtain the approximation:

$$\begin{aligned}
\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}) &= \int_\Gamma \int_D a(x,\boldsymbol{y}) \nabla u_\bullet^{SC}(x,\boldsymbol{y}) \cdot \nabla z_\bullet^{SC}(x,\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \\
&= \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}' \in Y_\bullet} \int_\Gamma \int_D a(x,\boldsymbol{y}) \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot \nabla z_{\bullet\bar{\boldsymbol{y}}'}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \\
&\approx \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}',\bar{\boldsymbol{y}}'' \in Y_\bullet} \omega_{\bar{\boldsymbol{y}}''} \int_D a(x,\bar{\boldsymbol{y}}'') \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot \nabla z_{\bullet\bar{\boldsymbol{y}}'}(x) L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}'') L_{\bar{\boldsymbol{y}}'}(\bar{\boldsymbol{y}}'') \,\mathrm{d}x \\
&= \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}',\bar{\boldsymbol{y}}'' \in Y_\bullet} \omega_{\bar{\boldsymbol{y}}''} \int_D a(x,\bar{\boldsymbol{y}}'') \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot \nabla z_{\bullet\bar{\boldsymbol{y}}'}(x) \delta_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}''} \delta_{\bar{\boldsymbol{y}}'\bar{\boldsymbol{y}}''} \,\mathrm{d}x \\
&= \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \omega_{\bar{\boldsymbol{y}}} \int_D a(x,\bar{\boldsymbol{y}}) \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot \nabla z_{\bullet\bar{\boldsymbol{y}}}(x) \,\mathrm{d}x
\end{aligned}$$

where deducing the associated quadrature weights, $\omega_{\bar{\boldsymbol{y}}}$, is simply a matter of recognising that we have a calculation of the form:

$$\begin{aligned}
\int_\Gamma W(x,\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}) &= \int_\Gamma \left(\sum_{\bar{\boldsymbol{y}} \in Y_\bullet} L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) W(x,\bar{\boldsymbol{y}}) \right) \mathrm{d}\rho(\boldsymbol{y}) \\
&= \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} W(x,\bar{\boldsymbol{y}}) \int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}),
\end{aligned}$$

which yields weights

$$\omega_{\bar{\boldsymbol{y}}} = \int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}).$$

This, in turn gives us the result

$$\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}) \approx \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} B_{\bar{\boldsymbol{y}}}(u_{\bullet\bar{\boldsymbol{y}}}(x), z_{\bullet\bar{\boldsymbol{y}}}(x)) \int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}). \tag{4.38}$$

Recalling equation (4.32), this implies

$$\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}) \approx \mathcal{F}(z_\bullet^{SC}) = \mathcal{Q}(u_\bullet^{SC}),$$

so that

$$\bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) \approx \mathcal{Q}(u_\bullet^{SC}), \tag{4.39}$$

which is hardly surprising, but nevertheless makes our use of the correction term redundant. We must therefore use an enhanced collocation point set $\widetilde{Y}_\bullet \supset Y_\bullet$. If we do this, we obtain the result

$$\begin{aligned}
\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}) &= \int_\Gamma \int_D a(x,\boldsymbol{y}) \nabla u_\bullet^{SC}(x,\boldsymbol{y}) \cdot \nabla z_\bullet^{SC}(x,\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \\
&= \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}' \in Y_\bullet} \int_\Gamma \int_D a(x,\boldsymbol{y}) \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot \nabla z_{\bullet\bar{\boldsymbol{y}}'}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \\
&\approx \sum_{\bar{\boldsymbol{y}}'' \in \widetilde{Y}_\bullet} \omega_{\bar{\boldsymbol{y}}''} \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}' \in Y_\bullet} \int_D a(x,\bar{\boldsymbol{y}}'') \nabla u_{\bullet\bar{\boldsymbol{y}}}(x) \cdot \nabla z_{\bullet\bar{\boldsymbol{y}}'}(x) L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}'') L_{\bar{\boldsymbol{y}}'}(\bar{\boldsymbol{y}}'') \,\mathrm{d}x \\
&= \left(\sum_{\bar{\boldsymbol{y}}'' \in Y_\bullet} + \sum_{\bar{\boldsymbol{y}}'' \in \widetilde{Y}_\bullet \setminus Y_\bullet} \right) \omega_{\bar{\boldsymbol{y}}''} \sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}' \in Y_\bullet} B_{\bar{\boldsymbol{y}}''}(u_{\bullet\bar{\boldsymbol{y}}}(x), z_{\bullet\bar{\boldsymbol{y}}'}(x)) L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}'') L_{\bar{\boldsymbol{y}}'}(\bar{\boldsymbol{y}}'') \\
&= \sum_{\bar{\boldsymbol{y}}'' \in \widetilde{Y}_\bullet \setminus Y_\bullet} \omega_{\bar{\boldsymbol{y}}''} \, \widetilde{B}_{\bar{\boldsymbol{y}}''}(u_\bullet^{SC}, z_\bullet^{SC})
\end{aligned}$$

with

$$\widetilde{B}_{\bar{\boldsymbol{y}}''}(u_\bullet^{SC}, z_\bullet^{SC}) := \begin{cases} B_{\bar{\boldsymbol{y}}''}(u_{\bullet\bar{\boldsymbol{y}}''}, z_{\bullet\bar{\boldsymbol{y}}''}), & \bar{\boldsymbol{y}}'' \in Y_\bullet, \\ \displaystyle\sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}' \in Y_\bullet} B_{\bar{\boldsymbol{y}}''}(u_{\bullet\bar{\boldsymbol{y}}}, z_{\bullet\bar{\boldsymbol{y}}'}) L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}'') L_{\bar{\boldsymbol{y}}'}(\bar{\boldsymbol{y}}''), & \bar{\boldsymbol{y}}'' \in \widetilde{Y}_\bullet \setminus Y_\bullet. \end{cases}$$

and the associated quadrature weights

$$\omega_{\bar{\boldsymbol{y}}''} = \int_\Gamma \widetilde{L}_{\bar{\boldsymbol{y}}''}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}).$$

This means that we have now generated an approximation

$$\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC}) \approx \sum_{\bar{\boldsymbol{y}}'' \in \widetilde{Y}_\bullet \setminus Y_\bullet} \widetilde{B}_{\bar{\boldsymbol{y}}''}(u_\bullet^{SC}, z_\bullet^{SC}) \int_\Gamma \widetilde{L}_{\bar{\boldsymbol{y}}''}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}) \tag{4.40}$$

which will therefore induce a more appropriate approximation of the corrected goal functional (4.33).

We can see that the computation of this bilinear form is expensive in comparison to the uncorrected goal functional, which is cheap due to the fact that we already obtain most of the required quantities from other parts of the adaptive algorithm. Despite this, the additional

computational expense associated with calculating the bilinear form is not prohibitive, as the correction term forms part of the quantity we are estimating, and not the estimates and indicators that are being computed at each iteration. As a result of this, the bilinear form computation only needs to be done when we need to estimate the goal functional, and outside of demonstrative examples, this will typically be at the end of the adaptive algorithm, and not during each iteration of the adaptive loop.

4.5 Adaptive algorithm

Having discussed the key aspect of estimating the error in the quantity of interest, we wish to examine how this functions in the context of an adaptive algorithm. In particular, we must examine what changes we need to make from the standard framework in order to obtain an approximation for the quantity of interest with an estimated error which is less than a given tolerance ε.

The solution step is conducted in the same way as we discussed in Chapter 3, with the difference being that we must solve two problems at each collocation point for any given iteration. Specifically, we solve problems (3.56) and (4.6) for each $\bar{\boldsymbol{y}} \in Y_\bullet$ at each iteration.

By inequality (4.34), we know that to estimate the error in the quantity of interest, we need error estimates associated with the primal and dual problems. These can, in principle, be computed in exactly the same way as we have done previously. With two sets of solutions and error estimates to compute, the computational time required for these steps of our goal-oriented algorithm will be doubled when compared with non-goal-oriented strategies. Despite this, in a similar manner to the deterministic setting, we can also reasonably expect that the quantity of interest will converge at up to twice the order, which will make this procedure a worthwhile trade-off.

In the context of our adaptive algorithm, the marking stage will need to accomplish two key aims. Firstly, we need to select whether we are proceeding with spatial or parametric refinement at a given iteration, and secondly, we must obtain a single marked set that is associated with that refinement type.

The first of these objectives was trivial in the standard setting. Indeed, we only had 2 sets of indicators, one of which could be assembled into an indirect spatial error estimate, with the other being assembled into an indirect parametric estimate. The choice of refinement type was then determined by whichever of the two indirect estimates was largest. Our problem in the goal-oriented setting is that our error estimation step provides us with four distinct sets of indicators, as we now have spatial and parametric indicator sets corresponding to both the primal and dual problems. We must therefore identify a way of successfully combining these to identify the correct refinement type.

Suppose that $\bar{\mu}_\bullet$ and $\bar{\eta}_\bullet$ represent indirect spatial estimates for the primal and dual problems, respectively. Suppose also that $\bar{\tau}_\bullet$ and $\bar{\sigma}_\bullet$ represent similar parametric estimates. We propose three different rules for determining the refinement type, each of which has their own motivating principle.

The first option is a simple maximum rule, where we simply refine based on the largest quantity of the four quantities, so that our refinement type is dictated by $\max\{\bar{\mu}_\bullet, \bar{\eta}_\bullet, \bar{\tau}_\bullet, \bar{\sigma}_\bullet\}$. Specifically, we can define our refinement type as a boolean, $\mathbb{1}_{\mathcal{M}}$, satisfying:

$$\mathbb{1}_{\mathcal{M}} := \begin{cases} 1, & \max\{\bar{\mu}_\bullet, \bar{\eta}_\bullet, \bar{\tau}_\bullet, \bar{\sigma}_\bullet\} \in \{\bar{\tau}_\bullet, \bar{\sigma}_\bullet\}, \\ 0, & \max\{\bar{\mu}_\bullet, \bar{\eta}_\bullet, \bar{\tau}_\bullet, \bar{\sigma}_\bullet\} \notin \{\bar{\tau}_\bullet, \bar{\sigma}_\bullet\}, \end{cases} \tag{4.41}$$

with an output of 1 implying parametric refinement and an output of 0 implying spatial refinement. The motivation for this approach is that at the very least, it guarantees the refinement of whichever of the four possible sets of indicators admits the largest errors. That being said, it ignores the fact that, for example, we may have a situation where $\bar{\mu}_\bullet > \bar{\tau}_\bullet \approx \bar{\sigma}_\bullet \gg \bar{\eta}_\bullet$, and this approach will select spatial refinement, where parametric refinement may have been more preferable.

As a potential alternative, we see that on expanding inequality (4.34), the products $\mu_\bullet\, \eta_\bullet$ and $\tau_\bullet\, \sigma_\bullet$ represent the most significant spatial and parametric contributions to the estimated error in the quantity of interest. Motivated by this observation, we suggest a modification of definition (4.41), so that the choice of refinement type is determined by the expression $\max\{\bar{\mu}_\bullet\bar{\eta}_\bullet, \bar{\tau}_\bullet\bar{\sigma}_\bullet\}$. This rule naturally compensates for the issue associated with the maximum rule; however, we obtain other difficulties when, for example, one of the four indirect estimates becomes close to zero much faster than the other three.

We therefore propose one final alternative that does have a clear theoretical motivation. We recall that the error in the goal functional is bounded above using inequality (4.34). By expanding this product, we obtain the obvious result:

$$\begin{aligned}(\mu_\bullet + \tau_\bullet)(\eta_\bullet + \sigma_\bullet) &\leqslant (\bar{\mu}_\bullet + \bar{\tau}_\bullet)(\bar{\eta}_\bullet + \bar{\sigma}_\bullet) \\ &\leqslant \frac{1}{2}\Big((\bar{\mu}_\bullet + \bar{\tau}_\bullet)^2 + (\bar{\eta}_\bullet + \bar{\sigma}_\bullet)^2\Big) \\ &\leqslant (\bar{\mu}_\bullet^2 + \bar{\eta}_\bullet^2) + (\bar{\tau}_\bullet^2 + \bar{\sigma}_\bullet^2),\end{aligned} \tag{4.42}$$

from which we infer a criterion based on the expression $\max\{\bar{\mu}_\bullet^2 + \bar{\eta}_\bullet^2, \bar{\tau}_\bullet^2 + \bar{\sigma}_\bullet^2\}$. The incentive for this 'sum-of-squares' rule is clear, although the bounding above of our original product estimate does mean that the relationship between that estimate and the criterion we suggest here is a little looser.

Of course, other approaches are available, including other combinations of the four indicators, or even, hybrid approaches. For example, one could choose the refinement type based on whether a spatial or parametric argument is maximised in at least two of the three aforementioned rules.

Regardless of this, it is evident that we can find a way of deciding whether we wish to proceed with spatial or parametric refinement. As we are only focusing on refinement relating to a source of errors that pertains to two sets of indicators, we can use the strategies that we employed in Section 3.5.2 to obtain two marked sets corresponding to the type of refinement that has been selected. These two sets will correspond to the primal and dual problems, respectively.

We can then follow one of the approaches suggested in Section 2.8, utilising a simple union, an aggregation-based rule, or a minimum cardinality rule in order to obtain a single marked set for the purposes of refinement, corresponding to either the spatial domain or the parameter domain. A possible marking strategy resulting from the above ideas is summarised in Algorithm 4.1. We recall our remark from Section 3.5.1 that we are primarily describing the marking of elements for the purposes of this theoretical work, although this can also easily be applied to the marking of edges using edge-based indicators.

Having now completed our objectives of choosing a refinement type and selecting a suitable marking set, we move to the refinement step, which functions in exactly the same way as the

Algorithm 4.1: Marking algorithm for goal-oriented stochastic collocation FEM

Data: Spatial indicators, $\mu_{\bullet\bar{\boldsymbol{y}},K}$ (primal) and $\eta_{\bullet\bar{\boldsymbol{y}},K}$ (dual); parametric indicators, $\tau_{\bullet\,\boldsymbol{i}}$ (primal) and $\sigma_{\bullet\,\boldsymbol{i}}$ (dual); marking parameters, $0 < \theta_x, \theta_{\boldsymbol{y}} < 1$.

Result: A choice of spatial or parametric refinement; a single marked set, either $\widehat{\mathcal{M}}_\bullet$ (spatial) or $\widetilde{\mathcal{M}}_\bullet$ (parametric).

Use indicators to calculate indirect estimates $\bar{\mu}_\bullet, \bar{\eta}_\bullet, \bar{\tau}_\bullet, \bar{\sigma}_\bullet$.

Determine boolean, $\mathbb{1}_{\mathcal{M}}$, corresponding to spatial (0) or parametric (1) refinement.

if $\mathbb{1}_{\mathcal{M}} = 1$ **then**

Determine marked sets $\widetilde{\mathcal{M}}_{\bullet u}, \widetilde{\mathcal{M}}_{\bullet z}$, of minimum cardinality, such that

$$\theta_{\boldsymbol{y}} \sum_{\boldsymbol{i}\in\mathcal{R}_{\boldsymbol{i}}} \tau_{\bullet\,\boldsymbol{i}} \|L_{\bar{\boldsymbol{y}}}\|_\Gamma \leqslant \sum_{\boldsymbol{i}\in\widetilde{\mathcal{M}}_{\bullet u}} \tau_{\bullet\,\boldsymbol{i}} \|L_{\bar{\boldsymbol{y}}}\|_\Gamma, \quad \theta_{\boldsymbol{y}} \sum_{\boldsymbol{i}\in\mathcal{R}_{\boldsymbol{i}}} \sigma_{\bullet\,\boldsymbol{i}} \|L_{\bar{\boldsymbol{y}}}\|_\Gamma \leqslant \sum_{\boldsymbol{i}\in\widetilde{\mathcal{M}}_{\bullet z}} \sigma_{\bullet\,\boldsymbol{i}} \|L_{\bar{\boldsymbol{y}}}\|_\Gamma$$

Define $\widetilde{\mathcal{M}}_\bullet := \widetilde{\mathcal{M}}_{\bullet u} \cup \widetilde{\mathcal{M}}_{\bullet z}$ (or use a similar criterion to combine the sets).

else

For each $\boldsymbol{y} \in Y_\bullet$, determine marked sets $\widehat{\mathcal{M}}_{\bullet u}, \widehat{\mathcal{M}}_{\bullet z}$, of minimum cardinality, such that:

$$\theta_x \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \sum_{K\in\mathcal{T}_\bullet} \mu_{\bullet\bar{\boldsymbol{y}},K} \|L_{\bar{\boldsymbol{y}}}\|_\Gamma \leqslant \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \sum_{K\in\widehat{\mathcal{M}}_{\bullet\bar{\boldsymbol{y}}u}} \mu_{\bullet\bar{\boldsymbol{y}},K} \|L_{\bar{\boldsymbol{y}}}\|_\Gamma \,,$$

$$\theta_x \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \sum_{K\in\mathcal{T}_\bullet} \mu_{\bullet\bar{\boldsymbol{y}},K} \|L_{\bar{\boldsymbol{y}}}\|_\Gamma \leqslant \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \sum_{K\in\widehat{\mathcal{M}}_{\bullet\bar{\boldsymbol{y}}z}} \eta_{\bullet\bar{\boldsymbol{y}},K} \|L_{\bar{\boldsymbol{y}}}\|_\Gamma$$

Define $\widehat{\mathcal{M}}_{\bullet u} := \bigcup_{\bar{\boldsymbol{y}}\in Y_\bullet} \widehat{\mathcal{M}}_{\bullet\bar{\boldsymbol{y}}u}$ and $\widehat{\mathcal{M}}_{\bullet z} := \bigcup_{\bar{\boldsymbol{y}}\in Y_\bullet} \widehat{\mathcal{M}}_{\bullet\bar{\boldsymbol{y}}z}$.

Define $\widehat{\mathcal{M}}_\bullet := \widehat{\mathcal{M}}_{\bullet u} \cup \widehat{\mathcal{M}}_{\bullet z}$ (or use a similar criterion to combine the sets).

end

Algorithm 4.2: Goal-oriented Stochastic Collocation FEM algorithm

Data: Initial spatial mesh, $\mathcal{T}_0$, initial index set $I_0 = \{\mathbf{1}\}$; error tolerance $\texttt{tol} = \varepsilon$; problem parameters, including marking criteria and a refinement procedure.

Result: Goal functional $\bar{\mathcal{Q}}(u_*^{SC}, z_*^{SC}) \approx \mathcal{Q}(u)$; corresponding error estimates satisfying $(\mu_* + \tau_*)(\eta_* + \sigma_*) < \varepsilon$.

Set iteration counter $k = 0$; iteration factor $l \in \mathbb{N}$.

while $(\mu_l + \tau_l)(\eta_l + \sigma_l) \geqslant \varepsilon$ **do**

Obtain the approximations, $u_{k\bar{\boldsymbol{y}}}, z_{k\bar{\boldsymbol{y}}} \in V_k$ by solving problems (3.56), (4.6) for each $\bar{\boldsymbol{y}} \in Y_k$.

For each $\bar{\boldsymbol{y}} \in Y_k, K \in \mathcal{T}_k$, compute spatial error indicators $\mu_{k\bar{\boldsymbol{y}},K}, \eta_{k\bar{\boldsymbol{y}},K}$ and for each $\boldsymbol{i} \in \widetilde{I}_k \setminus I_k$, compute parametric error indicators $\tau_{k\,\boldsymbol{i}}, \sigma_{k\,\boldsymbol{i}}$.

Use the indicators to determine indirect estimators $\bar{\mu_k}, \bar{\eta_k}, \bar{\tau_k}, \bar{\sigma_k}$.

Use the indirect estimators to determine the refinement type boolean $\mathbb{1}_{\mathcal{M}}$.

For each $\bar{\boldsymbol{y}} \in Y_k$, determine the marked spatial set $\widehat{\mathcal{M}}_k := \bigcup_{\bar{\boldsymbol{y}} \in Y_k} \widehat{\mathcal{M}}_{k,\bar{\boldsymbol{y}}}$ and/or the marked index set $\widetilde{\mathcal{M}}_k$.

if $k = jl,\ j \in \mathbb{N}$ **then**

Compute the direct error estimates μ_l, τ_l.

end

if $(\mu_l + \tau_l)(\eta_l + \sigma_l) \geqslant \varepsilon$ **then**

if $\mathbb{1}_{\mathcal{M}} = 1$ **then**

Define $I_{k+1} := \texttt{refine}\ (I_k, \widetilde{\mathcal{M}}_k)$, and set $\mathcal{T}_{k+1} := \mathcal{T}_k$.

else

Define $\mathcal{T}_{k+1} := \texttt{refine}\ (\mathcal{T}_k, \widehat{\mathcal{M}}_k)$ and set $I_{k+1} := I_k$.

end

Set $k \to k + 1$.;

end

end

Set $u_*^{SC} := u_l^{SC}, z_*^{SC} := z_l^{SC}, \mu_* := \mu_l, \eta_* := \eta_l, \tau_* := \tau_l, \sigma_* := \sigma_l$.

Calculate the goal functional $\bar{\mathcal{Q}}(u_*^{SC}, z_*^{SC})$.

standard setting discussed in the previous chapter. Our algorithm will continue to proceed until such a time that the direct, product-based estimate from inequality (4.34) falls below our required error tolerance. When this happens, we will exit the adaptive loop, and will have primal and dual stochastic collocation FEM approximations, u_*^{SC} and z_*^{SC}, as well as an error estimate for the corrected goal functional computed using these quantities.

Of course, we have not yet actually calculated this corrected goal functional; however, we have all of the necessary quantities required to accomplish this task, which allows us to proceed along the lines discussed in the previous sections of this chapter. A summary of the process, in its entirety, is given in Algorithm 4.2.

4.6 Implementation via the 'goafem' sub-toolbox

As with our previous chapters, we will now briefly discuss the structure of the software that will be used to implement the adaptive algorithm set out above, and to test the theoretical ideas that furnish it. In this case, we utilise the existing stochastic collocation FEM software discussed in Chapter 3, with an additional goal-oriented sub-toolbox that has been created to work in conjunction with the existing code to produce the results we require.

In a similar manner to the standard setting, the central components of our implementation are housed in a single main driver, `goafem_singlelevelSC`, which calls all of the major functions and scripts that are required for running the algorithm. The first step that this driver accomplishes is initialisation, which is done predominantly via a self-contained script, `goafem_stochcol_adaptive_global_settings`, for initialisation of all of the required quantities, including types of diffusion coefficient, probability density function, and others.

One feature of the initialisation that we highlight is the bespoke quantity generation driver `goafem_stochcol_qtychoice`. Rather than forcing the use of specific test problems, this driver uses cell-array structures to facilitate the selection of any implemented right-hand sides and quantities of interest with any other combination of initial variables, including the domain. This allows for a wider variety of tests to be performed than previous implementations outside of the goal-oriented framework, and is easily expandable to allow for further domains and quantities of interest to be defined at later stages.

The adaptive loop then proceeds in a similar manner as it did in Chapter 3. As in the standard setting discussed in the previous chapter, we can perform our calculations in parallel due to having distinct, decoupled problems at each collocation point. Hence, the solution step, as well as the generation of indicators, are parallelised across the set of collocation points, as they were before. In this case, we solve the primal and dual problems associated with each collocation point together using the function `goafem_stochcol_fem_solver`. Each of these is solved in the same manner as previously; we consider piecewise linear finite element approximations here, although higher order approximations can be utilised, as was done in the standard setting.

Our next step is to use the `goafem_stochcol_fem_estimator` function in order to call a spatial error estimation strategy. New functions have been provided to allow for different estimation strategies to be called with the primal and dual problems sequentially, although the focus of our implementation will be the two-level estimation strategy, which is implemented via the `goafem_diffpost_p1_with_p1_2level` function.

At this point, we also have the quantities required to, at negligible expense, compute the deterministic goal functional, $Q(u_{\bullet\bar{\boldsymbol{y}}})$, at each collocation point, using the bilinear form in equation (4.31). As with the standard setting, our main driver then computes additional information required by our estimates, such as the enhanced collocation point set, and the norms of Lagrange polynomials. After we have assembled the spatial error indicators, the function `goafem_stochcol_est_parametric` computes the indirect parametric estimates.

The choice of whether to perform spatial or parametric refinement is decided by a new function, `goafem_indicator`, where all three of the different procedures discussed in Section 4.5 are implemented. The implementation of our direct error estimates is similar to the standard framework that we discussed in Chapter 3, as is the execution of the the refinement routines which follow. With the remainder of our adaptive algorithm retaining the structure that we have utilised previously, we proceed through our adaptive loop until our given stopping criterion is reached.

The last major difference in the code is the calculation of the corrected goal functional. This is mostly computed in the main driver, but the bilinear form $\mathcal{B}(u_{\bullet}^{SC}, z_{\bullet}^{SC})$, whose computation was discussed at length in Section 4.4, is computed in the newly designed `goafem_doBilinearForm` function. As in the non-goal-oriented setting, we can run a reference

script, `goafem_referenceSC`, to understand more about the performance of the adaptive algorithm.

4.7 Experiments

We introduce a number of experiments that are designed to test different features of our approach to goal-oriented estimation. All computations in this chapter were performed with an Intel i7-9700K CPU @4.80GHz with 48GB of RAM. In what follows, we will assume the Clenshaw–Curtis rule for sets of collocation points with all of our tests, as well as an initial index set of $I_0 = \{\mathbf{1}\}$, leading to a single collocation point at the origin.

In terms of the spatial contribution to our error estimate, we calculate error indicators corresponding to element edges at each iteration using a two-level error estimation strategy. For the purposes of evaluating the parametric contribution, we use indicators emanating from indices associated with the reduced margin of the current index set. Further, we adopt Dörfler marking techniques with spatial marking parameter $\theta_x = 0.3$ and parametric marking parameter $\theta_{\boldsymbol{y}} = 0.3$.

Our first test problem in this chapter consists of a simple experiment that is designed to demonstrate the computing of the expected average of a solution to a given parametric PDE problem over a small subdomain. To this end, we set our spatial domain to have an L-shaped geometry, $D = (-1, 1)^2 \setminus (-1, 0]^2$, with subdomain $D_0 = (0.25, 0.75)^2$.

For the primal problem, we take $f(x) = 1$ in the right-hand side functional given in problem (3.7), while the goal functional for this problem is given by

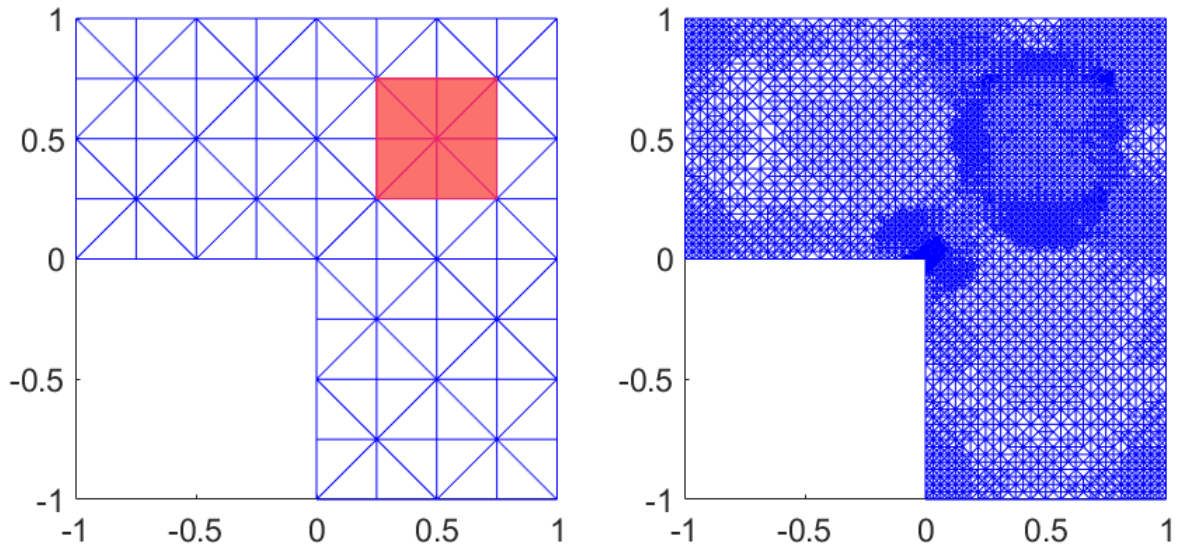
$$\mathcal{Q}(u) = \frac{1}{|D_0|} \int_\Gamma \int_D \chi_0(x) u(x, \boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}), \tag{4.43}$$

with the characteristic function $\chi_0 : D \to \{0, 1\}$ defined by

$$\chi_0(x) := \begin{cases} 1, & x \in D_0, \\ 0, & x \in D \setminus D_0. \end{cases} \tag{4.44}$$

This region is denoted in red in Figure 4.1 (a).

For the diffusion coefficient, we assume the affine representation of the diffusion coefficient



(a) (b)

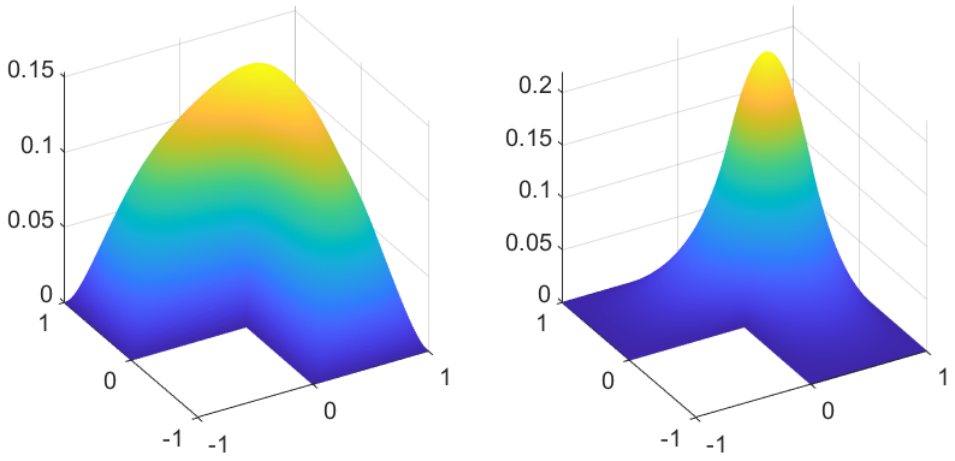
Figure 4.1: Diagrams pertaining to the spatial domain involved with the first test problem: the initial mesh $\mathcal{T}_0$ and subdomain D_0 in (a), and the mesh at $\mathcal{T}_{15}$ in (b).

given by equation (3.69), with the number of random variables, $M = 4$. The expansion coefficients a_m describe the same Fourier modes that we utilised in Chapter 3, with the same parameter values. We assume that each random variable admits uniform probability density function, so that $\varrho_m(y_m) = 1/2$ for each $m = 1, ..., M$.

In terms of parameters that are specific to adaptivity, we adopt a minimum cardinality rule for combining primal and dual marked sets and the sum-of-squares rule for determining refinement type. Our initial mesh given by Figure 4.1 (a), and we run our algorithm to a tolerance $\texttt{tol} = \texttt{1e} - \texttt{5}$.

We attain our required tolerance after 50 iterations, at which point, we have $13,902,030$ degrees of freedom involved with the problem, resulting from $1,027,463$ elements and 27 collocation points. We first note the refinement pattern exhibited by the adaptive refinement procedure, which can be seen in the plot of the spatial mesh at iteration 15, given in Figure 4.1 (b). In particular, we retain the heavier areas of refinement in the corners of the domain that we observed with our experiments in Chapter 3, including the geometric singularity induced by the re-entrant corner located at the origin. We further notice, however, that there are also heavy levels of refinement around the edges of the subdomain D_0, which represents a singular region

induced by the dual problem. What we can observe from this is that our adaptive algorithm successfully identifies the singularities emanating from both the primal and dual problems.

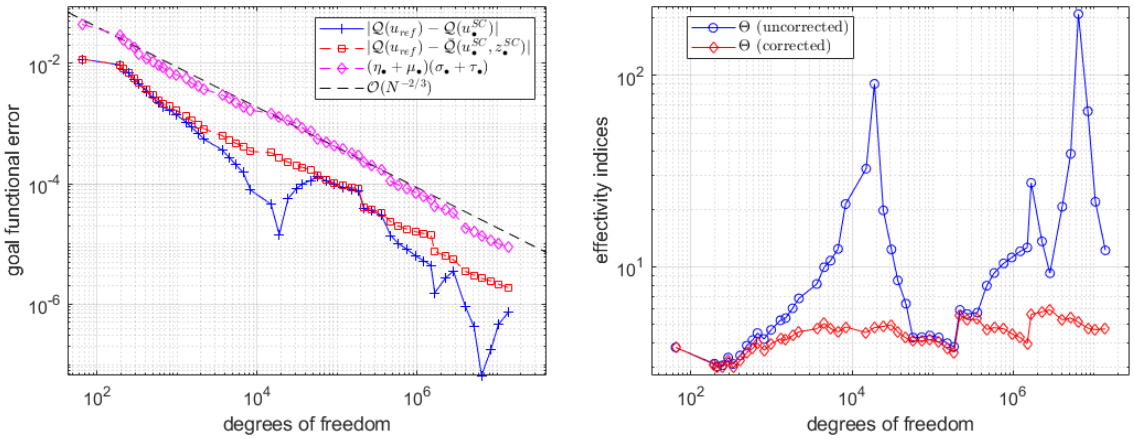


(a) (b)

Figure 4.2: The mean of the primal solution (a) and dual solution (b) associated with our first test problem for this set of experiments.

The final set of collocation points in the first two dimensions is also shown in Figure 4.1. In particular, the index set is more significantly evolved in the earlier parameters than in the latter ones, although this is expected as well, given the form of our diffusion coefficient in representation (3.70). The mean of the solutions that corresponds to this setup are given in Figure 4.2, and the estimated value of the quantity of interest is $\bar{\mathrm{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) = 0.124119$ to 6 decimal places. Of course, qualitatively, we can see that this resembles a sensible estimate if we look at the magnitude of the primal solution in Figure 4.2 (b). We observe the dual solution exhibits a more concentrated peak around the region of D_0, which is also entirely predictable given the formulation of the dual problem is concentrated around this subdomain.

We now turn to the errors admitted by our solution. In particular, we begin by describing the plot given in Figure 4.3 (a). In this plot, we give our actual (direct) error estimate at each iteration in magenta, whereupon we have the error for the final iteration just below our stopping tolerance. For this setup, our adaptive algorithm reliably reduces the error estimate at each iteration. Furthermore, this error estimate appears to decay at around order $\mathcal{O}(N^{-2/3})$, where N represents the total degrees of freedom associated with the approximation.



(a) (b)

Figure 4.3: The error decay associated with the first test problem in (a), with the error estimate (magenta), the uncorrected goal error (blue) and the corrected goal error (red). In (b), we show the effectivity indices.

These results appear to validate inequality (4.34), which predicts that the order of convergence for the quantity of interest should be doubled in comparison to the equivalent results for SCFEM approximations in the standard, non-goal-oriented setting. Hence, the convergence rates obtained here are consistent with the experiments performed in Chapter 3, as well as previous work in the non-goal-oriented setting, where we typically see error decays of around half the order (described as order $\mathcal{O}(N^{-0.35})$ in [41]). Furthermore, our findings also match similar results for the deterministic setting, where matters of convergence are more theoretically well-understood.

As with the non-goal-oriented setting, we want to determine how efficient our error estimation strategy is. Indeed, inequality (4.34) holds for the estimate that we have plotted, but without a lower bound, we do not know how sharp this inequality is in practice. This issue is particularly relevant to the goal-oriented setting given that our strategy involves a product-based estimate, which means that we expect the order of magnitude of the effectivity indices will be roughly doubled in comparison to the standard setting.

Therefore, as we have done previously in Chapter 3, we use reference solutions $u_{ref} \approx u$, $z_{ref} \approx z$, to examine the 'true' error in the quantity of interest. As previously, we employ a piecewise quadratic approximation on a uniform refinement on our final spatial mesh to

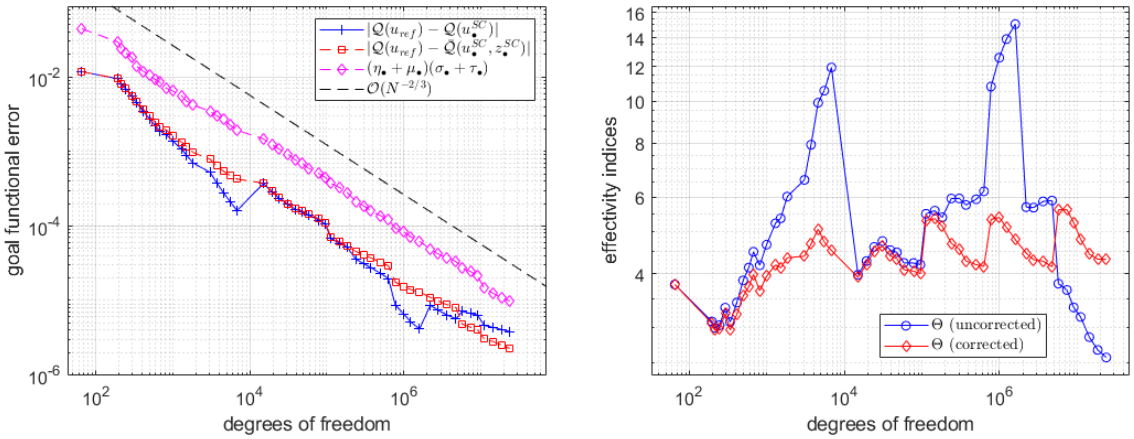
calculate per-collocation point samples, and we enhance our final index set by its reduced margin to obtain the 'reference' index set. Our final result will be numerical approximations that satisfy

$$\left|\mathcal{Q}(u) - \bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC})\right| \approx \left|\mathcal{Q}(u_{ref}) - \bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC})\right|.$$

On doing all of the above, we can then plot the error in the uncorrected goal functional, $\left|\mathcal{Q}(u_{ref}) - \mathcal{Q}(u_\bullet^{SC})\right|$, and the corrected goal functional, $\left|\mathcal{Q}(u_{ref}) - \bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC})\right|$, which are given by the blue and red lines respectively. From this, we make a few observations. We immediately note the disparity between the error in the corrected goal functional and the error in the uncorrected goal functional, which is rather more erratic by comparison. Our conclusion is that the newly defined correction term provides additional stability to the decay in the true quantity of interest, protecting it against sudden dips or rises.

In addition to guarding against sudden fluctuations, we see that the error decay in the corrected goal functional far more closely matches our error estimate. The associated effectivity indices, Θ, are given in Figure 4.3 (b), and confirm this observation further. The effectivity indices for the error in the uncorrected goal functional undergo considerable fluctuations, peaking at over 100. Furthermore, given the best theoretical result that we obtained for the uncorrected goal functional was inequality (4.2), there is no indication that these indices could not fluctuate even more at lower error tolerances. This is a significant contrast when compared with the far more stable effectivities for the corrected goal function, which average at around 4.40 and remain consistently well below 10, which represents a positive result in this setting.

As a variant on the experiment we have discussed, we also consider the above setup, with the number of random variables M, increased to 8. While this problem is predictably more computationally intensive, in terms of memory requirements and CPU time, with $23,773,926$ degrees of freedom required to achieve our required tolerance, the overall results are remarkably similar. We note the more stable error decay of the 'true' error in the corrected goal functional, with effectivity indices averaging around 4.3 for the entire process, but consistently remaining below 6. We further observe the possible deterioration of the convergence rate for the uncorrected goal functional towards the end of the adaptive process. With the results plotted in Figure 4.4, we deduce from our experiments that adjusting the number of random variables does not significantly affect our results in terms of convergence rates, or the efficiency of our estimate.



(a) (b)

Figure 4.4: The error decay associated with a modification of the setup to include 8 random variables in (a), with associated effectivity indices in (b).

For our second experiment, we will introduce something new. Up until now, we have assumed at various points that we had a simple representation of the right-hand side and goal-functional, with the underlying functionals F, Q being of the forms described in equations (3.6) and (4.5), respectively. We now modify these representations slightly, and write our functionals in the form:

$$\mathcal{F}(v) = \int_\Gamma \int_D f_0(x) v(x, \boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}) - \int_\Gamma \int_D \boldsymbol{f}(x) \cdot \nabla v(x, \boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}) \tag{4.45}$$

$$\mathcal{Q}(v) = \int_\Gamma \int_D q_0(x) v(x, \boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}) - \int_\Gamma \int_D \boldsymbol{q}(x) \cdot \nabla v(x, \boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}) \tag{4.46}$$

We note that these are common representations of the right-hand side functionals in the goal-oriented setting (see [50] and [85], amongst others). Further, the additional vector $\boldsymbol{f} := (f_1(x_1, x_2), f_2(x_1, x_2))$ allows us to introduce non-geometric singularities to our problem without significantly altering the overall structure of our work; indeed, we have

$$\int_D \boldsymbol{f}(x) \cdot \nabla v(x, \boldsymbol{y}) \, \mathrm{d}x = - \int_D v(x, \boldsymbol{y}) \nabla \cdot \boldsymbol{f}(x) \, \mathrm{d}x + \int_{\partial D} v(x, \boldsymbol{y}) \boldsymbol{f}(x) \cdot \boldsymbol{n} \, \mathrm{d}S \tag{4.47}$$

using Green's first identity.

We now define our problem using these modified representations. For our spatial domain, we introduce the square domain, $D = (-1, 1)^2$. This represents a simpler geometry than that of our first test problem which will enable us to isolate the singularities that we will introduce in our problem. These singularities are induced by representations (4.45) and (4.46), where we define

$f_0(x) = q_0(x) = 0$ and $\boldsymbol{f}(x) = (\chi_f(x), 0)$, $\boldsymbol{q}(x) = (\chi_q(x), 0)$, with the characteristic function $\chi_f : D \to \{0, 1\}$ defined by

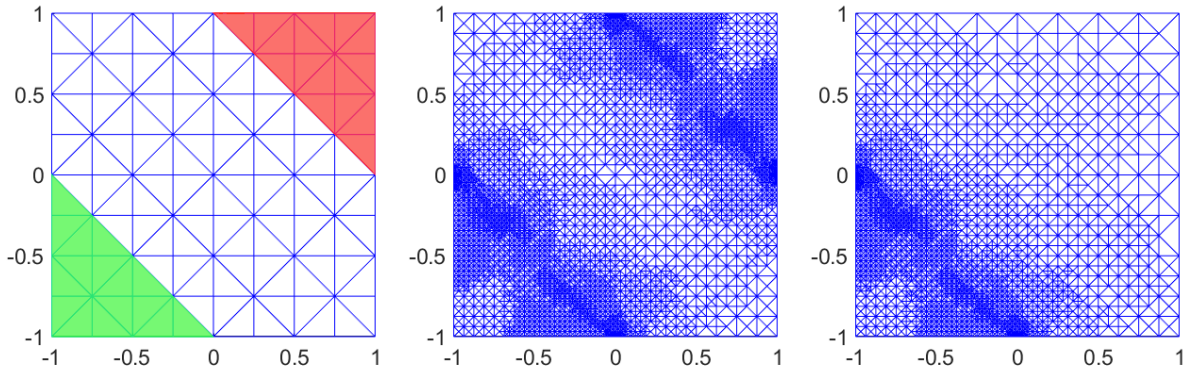
$$\chi_f(x) := \begin{cases} 1, & x \in D_f \subset D, \\ 0, & x \in D \setminus D_f, \end{cases} \tag{4.48}$$

and the characteristic function $\chi_q : D \to \{0, 1\}$ defined similarly with the subdomain D_q, rather than D_f. In this instance we take the subdomain D_f to be the triangle defined by the points $(-1, -1), (-1, 0), (0, -1)$, and D_q to be the triangle defined by the points $(0, 1), (1, 0), (1, 1)$. Via equation (4.47), we have

$$\mathcal{F}(v) = -\int_\Gamma \int_{D_f} \frac{\partial v}{\partial x_1}(x, \boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}), \tag{4.49}$$

$$\mathcal{Q}(v) = -\int_\Gamma \int_{D_q} \frac{\partial v}{\partial x_1}(x, \boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}). \tag{4.50}$$

The regions D_f, D_q are illustrated on the diagram given in Figure 4.5 (a), in green and red, respectively.



(a) (b) (c)

Figure 4.5: Diagrams pertaining to the discrete domains involved with the second problem: the initial mesh $\mathcal{T}_0$ and characteristic functions subdomains in (a), the mesh at $\mathcal{T}_{15}$ in (b), and the mesh that would be produced without a goal-oriented procedure in (c).

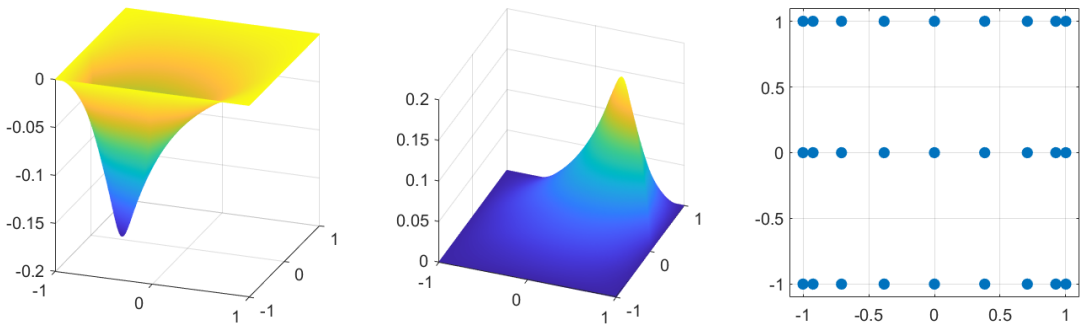
We consider the quadratic expansion (3.72), with the underlying coefficients a_m representing the same planar Fourier modes of increasing order that we utilised for representation (3.70), in our first test problem. We also retain the same uniform probability density function as the previous problem.

As before, we use a sum-of-squares rule for determining the type of refinement, and combine

primal and dual marked sets using a minimum cardinality rule. Our initial mesh takes on the same resolution as our previous problem, and is given by Figure 4.5 (a).

This time, running our algorithm to a tolerance $\texttt{tol} = \texttt{1e} - \texttt{5}$, our algorithm terminates after 53 iterations, whereupon we have $1,073,114$ elements and 67 collocation points in our approximation, producing $36,024,761$ degrees of freedom. The final estimated functional value is $\bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) = -0.00642726$ to 8 decimal places. Given the form of our right-hand side and goal functional, the refinement pattern associated with this problem is of particular interest. Indeed, the primal solution yields a singularity along the line $x_2 + x_1 = -1$, while the dual solution exhibits a singularity along the line $x_2 + x_1 = 1$.

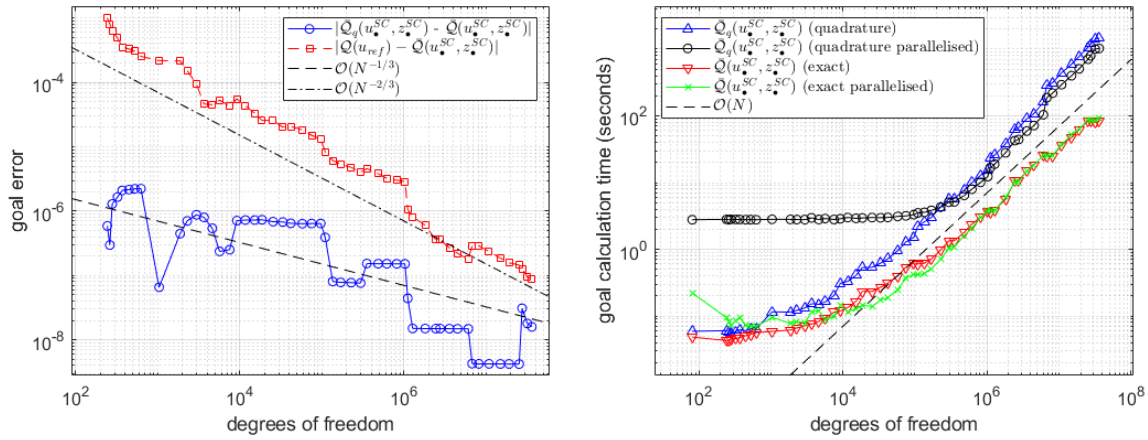
In particular, we see a far more stark contrast between refinement associated with the primal problem and the dual problem in comparison to our first test problem, due to the subdomains D_f and D_q being at opposite ends of domain of our problem formulation, D. This leads to particularly heavy refinement in regions close to singular lines associated with the subdomains of interest, which we observe in Figure 4.5 (b). The refinement pattern that we see in this figure mirrors the deterministic goal-oriented setting, as is the fact that the standard, non-goal-oriented, framework cannot identify the non-geometric singularity induced by the dual problem at all. This latter fact is demonstrated with Figure 4.5 (c).



(a) (b) (c)

Figure 4.6: The mean of the primal solution (a) and dual solution (b) associated with our second test problem, along with the final collocation point set for the first two dimensions (c).

The mean of the primal and dual solutions associated with this problem are given in Figure 4.6. These solutions, which peak along the singular lines associated with the primal and dual problems respectively, are unsurprising, and represent similar results to the deterministic



(a) (b)

Figure 4.7: A comparison for different methods of calculating the corrected goal functional, with the error of the quadrature rule in (a), and the time taken to compute $\bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC})$ in (b).

goal-oriented setting.

The final set of collocation points in the first two dimensions is also shown in Figure 4.6 (c), where the corresponding index set exhibits high levels of anisotropy. The final index set reached the fourth level in the first random variable (recalling definition (3.40), this corresponds to 9 collocation points when paired with any other indices), but only the second level (corresponding to 3 points) in the others. Naturally, this demonstrates the importance of adaptivity in the parameter domain.

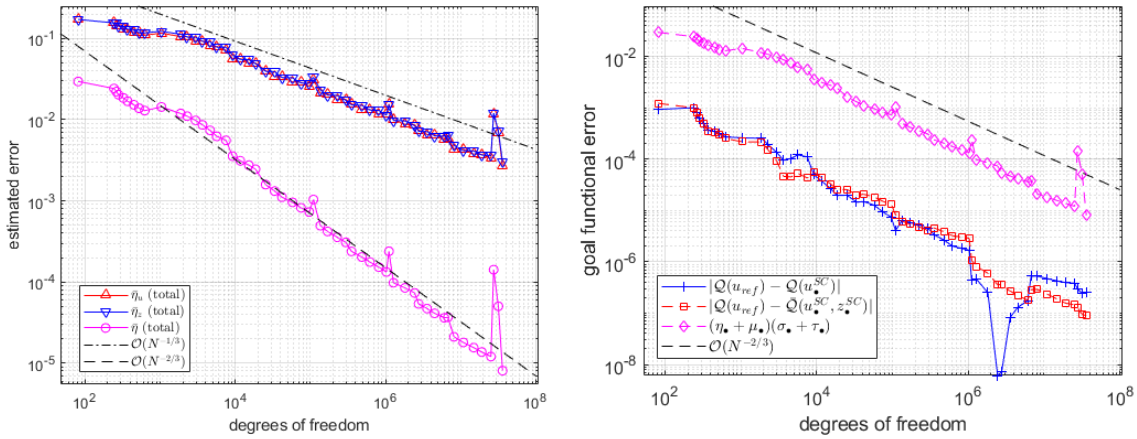
For this experiment, the effect of computing the correction term discussed in Section 4.4 is of particular interest, due to our use of a nonlinear diffusion coefficient that admits an exact representation. The corresponding result for the affine diffusion coefficient appears simple enough to warrant its use in any case; however, higher-order constructions of the bilinear form are increasingly intricate, and it is not immediately clear whether there are significant benefits to having an exact representation, rather than simply applying a quadrature rule. We therefore seek to examine the balance between the potential for an increase in computational cost against our desire to calculate the correction term as accurately as possible. The quadrature-based implementation of the bilinear form calculation, in all cases, will be performed using an enhanced collocation point set emanating from the reduced margin associated with the current index set at any given iteration.

Some of the results of this comparison are given in Figure 4.7. We first note that, to our relief, the quadrature-based calculations are easily accurate enough for our purposes. Observing the trends in Figure 4.7 (a), we see that, for the duration of our experiment, the error admitted by the quadrature rule remains consistently around an order of magnitude or more below that of the calculated error in the goal functional using the reference quantity. This is significant, as it means that the use of the quadrature rule does not meaningfully affect our numerical results. Despite this, we notice that the error decay for the quadrature rule does not appear to be as fast as the error in the goal functional. While the error in the quantity of interest decays at the expected rate of $\mathcal{O}(N^{-2/3})$, the error in our quadrature rule appears to decay at around $\mathcal{O}(N^{-1/3})$. This means that if we were to run our algorithm to finer tolerances that require around 10^8 degrees of freedom or more, it may be prudent to utilise a quadrature rule with a larger set of collocation points.

One notable surprise, however, is that the computational expense associated with the exact representation of the bilinear form is lower than that of the quadrature-based representation. After around $1e^5$ degrees of freedom, the time required to compute the goal functional appears to be bounded above by $\mathcal{O}(N)$ for the exact implementation. The quadrature calculation exhibits a similar growth in computational expense, although the total time remains consistently higher than the time required to perform the exact calculation. While it is possible that different implementations could alter these results, it appears that, based on our own rudimentary implementations, an exact implementation for the quadratic expansion of the diffusion coefficient is at least comparable with use of a quadrature rule.

To ensure that our results were not due to a lack of parallelisation for the computation of the bilinear form in the correction term, we also verified these results against a parallelised implementation of the quadrature rule. Notably, representation (4.40) allows our calculation of the deterministic coefficients to be parallelised over the set of collocation points in the quadrature rule. Parallelisation of the quadrature calculation was originally overlooked due to the first few iterations being comparatively expensive in comparison to the non-parallelised computation. On further examination, however, we find that this trend begins to reverse at around 1e5 degrees of freedom, which we demonstrate with Figure 4.7 (b).

We hypothesise that during the first few iterations, due to the fact that the required calculations are not sufficiently data-intensive, the advantages of parallelisation do not outweigh its initial



(a) (b)

Figure 4.8: The error decay associated with the second test problem. In (a), we show the interplay between primal and dual, and in (b), we show the error estimate (magenta), the uncorrected goal error (blue) and the corrected goal error (red).

overhead costs. However, as we proceed through our algorithm, our calculations will become significantly more data-intensive. At this point, the ability to perform different iterations of a loop simultaneously will confer a significant advantage in terms of computation time. This new trend fully materialises at around `1e6` degrees of freedom, at which point, the parallelised calculation was, on average, around 46% faster than the standard `for`-loop. However, our original conclusion remains the same – despite the modest improvements yielded by parallelisation at later stages in the adaptive algorithm, the exact implementation still appears to be faster than our quadrature-based implementation.

We now use the same procedure as we did for the first test problem in order to plot the true error in the goal functional alongside the error estimate at each iteration in Figure 4.8. On doing this, we see somewhat similar results to the first test problem. The error decay appears to exhibit a little more instability than for our first test problem. Nevertheless, the overall rate of $\mathcal{O}(N^{-2/3})$ appears to be the same.

The effect of the correction term is also clearly visible, with the true error decay in the corrected goal functional appearing to be more stable than that of the uncorrected goal functional. The corresponding effectivity indices are not plotted here, but examining on Figure 4.8 (b), they appear to be significantly larger for this problem, even for the corrected goal functional. Further

calculations reveal that the average of the effectivity indices is around 100. Based on additional experiments that were conducted in order to measure the loss of efficiency in our estimate, it is hypothesised that the sharpness of the Cauchy–Schwarz inequality for our product-based estimate has some relation to the size of the subdomains on which our problems are posed.

As this problem is constructed specifically to show the interplay between primal and dual, we include a demonstration of this with Figure 4.8 (a). Specifically, we see that the direct primal and dual estimates each decay at a rate of $\mathcal{O}(N^{-1/3})$, which leads to the product decaying at order $\mathcal{O}(N^{-2/3})$ as expected. This contrasts with the results that we would expect if we fail to account for the dual problem. By inequality (4.34), we know that the error decay in the goal functional for a non-goal-oriented will be bounded between $\mathcal{O}(N^{-1/3})$ and $\mathcal{O}(N^{-2/3})$.

The upper bound on the rate of convergence is obvious. The reason why the rate may be greater than $\mathcal{O}(N^{-1/3})$, however, is that refining based on indicators only associated with the primal problem will still induce general improvements in the spatial mesh that can benefit the dual. As an illustration of this, we recall Figure 4.5 (c). While it is clear that the heaviest refinement is concentrated at the singular region for the primal, we observe that even at iteration 15, the spatial refinement has started to propagate through the rest of the domain far away from this region. Hence, when we actually achieve our required tolerance, there will be refinements everywhere in the domain, which means that the error associated with the dual problem will also decrease. It is also worth mentioning here that reductions in the parametric contribution to the error will also benefit both the primal and the dual.

However, our observation that the rate of convergence for this problem is similar for both goal-oriented and non-goal-oriented procedures overlooks the importance of our adaptive algorithms here for multiple reasons. Firstly, on the theoretical side, even if the error estimate associated with a quantity of interest does decay at a relatively fast rate for a given problem, we could not possibly know this a priori, as the exact rate will depend on the precise nature of the problem. In particular, we do not know how refinements will propagate throughout our domains, and how and when this will affect the errors in the dual problem. Furthermore, we cannot even evaluate the 'true' convergence rate a posteriori without computing a reference goal, which does not itself represent a realistic scenario for practical attempts to estimate quantities of interest. More importantly, notwithstanding the above objection about knowing the 'true' error decay for a given problem, the only way of obtaining any sort of reasonable

estimate of the error in the quantity of interest necessarily involves some form of error estimation, which we do not have without a goal-oriented procedure.

For our final experiment, we examine a setup involving point-wise estimation. We consider the crack domain $D := (-1,1)^2 \setminus ([-1,0] \times \{0\})$. One observes that this domain is not Lipschitz, and so contradicts the assumptions made for our model problem. It is known, however, that the domain $D_\delta = (-1,1)^2 \setminus T_\delta$ is Lipschitz, where T_δ being the closure of triangle formed by the set of vertices $\{(-1,-\delta),(0,0),(-1,\delta)\}$. We can therefore interpret this domain, and the resulting elliptic problems on this domain, using the limit D_δ as $\delta \to 0$. For practical purposes, we set the slit size $\delta = 0.005$.

We now consider the same right-hand side as the first test problem, taking $f(x) = 1$ in definition (3.6). For the goal functional, we let

$$\mathcal{Q}(u) = \int_\Gamma \int_D q(x) u(x,\boldsymbol{y}) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \tag{4.51}$$

with the weighting function $q(x)$ is taken to be the mollifier described by definition (2.64) with radius $\mathfrak{r} = 0.15$, centred at the location $x = (0.4, -0.5)$.

In this setting, we take the exponential diffusion coefficient described by equation (3.74) with $M = 4$. The expansion coefficients a_m is set to be defined by $a_0 = 1$, and

$$a_m(x) = \sqrt{\lambda_m}\phi_m(x), \quad m = 1, ..., M \tag{4.52}$$

where $\{(\lambda_m, \phi_m)\}$ are eigenpairs of the integral operator

$$\int_D \mathrm{Cov}[a](x,x')\phi(x') \,\mathrm{d}x' \tag{4.53}$$

with a synthetic covariance function

$$\mathrm{Cov}[a](x,x') = \sigma_a^2 \exp\left(-\frac{|x_1 - x_1'|}{l_1} - \frac{|x_2 - x_2'|}{l_2}\right). \tag{4.54}$$

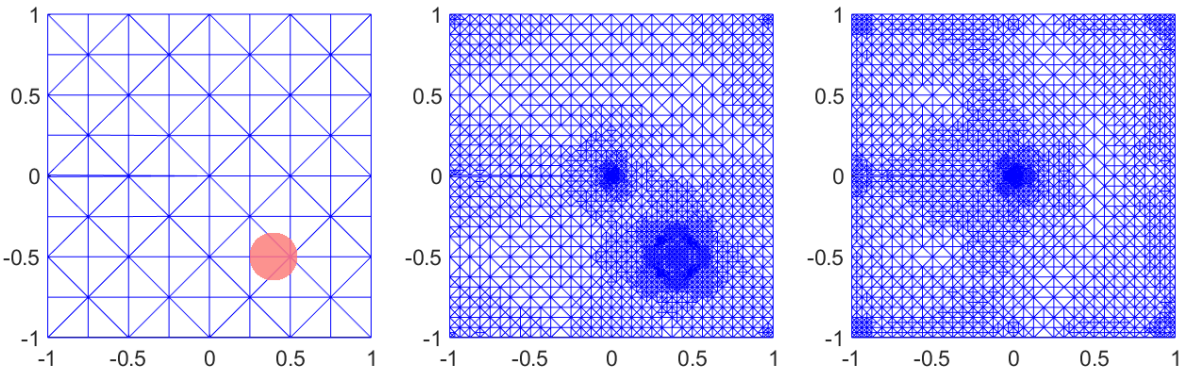
With these choices, our diffusion coefficient can be written in the form

$$a(x,\boldsymbol{y}) := \exp\left(\mathfrak{d}(x,\boldsymbol{y})\right), \quad \mathfrak{d}(x,\boldsymbol{y}) = \mathbb{E}[\mathfrak{d}](x) + \sum_{m=1}^{M} \sqrt{\lambda_m}\phi_m(x) y_m,$$

so that $a(x,\boldsymbol{y})$ is a exponentiated truncated Karhunen–Loève-type (KL) expansion of a second-order random field with covariance given by definition (4.54). In particular, a_0 is the mean of a stationary random field $\mathfrak{d}(x,\boldsymbol{y})$, with y_m representing images of pairwise uncorrelated random variables with zero mean.

Here, σ_a is the standard deviation, and l_1, l_2 represent correlation lengths, although we set $l_1 = l_2 = 1$ for our work. The underlying distribution of the random variables is assumed to be the same truncated Gaussian distribution described by definition (3.73). The purpose of this KL expansion is that it represents an optimal approximation of an infinite expansion for a diffusion coefficient that is obtained when that infinite expansion is truncated to the first M terms. A more comprehensive view of the theory surrounding Karhunen–Loève expansions may be found in [21].

This choice of the diffusion coefficient is significant for both theoretical and practical purposes. Firstly, the use of the exponential enforces the condition that the underlying random field is required to be positive. For our particular case, our choice of the KL expansion with random variables defined by (3.73) represents a truncated log-normal random field (we refer the reader to [108] for more details).

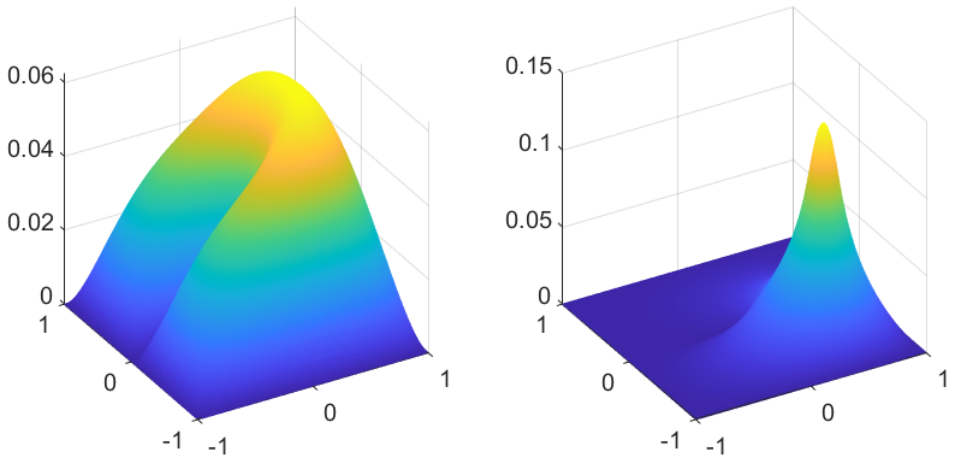


(a) (b) (c)

Figure 4.9: Diagrams pertaining to the discrete domains involved with the third problem: the initial mesh $\mathcal{T}_0$ and mollifier region in (a), the mesh $\mathcal{T}_{28}$ in (b), and the mesh $\mathcal{T}_{15}$ produced without a goal-oriented procedure in (c).

From a practical perspective, as with the previous test problem, we are utilising a non-affine diffusion coefficient, which, as we know, are typically more difficult to obtain numerical results for. However, unlike with the previous test problem, there does not exist an exact representation of the diffusion coefficient that can be implemented, and we are therefore reliant on the use of the quadrature-based calculation of the correction term. That being said, our previous results give us confidence that even for non-affine diffusion coefficients, this quadrature calculation is sufficiently accurate to yield results that are representative.

For the purposes of our experiment, we will vary the standard deviation associated with the covariance function (4.54), beginning with the value $\sigma_a = 0.5$. We will also continue our use of a sum-of-squares rule for determining the type of refinement, and the minimum cardinality rule to generate the required combination of primal and dual marked sets. Our initial mesh takes on the same resolution as our previous problems, and is given by Figure 4.9 (a).



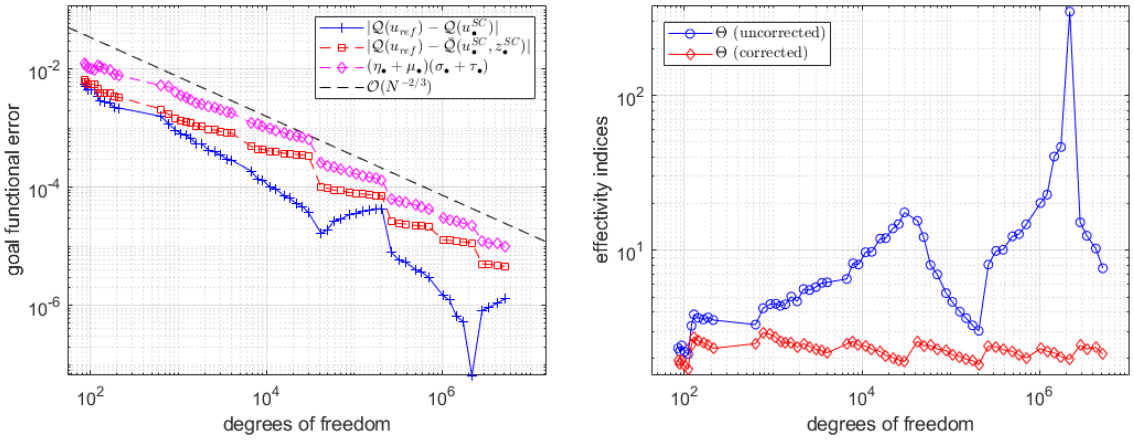
(a) (b)

Figure 4.10: The mean of the primal solution (a) and dual solution (b) associated with our third test problem.

We run our adaptive algorithm to a tolerance $\texttt{tol} = \texttt{1e}-\texttt{5}$, and end up with $1,020,695$ degrees of freedom after 46 iterations, emanating from $187,235$ elements and 59 collocation points. The value of our corrected goal functional, an estimate of the point-wise value of the primal solution at $x = (0.4, -0.5)$, is $\bar{\mathrm{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) = 0.054643$ to 6 decimal places. This appears to be a sensible estimate based on the plot of the primal solution that we have in Figure 4.10 (a).

The primal solution unsurprisingly peaks around the domain away from the boundary ∂D, which includes the crack inserted into the left-hand side of the domain. The dual solution, on the other hand, only exhibits notable behaviour around the region of interest, where the mollifier is located. The peak occurs at the centre of the mollifier, and is the location of the point-wise value we want to estimate for the primal problem. (Note that the value of the dual solution here is $z_\bullet^{SC} = 0.143180$ to 6 decimal places, which is of course, not equal to the goal functional at this point).

Figure 4.9 (b) shows particularly heavy refinement around the singularity introduced by the crack in the domain, D, as well as around the region of interest for the mollifier. This, once again, successfully reflects the interplay between primal and dual problems for this setup. A similar mesh produced without a goal-oriented procedure is shown in Figure 4.9 (c). Both of these spatial meshes are designed with around $5,000$ elements.

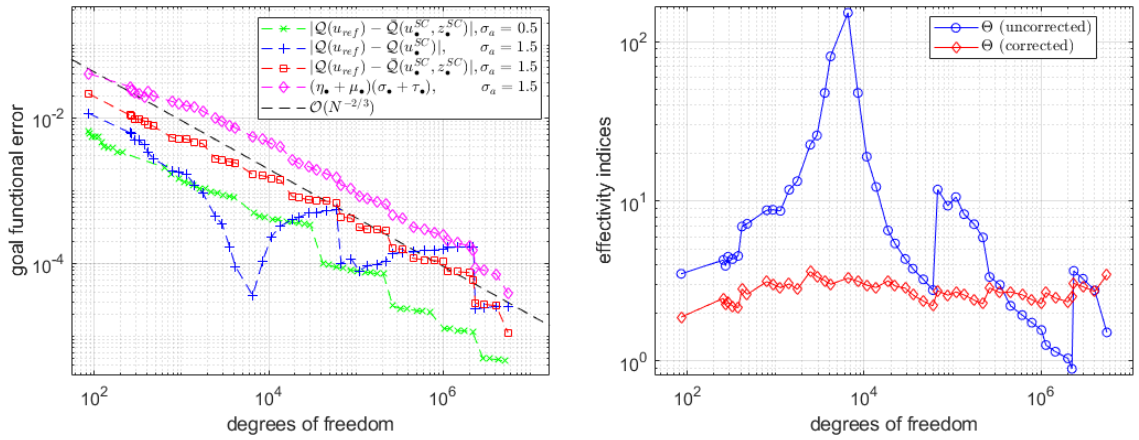


(a) (b)

Figure 4.11: The error decay associated with the third test problem, with $\sigma_a = 0.5$ in (a), with effectivity indices in (b).

The associated convergence rates are given again in Figure 4.11 (a). We see a very stable error decay associated with our error estimate, with a strictly decreasing error estimate after some pre-asymptotic behaviour at the first few iterations (around 100 degrees of freedom). The effect of the correction term is also significant. While the true error decay in the uncorrected goal functional appears to exhibit a considerable amount of instability, the corrected goal functional is much more consistent with its convergence rate, and like the error estimate, exhibits strictly monotonic decreasing behaviour after the first few iterations. In fact, the corrected goal functional, in general, tracks exceptionally closely with the error estimate, admitting average effectivity indices of just 2.29, which is a very pleasing result for a goal-oriented procedure.

We also investigate what happens if we modify the standard deviation of the KL expansion defined by the coefficients in equation (4.52). In particular, we assume a more volatile expansion, taking $\sigma_a = 1.5$. Running the adaptive process to a tolerance $\texttt{tol} = \texttt{4e} - \texttt{5}$, we require $5,558,331$ degrees of freedom to achieve our desired tolerance in 46 iterations, with a final approximation of $\bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) = 0.060807$ to 6 decimal places.



(a) (b)

Figure 4.12: The estimated errors associated with the third test problem with $\sigma_a = 1.5$ in (a), with effectivity indices in (b).

With respect to the convergence results, we observe from Figure 4.11 (b) that we consistently require more degrees of freedom to achieve a required tolerance. This is also reflected in the CPU time required to attain our final tolerance, with the adaptive procedure taking 4217.15 seconds, as opposed to requiring 1139.06 seconds to hit the same tolerance for the case when we set $\sigma_a = 0.5$. Despite this, we see that (in a similar way to when we changed the number of random variables for our first test problem) the rate of convergence is exactly the same, with the error estimate decaying at a very consistent rate.

The disparity between the corrected and uncorrected goal functionals is of particular note here, for multiple reasons. Of particular note is the fact that in this experiment, the uncorrected goal functional appeared to behave even more erratically than in the same experiment with the value $\sigma_a = 0.5$. Despite this, however, the error decay associated with the corrected goal functional remains remarkably stable, and again, tracks very closely with the error estimate, with average effectivity indices of around 2.73. We conclude that on increasing the volatility to $\sigma_a = 1.5$, the correction term proves to be even more vital for the purposes of stabilising the error decay. The consistency of our results is a particular highlight again with all effectivity indices remaining between around 2.15 and 3.63 after the first iteration. From these experiments, we conclude that that the correction term introduced in Section 4.3 is imperative for parametric PDE problems involving highly volatile input data.

Chapter 5

Dual-Weighted Residuals

Previously, we discussed an extension of stochastic collocation FEM approximations and their associated adaptive algorithms to the goal-oriented framework. We now return to the origins of the goal oriented framework and consider another approach involving a corrected goal functional along the lines of the residual method discussed in Section 2.9.

In the first part of this chapter, we utilise the same technique for generating a correction term that we have in the previous chapter, as well as the same splitting, to obtain estimates for the spatial and parametric contributions to the total error. We then proceed to discuss minor variants of this approach, as well as how the adaptive algorithm we discussed previously must be modified to incorporate our new error estimation strategy.

5.1 Dual-weighted residuals

Given arbitrary $\psi_\bullet \in \mathcal{V}$, we use the complete weak formulations (3.7) and (4.2), as well as the linearity of the underlying bilinear form, $\mathcal{B}$, in order to deduce

$$\begin{aligned} \mathcal{Q}(u - u_\bullet^{SC}) &= \mathcal{B}(u - u_\bullet^{SC}, z) \\ &= \mathcal{B}(u - u_\bullet^{SC}, z - \psi_\bullet) + \mathcal{B}(u - u_\bullet^{SC}, \psi_\bullet) \\ &= \mathcal{B}(u - u_\bullet^{SC}, z - \psi_\bullet) + \mathcal{F}(\psi_\bullet) - \mathcal{B}(u_\bullet^{SC}, \psi_\bullet). \end{aligned} \tag{5.1}$$

We therefore have

$$\begin{aligned}\mathcal{Q}(u) - \bar{\mathcal{Q}}_u(u_\bullet^{SC}, \psi_\bullet) &= \mathcal{B}(u - u_\bullet^{SC}, z - \psi_\bullet) \\ &= \mathcal{F}(z - \psi_\bullet) - \mathcal{B}(u_\bullet^{SC}, z - \psi_\bullet) \\ &=: \mathfrak{R}\left[u_\bullet^{SC}\right](z - \psi_\bullet)\end{aligned} \tag{5.2}$$

where

$$\bar{\mathcal{Q}}_u(u_\bullet^{SC}, \psi_\bullet) := \mathcal{Q}(u_\bullet^{SC}) + \mathcal{F}(\psi_\bullet) - \mathcal{B}(u_\bullet^{SC}, \psi_\bullet). \tag{5.3}$$

We begin to analyse the two terms in equation (5.2), using a residual-based approach. Using the description of our right-hand side functional given in definition (3.8), we have

$$\mathcal{F}(z - \psi_\bullet) = \int_\Gamma \int_D f(z - \psi_\bullet)\, \mathrm{d}x\, \mathrm{d}\rho(\boldsymbol{y}). \tag{5.4}$$

Further, using Green's first identity to integrate by parts element-wise leads to

$$\begin{aligned}\int_K a\nabla u_\bullet^{SC} \cdot \nabla(z - \psi_\bullet)\, \mathrm{d}x &= -\int_K (z - \psi_\bullet)\nabla \cdot (a\nabla u_\bullet^{SC})\, \mathrm{d}x + \int_{\partial K} (z - \psi_\bullet) a\nabla u_\bullet^{SC} \cdot \boldsymbol{n}\, \mathrm{d}S \\ &= -\int_K (z - \psi_\bullet)\nabla \cdot (a\nabla u_\bullet^{SC})\, \mathrm{d}x + \int_{\partial K} (z - \psi_\bullet) a \frac{\partial u_\bullet^{SC}}{\partial \boldsymbol{n}}\, \mathrm{d}S.\end{aligned}$$

Combining this with equation (5.4), and invoking the idea of element jumps given in definition (2.28), we use the same approach as with the residual estimator given in Section 2.9 to obtain

$$\begin{aligned}\mathfrak{R}\left[u_\bullet^{SC}\right](z - \psi_\bullet) = \sum_{K \in \mathcal{T}_\bullet} \Bigg(&\int_\Gamma \int_K \left(f + \nabla \cdot (a\nabla u_\bullet^{SC})\right)(z - \psi_\bullet)\, \mathrm{d}x\, \mathrm{d}\rho(\boldsymbol{y}) \\ &+ \int_\Gamma \int_{\partial K \setminus \partial D} \frac{a}{2} \left[\!\!\left[\frac{\partial u_\bullet^{SC}}{\partial n} \right]\!\!\right](z - \psi_\bullet)\, \mathrm{d}S\, \mathrm{d}\rho(\boldsymbol{y}) \Bigg).\end{aligned} \tag{5.5}$$

We must now turn our attention to the finer details of the above representation, in a similar manner to how we proceeded with deterministic dual-weighted residuals. The obvious issues we must immediately address are the lack of knowledge of the dual solution $z \in \mathcal{V}$, and our choice of $\psi_\bullet \in \mathcal{V}$. In the deterministic setting, we chose an approximation, and our arbitrary function, with the intention of inducing weights that represented an approximation error that could be calculated or bounded above by some other quantity. In this instance, we wish to induce similar weights, but also use this as an opportunity to distinguish between spatial and parametric contributions.

One sensible choice would be to recapitulate the splitting that we first described in equation (3.60), but for the dual problem, rather than the primal. Following this idea, we invoke an enhanced dual solution

$$z_\star^{SC} := \hat{z}_\bullet^{SC} + \left(\tilde{z}_\bullet^{SC} - z_\bullet^{SC}\right). \tag{5.6}$$

with refined dual approximations

$$\hat{z}_\bullet^{SC} := \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \hat{z}_{\bullet\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}), \tag{5.7}$$

and

$$\tilde{z}_\bullet^{SC} := \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet} \tilde{z}_{\bullet\bar{\boldsymbol{y}}}(x) \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}), \quad \text{with} \quad \tilde{z}_{\bullet\bar{\boldsymbol{y}}} = z_{\bullet\bar{\boldsymbol{y}}},\ \forall\, \bar{\boldsymbol{y}} \in Y_\bullet \subset \widetilde{Y}_\bullet. \tag{5.8}$$

Following these definitions, we make the assumption that $z \approx z_\star^{SC}$, and set $\psi_\bullet := z_\bullet^{SC}$, so that we obtain the approximation

$$\begin{aligned} z - \psi_\bullet &:\approx z_\star^{SC} - z_\bullet^{SC} \\ &= \left(\hat{z}_\bullet^{SC} - z_\bullet^{SC}\right) + \left(\tilde{z}_\bullet^{SC} - z_\bullet^{SC}\right). \end{aligned} \tag{5.9}$$

which decomposes the residual (5.5) into 2 distinct components:

$$\mathfrak{R}\left[u_\bullet^{SC}\right](z - \psi_\bullet) \approx \mathfrak{R}\left[u_\bullet^{SC}\right](\hat{z}_\bullet^{SC} - z_\bullet^{SC}) + \mathfrak{R}\left[u_\bullet^{SC}\right](\tilde{z}_\bullet^{SC} - z_\bullet^{SC}). \tag{5.10}$$

We first deal with the spatial contribution, $\mathfrak{R}\left[u_\bullet^{SC}\right](\hat{z}_\bullet^{SC} - z_\bullet^{SC})$. In analogous fashion to equation (4.10), we have

$$\hat{z}_\bullet^{SC} - z_\bullet^{SC} = \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} (\hat{z}_{\bullet\bar{\boldsymbol{y}}}(x) - z_{\bullet\bar{\boldsymbol{y}}}(x)) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}). \tag{5.11}$$

We define the residual quantities

$$\begin{aligned} r|_K &:= f + \nabla \cdot (a \nabla u_\bullet^{SC}) \\ j|_{\partial K} &:= \begin{cases} \dfrac{a}{2} \left[\!\!\left[\dfrac{\partial u_\bullet^{SC}}{\partial n} \right]\!\!\right], & \partial K \subset \partial D, \\ 0, & \partial K \cap \partial D = \emptyset, \end{cases} \end{aligned} \tag{5.12}$$

and write

$$\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}} := (\hat{z}_{\bullet\bar{\boldsymbol{y}}}(x) - z_{\bullet\bar{\boldsymbol{y}}}(x)) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \tag{5.13}$$

for brevity. Here, we have omitted any references to the iteration (that is, using '$\bullet$' notation) on r and j. We will continue to do so on other residual and weight quantities for brevity, as this can be done without any ambiguity. Then, using the Cauchy–Schwarz inequality, we have on each element

$$\int_\Gamma \int_K r_K(\hat{z}_\bullet^{SC} - z_\bullet^{SC})\, \mathrm{d}x\, \mathrm{d}\rho(\boldsymbol{y}) \leqslant \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \|r\|_{\Gamma;K} \|\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}$$

for the element residual integral, and similarly,

$$\int_\Gamma \int_{\partial K \setminus \partial D} j_{\partial K}(\hat{z}_\bullet^{SC} - z_\bullet^{SC})\, \mathrm{d}x\, \mathrm{d}\rho(\boldsymbol{y}) \leqslant \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \|j\|_{\Gamma;\partial K} \|\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}$$

for the edge residual integral. This leads us to the conclusion

$$\begin{aligned}\mathfrak{R}\big[u_\bullet^{SC}\big]\,(\hat{z}_\bullet^{SC}-z_\bullet^{SC}) &\leqslant \sum_{K\in\mathcal{T}_\bullet}\sum_{\bar{\boldsymbol{y}}\in Y_\bullet}\Big(\|r\|_{\Gamma;K}\|\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}+\|j\|_{\Gamma;\partial K}\|\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}\Big)\\ &\leqslant \sum_{K\in\mathcal{T}_\bullet}\sum_{\bar{\boldsymbol{y}}\in Y_\bullet}\Big(\|r\|_{\Gamma;K}^2+h_K^{-1}\|j\|_{\Gamma;\partial K}^2\Big)^{\frac{1}{2}}\Big(\|\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}^2+h_K\|\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}^2\Big)^{\frac{1}{2}}\end{aligned} \tag{5.14}$$

where the element diameter h_K describes a scaling factor for the edge residual contribution. We also observe that the first bracket in the above is independent of the sum over collocation points emanating from the dual solution.

We now turn to the parametric component of the residual, given by $\mathfrak{R}\left[u_\bullet^{SC}\right](\tilde{z}_\bullet^{SC}-z_\bullet^{SC})$. Applying the result of Lemma 4.1 to the dual approximations allows us to obtain

$$\tilde{z}_\bullet^{SC}-z_\bullet^{SC}=\sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet}(z_{\bullet\bar{\boldsymbol{y}}}(x)-z_\bullet^{SC}(x,\bar{\boldsymbol{y}}))\widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}). \tag{5.15}$$

Recalling definitions (5.12), we proceed by bounding the integrals for the element and edge residual contributions. Defining

$$\widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}:=(z_{\bullet\bar{\boldsymbol{y}}}(x)-z_\bullet^{SC}(x,\bar{\boldsymbol{y}}))\widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}), \tag{5.16}$$

while applying the Cauchy–Schwarz inequality in a similar manner to what we did above yields, on each element

$$\int_\Gamma\int_K r_K(\tilde{z}_\bullet^{SC}-z_\bullet^{SC})\,\mathrm{d}x\,\mathrm{d}\rho(\boldsymbol{y})\leqslant\sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet}\|r\|_{\Gamma;K}\|\widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}$$

for the element residual and

$$\int_\Gamma\int_K j_{\partial K\setminus\partial D}(\tilde{z}_\bullet^{SC}-z_\bullet^{SC})\,\mathrm{d}x\,\mathrm{d}\rho(\boldsymbol{y})\leqslant\sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet}\|j\|_{\Gamma;\partial K\setminus\partial D}\|\widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K\setminus\partial D}$$

for the edge residual. It then follows that the upper bound

$$\begin{aligned}\mathfrak{R}\big[u_\bullet^{SC}\big]\,(\tilde{z}_\bullet^{SC}-z_\bullet^{SC}) &\leqslant \sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet}\sum_{K\in\mathcal{T}_\bullet}\Big(\|r\|_{\Gamma;K}\|\widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}+\|j\|_{\Gamma;\partial K}\|\widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}\Big)\\ &\leqslant \sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet\setminus Y_\bullet}\sum_{K\in\mathcal{T}_\bullet}\Big(\|r\|_{\Gamma;K}^2+h_K^{-1}\|j\|_{\Gamma;\partial K}^2\Big)^{\frac{1}{2}}\Big(\|\widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}^2+h_K\|\widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}^2\Big)^{\frac{1}{2}}\end{aligned} \tag{5.17}$$

holds for the parametric residual. The result for this residual is written in terms of collocation points, although we recall that refinement of sparse grids is typically driven using multi-indices from the margin, $\mathcal{M}_I$, or reduced margin, $\mathcal{R}_I$, of an index set associated with the current sparse grid. In any case, we must write our sum in terms of an index sets, $\widetilde{I}_\bullet\setminus I_\bullet$, that used to

enhance the current sparse grid, and group collocation points in the set $\widetilde{Y}_\bullet \setminus Y_\bullet$ according to their associated indices $\boldsymbol{i} \in \widetilde{I}_\bullet \setminus I_\bullet$. This will require us to utilise the sets $\widetilde{Y}_{\bullet\boldsymbol{i}}$ that we briefly invoked in Chapter 3 when discussing error estimation. Putting all of this together, we have now established the following result.

Theorem 5.1. *Denote by u and z the unique solutions to problems (3.7) and (4.2), respectively, with $\mathcal{F}$ being dependent on the deterministic quantity, F, represented by the form given in definition (3.6). Let $u_\bullet^{SC}, z_\bullet^{SC}, z_\star^{SC}, \hat{z}_\bullet^{SC}, \tilde{z}_\bullet^{SC}$ be the approximations given by definitions (3.48), (4.7) and (5.6) – (5.8) respectively. Suppose also that the residual quantities $r|_K, j|_{\partial K}$ are as given in definition (5.12) and that the quantities $\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}, \widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}$ are as given in definitions (5.13) and (5.16). Then:*

$$\mathfrak{R}\left[u_\bullet^{SC}\right](z_\star^{SC} - z_\bullet^{SC}) \leqslant \sum_{K\in\mathcal{T}_\bullet} \rho_K^u \widehat{\omega}_K^u + \sum_{\boldsymbol{i}\in\widetilde{I}_\bullet\setminus I_\bullet} \left(\sum_{K\in\mathcal{T}_\bullet} \rho_K^u \widetilde{\omega}_{K,\boldsymbol{i}}^u\right) \tag{5.18}$$

where

$$\begin{aligned}
\rho_K^u &:= \left(\|r\|_{\Gamma;K}^2 + h_K^{-1}\|j\|_{\Gamma;\partial K}^2\right)^{\frac{1}{2}},\\
\widehat{\omega}_K^u &:= \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \widehat{\omega}_{K,\bar{\boldsymbol{y}}}^u, \quad \widehat{\omega}_{K,\bar{\boldsymbol{y}}}^u := \left(\|\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}^2 + h_K\|\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}^2\right)^{\frac{1}{2}},\\
\widetilde{\omega}_{K,\boldsymbol{i}}^u &:= \sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_{\bullet\boldsymbol{i}}} \left(\|\widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}^2 + h_K\|\widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}^2\right)^{\frac{1}{2}}.
\end{aligned} \tag{5.19}$$

In the bound given by inequality (5.18), we know that the first sum represents an estimate for the spatial contribution to the error, while the second sum corresponds to an estimate for the parametric contribution. Furthermore, despite the slightly unorthodox appearance of the latter of the two sums (where the weights are not written purely per-index), the writing of the inequality in this form allows for the easy decomposition of each sum into terms that can be used as local indicators. For each element $K \in \mathcal{T}_\bullet$, the spatial indicator is given by $\rho_K^u \widehat{\omega}_K^u$, while for each index $\boldsymbol{i} \in \widetilde{I}_\bullet \setminus I_\bullet$, the parametric indicator is the sum $\sum_{K\in\mathcal{T}_\bullet} \rho_K^u \widetilde{\omega}_{K,\boldsymbol{i}}^u$.

5.2 Symmetric dual-weighted residuals

We mentioned the possibility of exploiting the symmetry of our formulations in equality (2.73), and we now discuss this topic here. Recalling our complete weak formulations (3.7) and (4.2), we observe that

$$\mathcal{Q}(u) = \mathcal{B}(u, z) = \mathcal{F}(z), \tag{5.20}$$

and we further recall Lemma 4.4. Using these results, we conclude from the linearity of $\mathcal{F}$ and $\mathcal{Q}$ that $\mathcal{F}(z - z_\bullet^{SC}) = \mathcal{Q}(u - u_\bullet^{SC})$. As a result of this, we have that for an arbitrary $\varphi_\bullet \in \mathcal{V}$,

$$\begin{aligned}\mathcal{Q}(u - u_\bullet^{SC}) &= \mathcal{F}(z - z_\bullet^{SC}) \\ &= \mathcal{B}(u, z - z_\bullet^{SC}) \\ &= \mathcal{B}(u - \varphi_\bullet, z - z_\bullet^{SC}) + \mathcal{B}(\varphi_\bullet, z - z_\bullet^{SC}) \\ &= \mathcal{B}(u - \varphi_\bullet, z - z_\bullet^{SC}) + \mathcal{Q}(\varphi_\bullet) - \mathcal{B}(\varphi_\bullet, z_\bullet^{SC}).\end{aligned} \tag{5.21}$$

We can therefore write the error in the goal functional in the same way that we did with equation (5.2). Doing this yields

$$\begin{aligned}\mathcal{Q}(u) - \bar{\mathcal{Q}}_z(u_\bullet^{SC}, \psi_\bullet) &= \mathcal{B}(u - \varphi_\bullet, z - z_\bullet^{SC}) \\ &= \mathcal{Q}(u - \varphi_\bullet) - \mathcal{B}(u - \varphi_\bullet, z_\bullet^{SC}) \\ &= \mathfrak{R}_z\left[z_\bullet^{SC}\right](u - \varphi_\bullet),\end{aligned} \tag{5.22}$$

where

$$\bar{\mathcal{Q}}_z(\varphi_\bullet, z_\bullet^{SC}) := \mathcal{F}(z_\bullet^{SC}) + \mathcal{Q}(\varphi_\bullet) - \mathcal{B}(\varphi_\bullet, z_\bullet^{SC}). \tag{5.23}$$

For now, we restrict our attention a simple goal functional, dependent on a Q of the form given in representation (4.5). This is of a similar form to the right-hand side, and we stress that while changing this assumption will affect parts of the calculations, both the right-hand side and goal functional representations can be modified, as we will see in Section 5.3. Applying integration by parts per-element yields

$$\int_K a\nabla(u - \varphi_\bullet) \cdot z_\bullet^{SC} \,\mathrm{d}x = -\int_K (u - \varphi_\bullet)\nabla \cdot (a\nabla z_\bullet^{SC}) \,\mathrm{d}x + \int_{\partial K} (u - \varphi_\bullet) a \frac{\partial z_\bullet^{SC}}{\partial n} \,\mathrm{d}S,$$

which, combined with our previous assumptions gives us

$$\begin{aligned}\mathfrak{R}_z\left[z_\bullet^{SC}\right](u - \varphi_\bullet) = \sum_{K \in \mathcal{T}} \Bigg(&\int_\Gamma \int_K (u - \varphi_\bullet)\left(q + \nabla \cdot (a\nabla z_\bullet^{SC})\right) \,\mathrm{d}x \,\mathrm{d}\rho(\boldsymbol{y}) \\ &+ \int_\Gamma \int_{\partial K \setminus \partial D} (u - \varphi_\bullet) \frac{a}{2} \left[\!\left[\frac{\partial z_\bullet^{SC}}{\partial \boldsymbol{n}} \right]\!\right] \,\mathrm{d}S \,\mathrm{d}\rho(\boldsymbol{y}) \Bigg).\end{aligned} \tag{5.24}$$

Mirroring our previous approach, we now invoke the splitting set out in equation (3.60) to write

$$\mathfrak{R}_z\left[z_\bullet^{SC}\right](u - \varphi_\bullet) \approx \mathfrak{R}_z\left[z_\bullet^{SC}\right](\hat{u}_\bullet^{SC} - u_\bullet^{SC}) + \mathfrak{R}_z\left[z_\bullet^{SC}\right](\tilde{u}_\bullet^{SC} - u_\bullet^{SC}). \tag{5.25}$$

We deal with the spatial and parametric contributions in turn, with residual quantities defined by

$$\begin{aligned} r_z|_K &:= q + \nabla \cdot (a \nabla z_\bullet^{SC}) \\ j_z|_{\partial K} &:= \begin{cases} \frac{a}{2} \left[\!\left[\frac{\partial z_\bullet^{SC}}{\partial n} \right]\!\right], & \partial K \subset \partial D, \\ 0, & \partial K \cap \partial D = \emptyset. \end{cases} \end{aligned} \tag{5.26}$$

We further use equation (4.10) to write the quantity $\hat{u}_\bullet^{SC} - u_\bullet^{SC}$ in terms of the summands

$$\widehat{\mathrm{u}}_{\bar{\boldsymbol{y}}} := (\hat{u}_{\bullet \bar{\boldsymbol{y}}}(x) - u_{\bullet \bar{\boldsymbol{y}}}(x)) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \tag{5.27}$$

and do the same, using Lemma (4.1), to the quantity $\tilde{u}_\bullet^{SC} - u_\bullet^{SC}$, leading to us defining

$$\widetilde{\mathrm{u}}_{\bar{\boldsymbol{y}}} := (u_{\bullet \bar{\boldsymbol{y}}}(x) - u_\bullet^{SC}(x, \bar{\boldsymbol{y}})) \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}). \tag{5.28}$$

which leads to the upper bounds:

$$\begin{aligned} \mathfrak{R}_z\left[z_\bullet^{SC}\right](\hat{u}_\bullet^{SC} - u_\bullet^{SC}) &\leqslant \sum_{K \in \mathcal{T}_\bullet} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \left(\|r_z\|_{\Gamma;K} \|\widehat{\mathrm{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K} + \|j_z\|_{\Gamma;\partial K} \|\widehat{\mathrm{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K} \right) \\ &\leqslant \sum_{K \in \mathcal{T}_\bullet} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \left(\|r_z\|_{\Gamma;K}^2 + h_K^{-1} \|j_z\|_{\Gamma;\partial K}^2 \right)^{\frac{1}{2}} \left(\|\widehat{\mathrm{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}^2 + h_K \|\widehat{\mathrm{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}^2 \right)^{\frac{1}{2}} \end{aligned}$$

for the spatial contribution, and

$$\begin{aligned} \mathfrak{R}_z\left[z_\bullet^{SC}\right](\tilde{u}_\bullet^{SC} - u_\bullet^{SC}) &\leqslant \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet \setminus Y_\bullet} \sum_{K \in \mathcal{T}_\bullet} \left(\|r_z\|_{\Gamma;K} \|\widetilde{\mathrm{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K} + \|j_z\|_{\Gamma;\partial K} \|\widetilde{\mathrm{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K} \right) \\ &\leqslant \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet \setminus Y_\bullet} \sum_{K \in \mathcal{T}_\bullet} \left(\|r_z\|_{\Gamma;K}^2 + h_K^{-1} \|j_z\|_{\Gamma;\partial K}^2 \right)^{\frac{1}{2}} \left(\|\widetilde{\mathrm{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}^2 + h_K \|\widetilde{\mathrm{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}^2 \right)^{\frac{1}{2}} \end{aligned}$$

for the parametric contribution. This leads us to concluding the following results.

Theorem 5.2. *Denote by u and z the unique solutions to problems (3.7) and (4.2), respectively, with $\mathcal{Q}$ being dependent on the deterministic quantity, Q, as represented by the form given in definition (4.5). Let $u_\bullet^{SC}, z_\bullet^{SC}, u_\star^{SC}, \hat{u}_\bullet^{SC}, \tilde{u}_\bullet^{SC}$ be the approximations given by definitions (3.48), (4.7) and (3.57) – (3.59) respectively. Suppose also that the residual quantities $r|_K, j|_{\partial K}$ are as given in definition (5.12) and that the quantities $\widehat{\mathrm{u}}_{\bar{\boldsymbol{y}}}, \widetilde{\mathrm{u}}_{\bar{\boldsymbol{y}}}$ are as given in definitions (5.27) and (5.28). Then:*

$$\mathfrak{R}_z\left[z_\bullet^{SC}\right](u_\star^{SC} - u_\bullet^{SC}) \leqslant \sum_{K \in \mathcal{T}_\bullet} \rho_K^z \widehat{\omega}_K^z + \sum_{\boldsymbol{i} \in \widetilde{I}_\bullet \setminus I_\bullet} \left(\sum_{K \in \mathcal{T}_\bullet} \rho_K^z \widetilde{\omega}_{K,\boldsymbol{i}}^z \right) \tag{5.29}$$

where

$$\begin{aligned}
\rho_K^z &:= \left(\|r_z\|_{\Gamma;K}^2 + h_K^{-1} \|j_z\|_{\Gamma;\partial K}^2 \right)^{\frac{1}{2}}, \\
\widehat{\omega}_K^z &:= \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \widehat{\omega}_{K,\bar{\boldsymbol{y}}}^z, \quad \widehat{\omega}_{K,\bar{\boldsymbol{y}}}^z := \left(\|\widehat{\mathfrak{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}^2 + h_K \|\widehat{\mathfrak{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}^2 \right)^{\frac{1}{2}}, \\
\widetilde{\omega}_{K,\boldsymbol{i}}^z &:= \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_{\bullet\boldsymbol{i}}} \left(\|\widetilde{\mathfrak{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K}^2 + h_K \|\widetilde{\mathfrak{u}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;\partial K}^2 \right)^{\frac{1}{2}}.
\end{aligned} \tag{5.30}$$

Corollary 5.3. *Suppose that the assumptions required for Theorems 5.1 and 5.2 hold, with all relevant quantities being defined as they are in those theorems. Then we have*

$$\mathcal{Q}(u) - \bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) \lessapprox \frac{1}{2} \sum_{K \in \mathcal{T}_\bullet} (\rho_K^u \widehat{\omega}_K^u + \rho_K^z \widehat{\omega}_K^z) + \frac{1}{2} \sum_{\boldsymbol{i} \in \widetilde{I}_\bullet \setminus I_\bullet} \left(\sum_{K \in \mathcal{T}_\bullet} \rho_K^u \widetilde{\omega}_{K,\boldsymbol{i}}^u + \rho_K^z \widetilde{\omega}_{K,\boldsymbol{i}}^z \right) \tag{5.31}$$

where $\bar{\mathcal{Q}}$ is given by equation (4.33).

Proof. Substituting $\psi_\bullet := z_\bullet^{SC}$ and $\psi_\bullet := z_\bullet^{SC}$ into equations (5.3) and (5.23) gives us

$$\bar{\mathcal{Q}}_u(u_\bullet^{SC}, z_\bullet^{SC}) = \bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) = \bar{\mathcal{Q}}_z(u_\bullet^{SC}, z_\bullet^{SC}). \tag{5.32}$$

Further, using equations (5.2) and (5.22) yields

$$\mathcal{Q}(u) - \bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) \approx \mathfrak{R}\left[u_\bullet^{SC}\right](z - z_\bullet^{SC}) \tag{5.33}$$

and

$$\mathcal{Q}(u) - \bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) \approx \mathfrak{R}_z\left[z_\bullet^{SC}\right](u - u_\bullet^{SC}). \tag{5.34}$$

Taking the average of the above two equations and applying the upper bounds in Theorems 5.1 and 5.2 completes the proof. □

We denote the approach which utilises Corollary 5.3 as the *symmetric* dual-weighted residual method for the stochastic collocation FEM setting. Of course, we can take any weighted average of the residuals in inequality (5.31), and calculate an upper bound for the expression

$$\alpha\, \mathfrak{R}\left[u_\bullet^{SC}\right](z - z_\bullet^{SC}) + (1 - \alpha)\, \mathfrak{R}_z\left[z_\bullet^{SC}\right](u - u_\bullet^{SC})$$

for some $\alpha \in \mathbb{R}$, although without any a priori knowledge of the problems and their solutions, $\alpha = 1/2$ represents a sensible choice. Of course, with this symmetric method, we must, in effect, calculate and then take the average of two residuals and sets of indicators, increasing the computational expense associated with this approach. That being said, we are, at least, not

required to calculate two separate correction terms (which can be expensive to compute) given the observation made in equation (5.32).

While the standard dual-weighted residual method uses weights to factor the dual problem into our calculations, the interaction between primal and dual problems does not appear, prima facie, to be as obvious as for the approach based on products of norms that we discussed in Chapter 4. This symmetric method can alleviate this concern, and guarantee more predictable behaviour when looking at the error indicators to ascertain the distribution of errors across our domains.

5.3 Alternative representations of the right-hand side

In our results for dual-weighted residuals so far, we have assumed the right-hand side functionals are of the forms given in definitions (3.6) and (4.5). However, for the second test problem that we discussed in the previous chapter, we described a modification of this approach, representing our functionals using representations (4.45) and (4.46). In particular, with the representation (4.45), we have

$$\mathcal{F}(z-\psi_\bullet) = \int_\Gamma \int_D f_0(z-\psi_\bullet)\,\mathrm{d}x\,\mathrm{d}\rho(\boldsymbol{y}) - \int_\Gamma \int_D \boldsymbol{f}\cdot\nabla(z-\psi_\bullet)\,\mathrm{d}x\,\mathrm{d}\rho(\boldsymbol{y}) \tag{5.35}$$

so that, we modify equation (5.5) to

$$\begin{aligned}\mathfrak{R}\left[u_\bullet^{SC}\right](z-\psi_\bullet) = \sum_{K\in\mathcal{T}_\bullet} \Bigg(&\int_\Gamma \int_K \left(f + \nabla\cdot\boldsymbol{f} + \nabla\cdot(a\nabla u_\bullet^{SC})\right)(z-\psi_\bullet)\,\mathrm{d}x\,\mathrm{d}\rho(\boldsymbol{y}) \\ &+ \int_\Gamma \int_{\partial K\setminus\partial D} \left(\frac{a}{2}\left[\!\left[\frac{\partial u_\bullet^{SC}}{\partial n}\right]\!\right] + \boldsymbol{f}\cdot\boldsymbol{n}\right)(z-\psi_\bullet)\,\mathrm{d}S\,\mathrm{d}\rho(\boldsymbol{y})\Bigg),\end{aligned} \tag{5.36}$$

which follows from equation (4.47). By applying the same ideas as previously, we can conclude that Theorem 5.1 holds with the element and edge residual quantities defined as:

$$\begin{aligned} r|_K &:= f + \nabla\cdot\boldsymbol{f} + \nabla\cdot(a\nabla u_\bullet^{SC}) \\ j|_{\partial K} &:= \begin{cases} \dfrac{a}{2}\left[\!\left[\dfrac{\partial u_\bullet^{SC}}{\partial n}\right]\!\right] + \boldsymbol{f}\cdot\boldsymbol{n}, & \partial K \subset \partial D, \\ 0, & \partial K \cap \partial D = \emptyset. \end{cases} \end{aligned} \tag{5.37}$$

Alternatively, we can also use a similar application of Green's first identity to (4.46) to modify equation (5.24) to

$$\begin{aligned}\mathfrak{R}_z\left[z_\bullet^{SC}\right](u-\varphi_\bullet) = \sum_{K\in\mathcal{T}}\Bigg(&\int_\Gamma\int_K (u-\varphi_\bullet)\left(q+\nabla\cdot\boldsymbol{q}+\nabla\cdot(a\nabla z_\bullet^{SC})\right)\,\mathrm{d}x\,\mathrm{d}\rho(\boldsymbol{y})\\ &+\int_\Gamma\int_{\partial K\setminus\partial D}(u-\varphi_\bullet)\left(\frac{a}{2}\left[\!\left[\frac{\partial z_\bullet^{SC}}{\partial \boldsymbol{n}}\right]\!\right]+\boldsymbol{q}\cdot\boldsymbol{n}\right)\mathrm{d}S\,\mathrm{d}\rho(\boldsymbol{y})\Bigg).\end{aligned} \tag{5.38}$$

and Theorem 5.2 then holds with residual quantities defined by:

$$\begin{aligned} r_z|_K &:= q_0+\nabla\cdot\boldsymbol{q}+\nabla\cdot(a\nabla z_\bullet^{SC})\\ j_z|_{\partial K} &:= \begin{cases}\dfrac{a}{2}\left[\!\left[\dfrac{\partial z_\bullet^{SC}}{\partial n}\right]\!\right]+\boldsymbol{q}\cdot\boldsymbol{n}, & \partial K\subset\partial D,\\ 0, & \partial K\cap\partial D=\emptyset.\end{cases}\end{aligned} \tag{5.39}$$

The consequences of the symmetric variant discussed in Corollary 5.3 then also follow. Perhaps unsurprisingly, other minor modifications can be made to either the right-hand side functional for problem (3.7), or the goal functional in problem (4.2), without it significantly altering the results of the theory.

5.4 Evaluation of the associated quantities

Having discussed the theoretical results in the previous sections, we must now discuss more practical considerations, such as the computation of the residual norms and associated weights. In a similar manner to what we saw in Chapter 4, we have additional difficulties with calculating some of these quantities due to the diffusion coefficient.

In particular, the residual quantities, as with the corrected goal functional, can be calculated exactly for polynomial expansions of the diffusion coefficient, although the calculations become increasingly intricate for higher-order polynomial expansions. This is a fact which is not helped by the square appearing in the integrands, and a quadrature-based approach may therefore be preferable in almost all cases. That being said, we include representations for linear and quadratic expansions of the diffusion coefficient, if only to provide exact quantities that are implementable for the purposes of comparisons with quadrature-based calculations.

5.4.1 Evaluation of the element residual

We turn to the evaluation of the residuals, beginning with the element residual norm $\|r\|_{\Gamma;K}$. To simplify the calculations that follow we assume that the element residual norm we want to calculate is given by

$$\|r\|_{\Gamma;K} = \left(\int_\Gamma \int_K \left(f + \nabla \cdot (a \nabla u_\bullet^{SC}) \right)^2 \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}) \right)^{1/2}. \tag{5.40}$$

If we want to utilise the representation of the right-hand side given in equality (4.45), then we observe that by comparing definitions (5.12) and (5.37), we may replace f by $f_0 + \nabla \cdot \boldsymbol{f}$ in what follows.

In order to obtain an exact representation for the diffusion coefficient, we want to obtain an integral that is separable into a sum over products of deterministic and spatial components. We can expand the integrand above, which leads us to three integrals,

$$\begin{aligned}
\mathfrak{I}_1 &:= \int_\Gamma \int_K f(x)^2 \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}), \\
\mathfrak{I}_2 &:= 2 \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \int_\Gamma \int_K f(x) \nabla \cdot (a(x,\boldsymbol{y}) \nabla u_{\bullet \bar{\boldsymbol{y}}}(x)) \, L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}), \\
\mathfrak{I}_3 &:= \sum_{\bar{\boldsymbol{y}}, \bar{\boldsymbol{y}}' \in Y_\bullet} \int_\Gamma \int_K \nabla \cdot (a(x,\boldsymbol{y}) \nabla u_{\bullet \bar{\boldsymbol{y}}}(x)) \, \nabla \cdot \left(a(x,\boldsymbol{y}) \nabla u_{\bullet \bar{\boldsymbol{y}}'}(x) \right) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}).
\end{aligned} \tag{5.41}$$

Trivially, we have

$$\mathfrak{I}_1 := \int_K f(x)^2 \, \mathrm{d}x \int_\Gamma \mathrm{d}\rho(\boldsymbol{y}), \tag{5.42}$$

independently of the diffusion coefficient. If our diffusion coefficient is an affine expansion of random parameters, as in representation (3.69), then by defining $y_0 := 1$, we have the remaining integrals given by

$$\begin{aligned}
\mathfrak{I}_2 &:= 2 \sum_{m=0}^{M} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \int_\Gamma \int_K f(x) \nabla \cdot (a_m(x) y_m \nabla u_{\bullet \bar{\boldsymbol{y}}}(x)) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}) \\
&= 2 \sum_{m=0}^{M} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \int_K f(x) \nabla \cdot (a_m(x) \nabla u_{\bullet \bar{\boldsymbol{y}}}(x)) \, \mathrm{d}x \int_\Gamma y_m L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \, \mathrm{d}\rho(\boldsymbol{y})
\end{aligned} \tag{5.43}$$

and

$$\begin{aligned}
\mathfrak{I}_3 &:= \sum_{m,l=0}^{M} \sum_{\bar{\boldsymbol{y}}, \bar{\boldsymbol{y}}' \in Y_\bullet} \int_\Gamma \int_K \nabla \cdot (a_m(x) y_m \nabla u_{\bullet \bar{\boldsymbol{y}}}(x)) \, \nabla \cdot \left(a_l(x) y_l \nabla u_{\bullet \bar{\boldsymbol{y}}'}(x) \right) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \, \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}) \\
&= \sum_{m,l=0}^{M} \sum_{\bar{\boldsymbol{y}}, \bar{\boldsymbol{y}}' \in Y_\bullet} \int_K \nabla \cdot (a_m(x) \nabla u_{\bullet \bar{\boldsymbol{y}}}(x)) \nabla \cdot (a_l(x) \nabla u_{\bullet \bar{\boldsymbol{y}}'}(x)) \, \mathrm{d}x \int_\Gamma y_m y_l L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \, \mathrm{d}\rho(\boldsymbol{y})
\end{aligned} \tag{5.44}$$

which can all be directly calculated. For a quadratic diffusion coefficient of the form given in representation (3.72), then the calculations become more complex. We again expand the diffusion coefficient to write it in the form (4.35), which yields

$$\begin{aligned}\mathfrak{I}_2 &:= 2\sum_{m,l=0}^{M}\sum_{\bar{\boldsymbol{y}}\in Y_\bullet}\int_\Gamma\int_K f(x)\nabla\cdot(y_m a_m(x) y_l a_l(x)\nabla u_{\bullet\bar{\boldsymbol{y}}}(x))L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}x\,\mathrm{d}\rho(\boldsymbol{y})\\ &= 2\sum_{m,l=0}^{M}\sum_{\bar{\boldsymbol{y}}\in Y_\bullet}\int_K f(x)\nabla\cdot(a_m(x)a_l(x)\nabla u_{\bullet\bar{\boldsymbol{y}}}(x))\,\mathrm{d}x\int_\Gamma y_m y_l L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y})\end{aligned} \tag{5.45}$$

and (with the intermediary stage omitted for brevity),

$$\begin{aligned}\mathfrak{I}_3 := \sum_{m,l,n,k=0}^{M}\sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'\in Y_\bullet}\int_K \nabla\cdot(a_m(x)a_l(x)u_{\bullet\bar{\boldsymbol{y}}})\nabla\cdot(a_n(x)a_k(x)u_{\bullet\bar{\boldsymbol{y}}'})\,\mathrm{d}x\\ \times\int_\Gamma y_m y_l y_n y_k L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y}),\end{aligned} \tag{5.46}$$

As with the calculation of the corrected goal functional in Chapter 4, we can also approximate equation (5.40) using a quadrature rule. If we use a collocation point set $Y_\bullet$, then expanding the stochastic collocation FEM solution and using the delta property of the underlying basis functions,

$$\begin{aligned}\|r\|^2_{\Gamma;K} &= \int_\Gamma\int_K\left(f(x)+\nabla\cdot\left(a(x,\boldsymbol{y})\nabla\sum_{\bar{\boldsymbol{y}}\in Y_\bullet}u_{\bullet\bar{\boldsymbol{y}}}(x)L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\right)\right)^2\mathrm{d}x\,\mathrm{d}\rho(\boldsymbol{y})\\ &\approx \sum_{\bar{\boldsymbol{y}}'\in Y_\bullet}\int_K\left(f(x)+\nabla\cdot\left(a(x,\bar{\boldsymbol{y}}')\nabla\sum_{\bar{\boldsymbol{y}}\in Y_\bullet}u_{\bullet\bar{\boldsymbol{y}}}(x)L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}')\right)\right)^2\mathrm{d}x\,\omega_{\bar{\boldsymbol{y}}}\\ &= \sum_{\bar{\boldsymbol{y}}\in Y_\bullet}\int_K\left(f(x)+\nabla\cdot(a(x,\bar{\boldsymbol{y}})\nabla u_{\bullet\bar{\boldsymbol{y}}}(x))\right)^2\mathrm{d}x\,\omega_{\bar{\boldsymbol{y}}}\end{aligned}$$

so that

$$\|r\|^2_{\Gamma;K}\approx\sum_{\bar{\boldsymbol{y}}\in Y_\bullet}\|r_{\bar{\boldsymbol{y}}}\|^2_K\int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\,\mathrm{d}\rho(\boldsymbol{y}), \tag{5.47}$$

where the sampled residual

$$r_{\bar{\boldsymbol{y}}}|_K := f(x)+\nabla\cdot(a(x,\bar{\boldsymbol{y}})\nabla u_{\bullet\bar{\boldsymbol{y}}}(x))\,. \tag{5.48}$$

Alternatively, with an enhanced collocation point set $\widetilde{Y}_\bullet \supset Y_\bullet$, we obtain

$$\begin{aligned}
\|r\|^2_{\Gamma;K} &= \int_\Gamma \int_K \left(f(x) + \nabla \cdot \left(a(x,\boldsymbol{y}) \nabla \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \right) \right)^2 \mathrm{d}x \, \mathrm{d}\rho(\boldsymbol{y}) \\
&\approx \sum_{\bar{\boldsymbol{y}}' \in \widetilde{Y}_\bullet} \int_K \left(f(x) + \nabla \cdot \left(a(x,\bar{\boldsymbol{y}}') \nabla \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') \right) \right)^2 \mathrm{d}x \, \omega_{\bar{\boldsymbol{y}}'} \\
&= \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \int_K \left(f(x) + \nabla \cdot (a(x,\bar{\boldsymbol{y}}) \nabla u_{\bullet\bar{\boldsymbol{y}}}(x)) \right)^2 \mathrm{d}x \, \omega_{\bar{\boldsymbol{y}}'} \\
&\quad + \sum_{\bar{\boldsymbol{y}}' \in \widetilde{Y}_\bullet \setminus Y_\bullet} \int_K \left(f(x) + \nabla \cdot \left(a(x,\bar{\boldsymbol{y}}') \nabla \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}}(x) L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') \right) \right)^2 \mathrm{d}x \, \omega_{\bar{\boldsymbol{y}}'}
\end{aligned}$$

so that, substituting in the appropriate quadrature weights,

$$\|r\|^2_{\Gamma;K} \approx \sum_{\bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet} \|\widetilde{r}_{\bar{\boldsymbol{y}}}\|^2_K \int_\Gamma \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \, \mathrm{d}\rho(\boldsymbol{y}), \tag{5.49}$$

where the sampled residual

$$\widetilde{r}_{\bar{\boldsymbol{y}}}|_K := \begin{cases} r_{\bar{\boldsymbol{y}}}, & \bar{\boldsymbol{y}} \in Y_\bullet, \\ f(x) + \nabla \cdot \left(a(x,\bar{\boldsymbol{y}}) \nabla \sum_{\bar{\boldsymbol{y}}' \in Y_\bullet} u_{\bullet\bar{\boldsymbol{y}}'}(x) L_{\bar{\boldsymbol{y}}'}(\bar{\boldsymbol{y}}) \right), & \bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet \setminus Y_\bullet. \end{cases} \tag{5.50}$$

We note that analogous results exist for the element residual corresponding to the dual problem, if it is our objective to invoke a symmetric dual-weighted residual approach.

5.4.2 Evaluation of the edge residual

We now turn our attention to the edge residual norm $\|j\|_{\Gamma;\partial K}$. For the right-hand side assumed in equality (4.45)

$$\|j\|_{\Gamma;\partial K} = \left(\int_\Gamma \int_{\partial K \setminus \partial D} \left(\frac{a}{2} \left[\!\!\left[\frac{\partial u^{SC}_\bullet}{\partial n} \right]\!\!\right] + \boldsymbol{f} \cdot \boldsymbol{n} \right)^2 \mathrm{d}S \, \mathrm{d}\rho(\boldsymbol{y}) \right)^{1/2}. \tag{5.51}$$

If we seek an exact representation of the above, we want to calculate the following integrals:

$$\begin{aligned}
\mathfrak{J}_1 &:= \int_\Gamma \int_{\partial K \setminus \partial D} \boldsymbol{f} \cdot \boldsymbol{n} \; \mathrm{d}S \, \mathrm{d}\rho(\boldsymbol{y}), \\
\mathfrak{J}_2 &:= \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \int_\Gamma \int_{\partial K \setminus \partial D} a(x,\boldsymbol{y}) \left[\!\!\left[\frac{\partial u_{\bullet\bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \boldsymbol{f} \cdot \boldsymbol{n} \, \mathrm{d}S \, \mathrm{d}\rho(\boldsymbol{y}), \\
\mathfrak{J}_3 &:= \frac{1}{4} \sum_{\bar{\boldsymbol{y}}, \bar{\boldsymbol{y}}' \in Y_\bullet} \int_\Gamma \int_{\partial K \setminus \partial D} a(x,\boldsymbol{y})^2 \left[\!\!\left[\frac{\partial u_{\bullet\bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] \left[\!\!\left[\frac{\partial u_{\bullet\bar{\boldsymbol{y}}'}(x)}{\partial n} \right]\!\!\right] L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \, \mathrm{d}S \, \mathrm{d}\rho(\boldsymbol{y}).
\end{aligned} \tag{5.52}$$

We note that for the simplified version of the right-hand side we saw in Section 5.1, we have $\mathfrak{J}_1 = \mathfrak{J}_2 = 0$, and we are not required to calculate these integrals. Irrespective of the diffusion coefficient, we trivially have

$$\mathfrak{J}_1 = \int_{\partial K \setminus \partial D} \boldsymbol{f}(x) \cdot \boldsymbol{n}(x) \, \mathrm{d}S \int_\Gamma \mathrm{d}\rho(\boldsymbol{y}) \tag{5.53}$$

As with the element residual integrals, we expand the diffusion coefficient and separate the variables to write $\mathfrak{J}_2$ and $\mathfrak{J}_3$ in terms of sums over products of calculable integrals. For affine expansions of the diffusion coefficient, we have

$$\begin{aligned} \mathfrak{J}_2 &= \sum_{m=0}^{M} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \int_\Gamma \int_{\partial K \setminus \partial D} a_m(x) y_m \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \boldsymbol{f}(x) \cdot \boldsymbol{n}(x) \, \mathrm{d}S \, \mathrm{d}\rho(\boldsymbol{y}) \\ &= \sum_{m=0}^{M} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \int_{\partial K \setminus \partial D} a_m(x) \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] \boldsymbol{f}(x) \cdot \boldsymbol{n}(x) \, \mathrm{d}S \int_\Gamma y_m L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \, \mathrm{d}\rho(\boldsymbol{y}) \end{aligned} \tag{5.54}$$

and similarly,

$$\begin{aligned} \mathfrak{J}_3 = \frac{1}{4} \sum_{m,l=0}^{M} \sum_{\bar{\boldsymbol{y}}, \bar{\boldsymbol{y}}' \in Y_\bullet} \int_{\partial K \setminus \partial D} a_m(x) a_l(x) \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}}(x)}{\partial \boldsymbol{n}} \right]\!\!\right] \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}'}(x)}{\partial \boldsymbol{n}} \right]\!\!\right] \mathrm{d}S \\ \times \int_\Gamma y_m y_l L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \, \mathrm{d}\rho(\boldsymbol{y}). \end{aligned} \tag{5.55}$$

For quadratic expansions of the diffusion coefficient,

$$\begin{aligned} \mathfrak{J}_2 &= \sum_{m,l=0}^{M} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \int_\Gamma \int_{\partial K \setminus \partial D} a_m(x) y_m a_l(x) y_l \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \boldsymbol{f}(x) \cdot \boldsymbol{n}(x) \, \mathrm{d}S \, \mathrm{d}\rho(\boldsymbol{y}) \\ &= \sum_{m,l=0}^{M} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \int_{\partial K \setminus \partial D} a_m(x) a_l(x) \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] \boldsymbol{f}(x) \cdot \boldsymbol{n}(x) \, \mathrm{d}S \int_\Gamma y_m y_l L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \, \mathrm{d}\rho(\boldsymbol{y}) \end{aligned} \tag{5.56}$$

and

$$\begin{aligned} \mathfrak{J}_3 = \frac{1}{4} \sum_{m,l,n,k=0}^{M} \sum_{\bar{\boldsymbol{y}}, \bar{\boldsymbol{y}}' \in Y_\bullet} \int_{\partial K \setminus \partial D} a_m(x) a_l(x) a_n(x) a_k(x) \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}'}(x)}{\partial n} \right]\!\!\right] \mathrm{d}S \\ \times \int_\Gamma y_m y_l y_n y_k L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) L_{\bar{\boldsymbol{y}}'}(\boldsymbol{y}) \, \mathrm{d}\rho(\boldsymbol{y}). \end{aligned} \tag{5.57}$$

Alternatively, using a quadrature rule we can estimate the entire edge residual. Doing so with the collocation point set $Y_\bullet$, yields, via the usual procedure,

$$\begin{aligned} \|j\|^2_{\Gamma;K} &= \int_\Gamma \int_{\partial K \setminus \partial D} \left(\frac{a(x, \boldsymbol{y})}{2} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) + \boldsymbol{f}(x) \cdot \boldsymbol{n}(x) \right)^2 \mathrm{d}S \, \mathrm{d}\rho(\boldsymbol{y}) \\ &\approx \sum_{\bar{\boldsymbol{y}}' \in Y_\bullet} \int_{\partial K \setminus \partial D} \left(\frac{a(x, \bar{\boldsymbol{y}}')}{2} \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') + \boldsymbol{f}(x) \cdot \boldsymbol{n}(x) \right)^2 \mathrm{d}S \, \omega_{\bar{\boldsymbol{y}}'} \\ &= \sum_{\bar{\boldsymbol{y}} \in Y_\bullet} \int_{\partial K \setminus \partial D} \left(\frac{a(x, \bar{\boldsymbol{y}})}{2} \left[\!\!\left[\frac{\partial u_{\bullet \bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] + \boldsymbol{f}(x) \cdot \boldsymbol{n}(x) \right)^2 \mathrm{d}S \, \omega_{\bar{\boldsymbol{y}}} \end{aligned}$$

so that, substituting in the appropriate weights,

$$\|j\|^2_{\Gamma;\partial K} \approx \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \|j_{\bar{\boldsymbol{y}}}\|^2_{\partial K} \int_\Gamma L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}), \tag{5.58}$$

where the sampled residual

$$j_{\bar{\boldsymbol{y}}}|_K := \frac{a(x,\bar{\boldsymbol{y}})}{2} \left[\!\!\left[\frac{\partial u_{\bullet\bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] + \boldsymbol{f}(x)\cdot\boldsymbol{n}(x). \tag{5.59}$$

Alternatively, with an enhanced collocation point set $\widetilde{Y}_\bullet \supset Y_\bullet$, we obtain

$$\begin{aligned}
\|j\|^2_{\Gamma;K} &= \int_\Gamma \int_{\partial K\setminus\partial D} \left(\frac{a(x,\boldsymbol{y})}{2} \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \left[\!\!\left[\frac{\partial u_{\bullet\bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] L_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) + \boldsymbol{f}(x)\cdot\boldsymbol{n}(x) \right)^2 \mathrm{d}S \,\mathrm{d}\rho(\boldsymbol{y}) \\
&\approx \sum_{\bar{\boldsymbol{y}}'\in\widetilde{Y}_\bullet} \int_{\partial K\setminus\partial D} \left(\frac{a(x,\bar{\boldsymbol{y}}')}{2} \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \left[\!\!\left[\frac{\partial u_{\bullet\bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') + \boldsymbol{f}(x)\cdot\boldsymbol{n}(x) \right)^2 \mathrm{d}S \,\omega_{\bar{\boldsymbol{y}}'} \\
&= \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \int_{\partial K\setminus\partial D} \left(\frac{a(x,\bar{\boldsymbol{y}})}{2} \left[\!\!\left[\frac{\partial u_{\bullet\bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] + \boldsymbol{f}(x)\cdot\boldsymbol{n}(x) \right)^2 \mathrm{d}S \,\omega_{\bar{\boldsymbol{y}}'} \\
&\quad + \sum_{\bar{\boldsymbol{y}}'\in\widetilde{Y}_\bullet\setminus Y_\bullet} \int_{\partial K\setminus\partial D} \left(\frac{a(x,\bar{\boldsymbol{y}}')}{2} \sum_{\bar{\boldsymbol{y}}\in Y_\bullet} \left[\!\!\left[\frac{\partial u_{\bullet\bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] L_{\bar{\boldsymbol{y}}}(\bar{\boldsymbol{y}}') + \boldsymbol{f}(x)\cdot\boldsymbol{n}(x) \right)^2 \mathrm{d}S \,\omega_{\bar{\boldsymbol{y}}'}
\end{aligned}$$

so that

$$\|j\|^2_{\Gamma;\partial K} \approx \sum_{\bar{\boldsymbol{y}}\in\widetilde{Y}_\bullet} \|\widetilde{j}_{\bar{\boldsymbol{y}}}\|^2_{\partial K} \int_\Gamma \widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y}) \,\mathrm{d}\rho(\boldsymbol{y}), \tag{5.60}$$

where the sampled residual

$$\widetilde{j}_{\bar{\boldsymbol{y}}}|_K := \begin{cases} j_{\bar{\boldsymbol{y}}}, & \bar{\boldsymbol{y}} \in Y_\bullet, \\ \dfrac{a(x,\bar{\boldsymbol{y}})}{2} \displaystyle\sum_{\bar{\boldsymbol{y}}'\in Y_\bullet} \left[\!\!\left[\frac{\partial u_{\bullet\bar{\boldsymbol{y}}}(x)}{\partial n} \right]\!\!\right] L_{\bar{\boldsymbol{y}}'}(\bar{\boldsymbol{y}}) + \boldsymbol{f}(x)\cdot\boldsymbol{n}(x), & \bar{\boldsymbol{y}} \in \widetilde{Y}_\bullet \setminus Y_\bullet. \end{cases} \tag{5.61}$$

As before, if we want to use a symmetric dual weighted residual, we can utilise a similar result for the edge-based residual associated with the dual problem.

5.4.3 Evaluation of the residual weights

We finally discuss the calculation of weights. We recall the quantities $\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}, \widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}$ from definitions (5.13) and (5.16). These quantities already represent products of spatial and parametric components so that the norms in question are immediately calculable. For the spatial contribution to the residual, the associated element-localised norm for the weight is

$$\begin{aligned}
\|\widehat{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K} &= \|(\hat{z}_{\bullet\bar{\boldsymbol{y}}}(x) - z_{\bullet\bar{\boldsymbol{y}}}(x)) L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\|_{\Gamma;K} \\
&= \|\hat{z}_{\bullet\bar{\boldsymbol{y}}}(x) - z_{\bullet\bar{\boldsymbol{y}}}(x)\|_K \|L_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\|_\Gamma
\end{aligned} \tag{5.62}$$

and for the parametric contribution, we have

$$\begin{aligned}\|\widetilde{\mathfrak{z}}_{\bar{\boldsymbol{y}}}\|_{\Gamma;K} &= \left\|(z_{\bullet\bar{\boldsymbol{y}}}(x) - z_{\bullet}^{SC}(x,\bar{\boldsymbol{y}}))\widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\right\|_{\Gamma;K} \\ &= \left\|z_{\bullet\bar{\boldsymbol{y}}}(x) - z_{\bullet}^{SC}(x,\bar{\boldsymbol{y}}))_K\right\|\left\|\widetilde{L}_{\bar{\boldsymbol{y}}}(\boldsymbol{y})\right\|_{\Gamma}\end{aligned} \tag{5.63}$$

with equivalent results for the edge-localised norms.

This norms can themselves be calculated directly, or in the case of the spatial norms within each of the above products, these can further be estimated, both theoretically, and practically speaking, using similar techniques to those that are involved in the element-residual based estimators devised in [75].

We also note that in the case of symmetric dual-weighted residuals, we also require weights corresponding to the primal problem, although, as with the residuals themselves, these can be calculated in exactly the same manner.

5.5 Adaptive algorithm

We must now turn our attention to how the error estimation strategies discussed previously are used in order to furnish an adaptive algorithm. In particular, there are differences between this approach and both of the previous adaptive algorithms we have described, for the standard setting, and for the goal-oriented setting where a hierarchical product estimation strategy is used.

Unsurprisingly, one aspect of our adaptive algorithm that does not change is the solution step. As with our work in the previous chapter, we require the solution of both problems (3.56) and (4.6) for each $\bar{\boldsymbol{y}} \in Y_{\bullet}$ at each iteration.

Our precise estimation step, and the quantities that we are required to compute, will depend on which variant of the dual-weighted residual method we are utilising. Nevertheless, the procedures to calculate the appropriate quantities for Theorem 5.1 or Corollary 5.3 are given in Section 5.4.

It is at this point that the dual-weighted residual approach begins to simplify our procedure in comparison to the hierarchical product-based procedure. We recall that our marking step

requires us to choose between spatial and parametric refinement, and to produce a single marked set corresponding to that refinement type.

In the dual-weighted residual setting, we obtain two, rather than four, distinct sets of indicators associated with localised spatial and parametric contributions to our total error estimate. This is because the dual problem is already incorporated into our estimation via the weightings of the residuals, and we therefore do not require separate estimates and sets of indicators corresponding to the primal and dual problems. Hence, our marking step here starts to more closely resemble the standard, non-goal-oriented setting.

Assuming that we are using the first approach to dual-weighted residuals that we discussed, we have by Theorem 5.1 that estimates for the spatial and parametric contributions to the error are given by

$$\bar{\mu}_\bullet := \sum_{K\in\mathcal{T}_\bullet} \rho_K^u \widehat{\omega}_K^u, \quad \bar{\tau}_\bullet := \sum_{\boldsymbol{i}\in\widetilde{I}_\bullet\setminus I_\bullet} \sum_{K\in\mathcal{T}_\bullet} \rho_K^u \widetilde{\omega}_{K,\boldsymbol{i}}^u. \tag{5.64}$$

Here, we continue to avoid any references to the iteration on the residual and weight quantities for brevity, and because there is no requirement to retain this information beyond a given iteration of an adaptive algorithm in any case. Our choice of whether to perform spatial or parametric refinement can be simply determined by which of $\bar{\mu}_\bullet, \bar{\tau}_\bullet$ is largest. Further, we recall from Section 5.1 that the quantities $\mu_{\bullet K} := \rho_K^u \widehat{\omega}_K^u$ may be used as spatial indicators for each $K \in \mathcal{T}_\bullet$, while $\tau_{\bullet\boldsymbol{i}} := \sum_{K\in\mathcal{T}_\bullet} \rho_K^u \widetilde{\omega}_{K,\boldsymbol{i}}^u$ can be used as parametric indicators for each $\boldsymbol{i} \in \widetilde{I}_\bullet \setminus I_\bullet$.

In terms of generating direct error estimates, $\mu_\bullet$ and $\tau_\bullet$, if required, we can use the working that precedes the statement of Theorem 5.1 to obtain these; for example, a combination of the work in inequalities (5.14) and (5.17) would suffice here, although alternative bounds can be considered.

There are a few differences between this chapter and previous chapters when it comes to the practicalities of generating indicators and the resulting marked set. Firstly, unlike in previous chapters, we restrict our attention of element-based indicators. In theory, one could attempt to generate quantities which are edge-dependent by combining the residual jump component first described in defintition (5.12) with an equidistributed element component; however, this suggestion lacks sufficient theoretical motivation to warrant its implementation.

Algorithm 5.1: Dual-weighted residual Stochastic Collocation FEM algorithm

Data: Initial spatial mesh, $\mathcal{T}_0$, initial index set $I_0 = \{\mathbf{1}\}$; error tolerance $\texttt{tol} = \varepsilon$; problem parameters, including marking criteria and a refinement procedure.

Result: Goal functional $\bar{\mathcal{Q}}(u_*^{SC}, z_*^{SC}) \approx \mathcal{Q}(u)$; corresponding error estimates satisfying $(\mu_* + \tau_*) < \varepsilon$.

Set iteration counter $k = 0$; iteration factor $l \in \mathbb{N}$.

while $(\mu_l + \tau_l) \geqslant \varepsilon$ **do**

Obtain the approximations, $u_{k\bar{\boldsymbol{y}}}, z_{k\bar{\boldsymbol{y}}} \in V_k$ by solving problems (3.56), (4.6) for each $\bar{\boldsymbol{y}} \in Y_k$.

For each $K \in \mathcal{T}_k$, compute residuals ρ_K^u and spatial weight contribution $\widehat{\omega}_K^u$, and for each $K \in \mathcal{T}_k$ and $\boldsymbol{i} \in \widetilde{Y}_{k\boldsymbol{i}}$, the parametric weight contributions, $\widetilde{\omega}_{K,\boldsymbol{i}}^u$

Use the residuals and weights to determine indirect spatial and parametric estimates $\bar{\mu_k}, \bar{\tau_k}$.

Determine the marked spatial set $\widehat{\mathcal{M}}_k$ and/or the marked index set $\widetilde{\mathcal{M}}_k$.

if $k = jl,\ j \in \mathbb{N}$ **then**

Compute the direct error estimates μ_l, τ_l.

end

if $(\mu_l + \tau_l) \geqslant \varepsilon$ **then**

if $\tau_k > \mu_k$ **then**

Define $I_{k+1} := \texttt{refine}\ (I_k, \widetilde{\mathcal{M}}_k)$, and set $\mathcal{T}_{k+1} := \mathcal{T}_k$.

else

Define $\mathcal{T}_{k+1} := \texttt{refine}\ (\mathcal{T}_k, \widehat{\mathcal{M}}_k)$ and set $I_{k+1} := I_k$.

end

Set $k \to k + 1$.;

end

end

Set $u_*^{SC} := u_l^{SC}, z_*^{SC} := z_l^{SC}, \mu_* := \mu_l, \tau_* := \tau_l$.

Calculate the goal functional $\bar{\mathcal{Q}}(u_*^{SC}, z_*^{SC})$.

Secondly, there is no requirement for us to calculate marked sets of elements per-collocation point in the single-level stochastic collocation FEM setting, as the spatial component in inequality (5.18) is already written without any sum over collocation points. With the availability of per-index indicators for the parameter domain, our marking strategy here is similarly obvious.

Our refinement stage is also performed in the same way as in previous chapters, with our algorithm proceeding through subsequent iterations until such a point that our estimate in inequality (5.18) falls below our required tolerance threshold, ε. At this point, we can use our final stochastic collocation FEM approximations u_*^{SC}, z_*^{SC}, to calculate our final estimate of the goal functional by computing the required correction term, as we did with the hierarchical product estimation strategy. We summarise the modifications we have made to our adaptive process in Algorithm 5.1. In order to simplify the pseudo-code, we consider the standard dual-weighted residual procedure that we previously described in Section 5.1.

5.6 Implementation

To provide the final link between our algorithm and the resulting experiments, we describe the process of implementing our algorithm in MATLAB. We attempt to provide a similar structure to what we have outlined in previous chapters, and in the process, utilise many of the same tools that we have used for the implementation of the algorithms that we have described in previous chapters.

The main components of the adaptive process are contained in the `dwr_singlelevelSC` driver, with the initialisation performed using the `dwr_stochcol_global_settings`. We maintain the quantity cell-array structure to provide flexibility with regards to computing numerical examples corresponding to problems with various features that are desirable for testing purposes. We consider the same underlying test problems as we did in Chapter 4, and we therefore may utilise the same quantity generation driver as we did previously, namely the `goafem_stochcol_qtychoice` script.

There are no significant changes to the implementation of our solution step, and we utilise the same solver, `goafem_stochcol_fem_solver`, as we did in the previous chapter. As before,

we are utilising this solver to compute piecewise linear approximations to solve deterministic problems corresponding to each collocation point in a parallelised loop. By contrast, the construction of error estimators is now performed later in the code. Due to some of our calculations requiring all of the per-collocation point solution data, this precludes us from computing it in the same parallelised loop as before. The residuals and weights are now calculated in two separate functions that are housed in the main driver.

All of our required residual quantities are calculated in the `dwr_stochcol_diffpost` function. The first task that this function completes is the collation of all of the required function data for the purposes of calculating the residuals. For example, we require a significant number of cross multiplications of coefficient products, such as definition (5.43), for exact calculations an affine coefficient. We also require a significant number of stochastic integrals, including for exact representations, or for quadrature-based implementations. We organise all of these integrals here as well. Where possible, symmetry is considered in order to reduce computational expense associated with this process.

From here, the two functions `dwr_stochcol_intres_p1` and `dwr_stochcol_edgeres_p1` are designed to calculate element and edge residual contributions respectively. If we are utilising the symmetric variant of the dual-weighted residual method, then residuals associated with the dual problem are also calculated here.

The weights corresponding to our residuals are subsequently calculated via another function in the main driver, `dwr_stochcol_weights`. This calculates estimates of the norms required associated with both the spatial and parametric contributions to the total error estimate. The former is done via the construction in [75], while the latter is done using the same indirect error estimation procedures associated with parametric enhancements in other settings, including those we have alluded to in previous chapters.

Direct error estimation procedures follow this if required, and are designed to follow along the lines previously suggested in Section 5.5. The required calculations are performed together in the `dwr_stochcol_direct_estimator` function. Marking and refinement are performed exactly as in the non-goal-oriented framework, with no significant new functions required to begin subsequent iterations of the adaptive procedure.

When we achieve our required tolerance, we use the same routine as for the hierarchical

product estimation strategy to compute the corrected goal functional, using the function `goafem_doBilinearForm` to calculate the quantity $\mathcal{B}(u_\bullet^{SC}, z_\bullet^{SC})$. As with our algorithms discussed in previous chapters, we can then use the reference script `dwr_referenceSC` to produce additional results that can be used to evaluate the performance of the algorithm we have used.

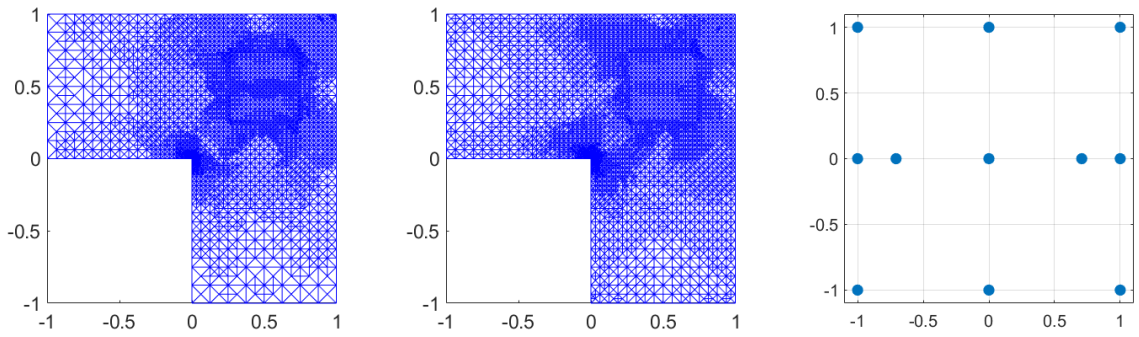
5.7 Experiments

We now complete this chapter with a couple of experiments that are designed to test our algorithm's performance against the results that we attained in the previous chapter. We adopt many of the initial parameters from our experiments in the previous chapter. We maintain the Clenshaw–Curtis rule for generating collocation points associated with our sparse grid, with an initial index set $I_0 = \{\mathbf{1}\}$. Furthermore, we use Dörfler marking techniques with spatial marking parameter $\theta_x = 0.3$ and parametric marking parameter $\theta_{\boldsymbol{y}} = 0.3$. Throughout our work, we will utilise quadrature-based rules to calculate the residuals that we require.

We begin with a familiar example that corresponds to our first test problem that we studied in the previous chapter, namely the computation of the mean of the solution over a small subdomain. We recall that this test problem involves an L-shape domain, D, with subdomain $D_0 = (0.25, 0.75)^2$, while the right-hand side functional in (3.7) admits the input $f(x) = 1$, with the quantity of interest given by equation (4.43). We further retain the affine diffusion coefficient (3.69), with $M = 4$, and the random variables satisfying a uniform probability density function. We also utilise the same initial mesh, as plotted in Figure 4.1 (a).

To test the competitiveness of the DWR method, we opt to run our adaptive algorithm until our approximation requires around the same number of degrees of freedom as our final approximation from the corresponding result in the previous chapter. Specifically, we terminate our algorithm when $N =$ `1e7` in order to see how accurate the resulting approximation of our goal functional is. We attain this target after 80 iterations, at which point, our final approximation has exactly $12,342,915$ degrees of freedom, resulting from $912,111$ elements and 27 collocation points.

The high iteration count is a reflection of the fact that we add comparatively few elements at



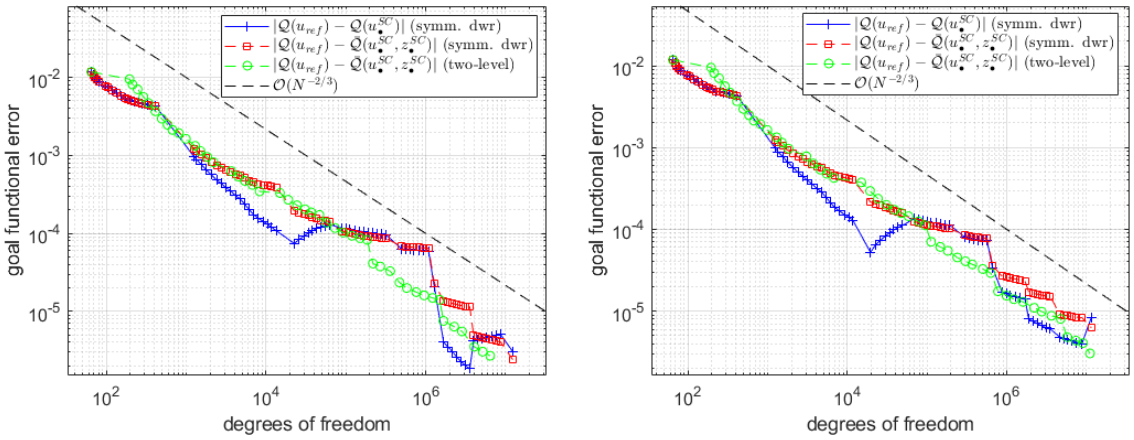
(a) (b) (c)

Figure 5.1: The mesh $\mathcal{T}_{40}$ for the DWR method in (a), and the symmetric DWR method in (b). The final set of collocation points, in the first two directions, for the symmetric DWR method, is given in (c).

each iteration with this method. The additional iterations do cause additional computational expense, and the overall adaptive process takes longer to run than the algorithm utilising the estimator product as a result. Further experimentation, however, shows that this can be mitigated by adjusting the Dörfler marking parameters, particularly the spatial marking parameter, from $\theta_x = 0.3$ to some higher value, which would result in more aggressive spatial refinement.

The primal and dual solutions (which are not plotted here) adopt the same qualitative features as they did when they were calculated in the previous chapter. Of course, this is unsurprising, as our solution step remains largely the same. Of some interest, however, is the resulting mesh and collocation point index set. We plot some examples of these in Figure 5.1.

The refinement patterns exhibited by the standard dual-weighted residual and its symmetric variant are very similar, which is reassuring, as both methods should, to an extent, account for both the primal and dual problems. However, we notice that the symmetric variant of the DWR method appears to produce refinements that are more smoothly distributed, in comparison to the refinement pattern for the original variant, which is more clustered and concentrated by comparison. The symmetric DWR method, theoretically speaking, describes an averaging of the results from taking primal residuals and dual weights, and dual residuals and primal weights, so this apparent smoothing of the resulting refinement pattern is not necessarily a surprise.



(a) (b)

Figure 5.2: A comparison of results for the symmetric DWR method with the results for the estimator product using two-level error estimators, involving the first test problem with 4 random variables in (a), and 8 random variables in (b).

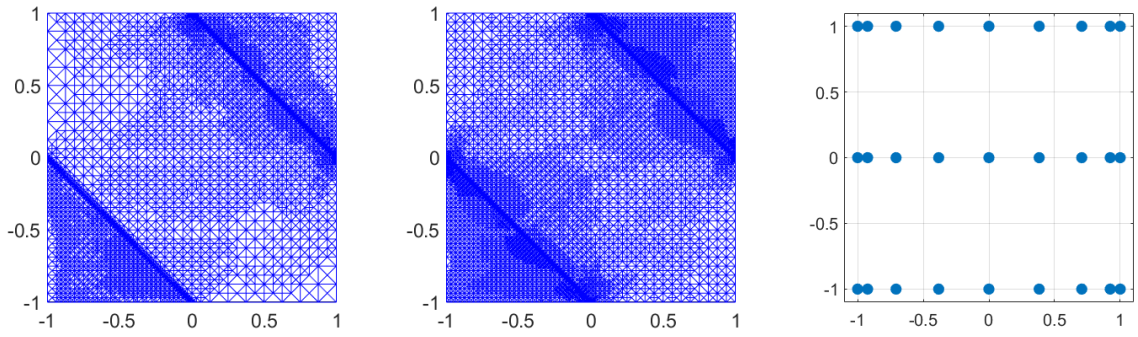
We now consider our primary motivation for this set of experiments, namely, the approximation of the quantity of interest. The approximation we obtain from the symmetric dual-weighted residual is given by 0.12411806 to 8 decimal places (compared with the approximation that we obtained of 0.12411857 in the Section 4.7). This estimate for the goal functional is calculated as having an error approximately equal to $2.3994\mathtt{e}-6$. The error decay in the corrected goal functional is given in Figure 5.2, which we compare with our results from using a two-level error estimation technique to furnish the estimator product strategy we described previously.

The error in the corrected goal functional appears to decrease monotonically, with the only minor cause for concern being the small (relative) rise for the corrected goal functional at around 10^6 degrees of freedom. It is hypothesised that the reason for this is a sub-optimal choice of refinement, possibly resutling from overly coarse estimates of the spatial and parametric contributions to the total error. That being said, our strategy results in an overall convergence rate of $\mathcal{O}(N^{-2/3})$, meaning that it matches the results obtained from the estimator product strategy, with it being competitive throughout most of the adaptive process. We also note, yet again, the stabilisation effect of the correction term, which mirrors what we observed from our results obtained using the estimator product strategy.

Finally, we also plot equivalent results obtained by increasing the number of random variables, M, to 8. As we can see, our conclusions remain the same, with the same convergence rate, and the correction term successfully stabilising our approximation of the goal functional that is obtained using the dual-weighted residual strategy.

For a second experiment, we invoke the second test problem that we described in the previous chapter. This test problem was posed on a large square domain $(-1,1)^2$, and used the alternate representations of the functionals for the primal and dual problems given by definitions (4.45) and (4.46), respectively. Adopting the same definitions of the right-hand side and goal functionals for this specific test problem, we have the functionals (4.49) and (4.50), respectively.

We further retain the quadratic diffusion coefficient (3.72), with random variables being described by a uniform probability density function. We also use the same initial mesh, as given in Figure 4.5 (a).



(a) (b) (c)

Figure 5.3: The mesh $\mathcal{T}_{40}$ for the DWR method (a), and the symmetric DWR method (b). The final set of collocation points, in the first two directions, for the symmetric DWR method, is given in (c).

Motivated by the final approximation obtained in Section 4.7, we run our algorithm until we obtain an approximation with $N = \texttt{3e7}$. The procedure utilising the standard dual-weighted residual method terminates after 72 iterations, where our approximation involves $669,560$ elements and 95 collocation points, with $31,852,265$ degrees of freedom in total. The symmetric dual-weighted residual method, meanwhile, produces an algorithm which terminates after 71 iterations. In this instance, the approximation involves $36,024,761$ degrees of freedom, emanating from $734,601$ elements and 67 collocation points. Again, the primal and dual

solutions are not plotted due to them exhibiting the same qualitative features as what we obtained from the equivalent results in Section 4.7.

For comparison purposes, we plot the meshes for algorithms using the standard dual-weighted residual and the symmetric version, at iteration 40, in Figure 5.3. We observe that, as expected, there is significant refinement on the lines corresponding to non-geometric singularities corresponding to the primal and dual problems, although the refinement here is heavier, relatively speaking, in comparison to the equivalent meshes corresponding to the procedure we utilised in the previous chapter. There is also a significant contrast between the meshes here, with the asymmetry of the mesh produced using standard dual-weighted residuals being particularly noticeable. Use of symmetric dual-weighted residuals yields, as expected, a refinement pattern with rotational symmetry, with a slightly more evenly distributed refinement pattern.

In the case of both the standard dual-weighted residual method and the symmetric variant, the number of collocation points is a reflection of the development of the underlying index set. As can be seen from Figure 5.3 (c), the final index set exhibits a high degree of anisotropy, reaching the fourth level of refinement in the first random parameter, but only reaching the second level in subsequent parameters (this occurred for both the standard and symmetric variants).

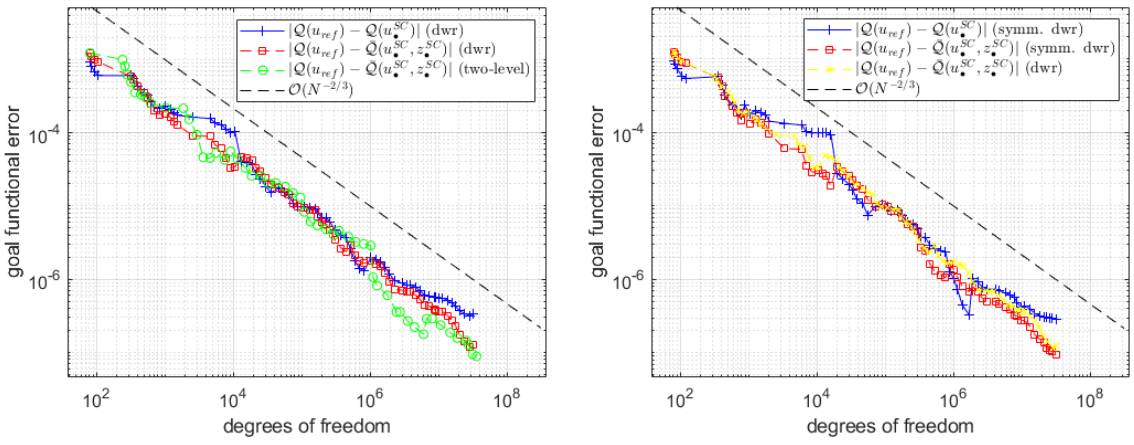


Figure 5.4: A comparison of results for the decay of the error in the goal functional. We compare the dual-weighted residual method with estimator products in (a), while we compare different dual-weighted residual methods in (b).

Our estimates for the goal functional, to 10 decimal places, are -0.0064272164 for the algorithm driven by standard dual-weighted residuals, and -0.0064272492 for the algorithm driven by the symmetric dual-weighted residuals. The calculated errors for these are approximately `1.2607e-7` and `9.3046e-8`, respectively. Recalling that our estimate from the previous chapter was -0.0064272497 (again, to 10 decimal places), we see that the estimates we have generated from these algorithms are suitably accurate.

In Figure 5.4, we look at the error decay in the goal functional for the algorithms driven by standard and symmetric dual-weighted residuals, respectively. We also plot, for reference purposes, the equivalent results attained in the previous chapter. As we can see, the overall error decay appears to be $\mathcal{O}(N^{-2/3})$ for the standard and symmetric variants, with both being able to consistently match the results that we obtained in the previous chapter. Figure 5.4 (b) shows that error exhibited by the symmetric dual-weighted residual appears to be consistently slightly lower than that of the standard variant.

We note that the error decay, in general, appears to contain similar small oscillations that were obtained with the estimator product strategy. Despite this, these oscillations seem to be smaller with the algorithms driven by the dual-weighted residual estimates. While the error in the goal functional is not strictly decreasing, it, at the very least, appears to be a little more stable than the error calculated from using the estimation product.

The correction term assumes the same role that it has done in our previous examples for both the estimator product and the dual-weighted residual strategies, namely, the stabilisation of the error decay. The stabilisation effect appears to be more significant in places for the symmetric dual-weighted residuals, although the reason for the additional instability in the uncorrected goal functional is unknown. In any case, we see that these results that we obtain here are comparable with those obtained for the algorithm utilising the estimator product strategy.

Chapter 6

Concluding Remarks

In recent times, mathematical models involving random or uncertain parameters have become increasingly prominent due to new theoretical knowledge and computing advances. As a result of this, strategies for developing accurate numerical approximations to solutions of parametric PDE problems have also become increasingly important. In this setting, efficient procedures are compulsory in order to minimise the fast-growing computational expense associated with solving problems involving large numbers of random variables. Many contemporary works attempt to mitigate this 'curse of dimensionality' through adaptive algorithms designed to approximate the solutions of these parametric PDEs [109, 110].

With this work, we have considered a steady state parameter-dependent diffusion equation in order to demonstrate a number of results in the stochastic collocation FEM setting. In providing these new results, we have broadened our understanding of adaptive algorithms associated with this method. Our findings include the superiority of using Clenshaw–Curtis abscissas to construct the stochastic collocation FEM approximations used in our work. This is particularly important given the advantages already associated with Clenshaw–Curtis points for quadrature rules, which have increased relevancy within the goal-oriented framework (for example, with the calculation of the correction term). We have also revealed the importance, in some instances, of using a profit-based (or weighted) criterion for refinement of the sparse grid in the context of a parametric adaptivity. In particular, this quickly becomes imperative when the margin of an index set, rather than the reduced margin, is used for the purposes of enriching the sparse grid associated with the stochastic collocation FEM approximation.

Furthermore, while other works have made contributions to goal-oriented error estimation for problems involving parametric PDEs, we have provided significant novelty with our results in this area. To our knowledge, we are the first to make use of reliable compartmentalised spatial and parametric hierarchical indicators from the equivalent non-goal-oriented setting to furnish a newly devised adaptive algorithm in the goal-oriented setting. By doing this, we fully reveal the interplay between primal and dual problems, as well as spatial and parametric contributions to the total error estimate, throughout an adaptive process, in a way that has not been achieved before.

In particular, the novel correction term that we introduced in Section 4.3 reveals the impact of a lack of global orthogonality on goal-oriented error estimation techniques for stochastic collocation FEM. In turn, employing this correction term produces a stabilising effect on the estimate of the goal functional that appears to become more significant for parametric PDE problems involving high levels of volatility.

The algorithms we have described are versatile, with applicability to PDE problems involving coefficients with affine and non-affine parameter dependence. Moreover, we found no deterioration to our convergence results when applying our adaptive strategies to problems which varying numbers of random variables, probability distributions, domains and goal functionals. While we have merely considered a simple elliptic problem here, there are many minor modifications that we can make that allows our findings to be applied to other parametric PDE problems. We highlight some of the modifications and extensions that may be made to our work below.

Throughout our work on goal-oriented adaptivity, we have assumed that the goal functional is linear. Despite this apparent limitation, it is possible to extend our work to a much larger class of goal-functionals $\mathcal{Q} : \mathcal{V} \to \mathbb{R}$, including non-linear goal functionals. Non-linear goal functionals have been considered in the deterministic setting previously, with optimal convergence rates recently proved for quadratic goal functionals in [111], and this represents an important area to expand our current research in the parametric PDE setting. In terms of non-intrusive methods, multilevel Monte Carlo methods have been utilised in the consideration of Frechét differentiable nonlinear functionals in [112]. For intrusive methods, goal-oriented adaptivity for nonlinear functionals has also recently been performed in the stochastic Galerkin setting in [113].

We suggest a similar approach to the stochastic Galerkin setting, considering sample-wise and complete weak formulations which are dependent on the corresponding primal formulations. In the stochastic Galerkin setting, the resulting estimate for the error in the quantity of interest can be written in the form

$$\left|\mathcal{Q}(u) - \bar{\mathcal{Q}}(u_{\bullet}^{SC}, z_{\bullet}^{SC})\right| \lesssim (\mu_{\bullet}^2 + \tau_{\bullet}^2)^{1/2}(\mu_{\bullet}^2 + \tau_{\bullet}^2 + \eta_{\bullet}^2 + \sigma_{\bullet}^2)^{1/2}, \tag{6.1}$$

for suitable error estimates $\mu_{\bullet}, \tau_{\bullet}, \eta_{\bullet}, \sigma_{\bullet}$. While the precise result in the stochastic collocation FEM setting may differ, we hypothesise that it is possible to deduce a similar relationship between primal and dual estimates, and their contribution to the error in the quantity of interest.

We have previously suggested that our work could be expanded to other classes of parametric PDE problem. Indeed, many of the principles described here will extend to a variety of problems that can be written in terms of the sampled weak formulations (3.4) and (4.3), as well as the complete weak formulations (3.7) and (4.2). Some of the possible areas for consideration include problems where the underlying PDEs are more intricate. Time-dependent advection-diffusion processes with dependence on uncertain parameters have been recently considered in the stochastic collocation FEM setting with the work done in [114], for example.

The opportunity to extend the results of our work further is also provided by the incorporation of uncertainty into other inputs of the PDE problem. As noted in Chapter 3, the possibility of using uncertain source terms in the problem formulation has been explored before in [88], as well as others. Further, a number of recent works have investigated parametric PDE problems involving parametrised domains, such as [115] and [116]. These parametrised domains are typically considered as perturbations of Lipschitz 'reference' domains and a strategy involving diffeomorphisms, $\Phi(\boldsymbol{y}) : D \to \mathcal{D}(\boldsymbol{y})$, (see [117]) is then invoked to map the parameter-dependent domain to the reference domain. The additional cost of having more random parameters to contend with is not an obstacle to us, because, as we noted for our experimental results in Section 4.7, increasing the number of random parameters does not affect our conclusions.

Another aspect of complexity we can consider is the introduction of parameter-dependent spatial features into the problem, which, as we suggested in the introduction, can be dealt with using efficient multilevel approximations. For our specific setup (with piecewise linear

approximations, which are the limiting factor here), the optimal convergence rate is $\mathcal{O}(N^{-1/2})$ in the non-goal-oriented setting. It is known that when applied to problems such as the 'one peak problem' seen in [42] (and originally introduced in [118] to test adaptivity in the Monte Carlo setting), multilevel adaptivity can achieve this optimal rate. In the case of the estimator product strategy, the general principles described in [42] can easily be applied to our work with minor modifications to the adaptive algorithm. In the case of the dual-weighted residual approach, small modifications are also required to some of the theoretical developments (such as the distributing of sums over collocation points into the residual in inequality (5.18) to obtain indicators that are collocation point dependent), although this does not pose a significant challenge.

There are still some open problems related to our work on goal-oriented adaptivity that must be addressed. Firstly, for the purposes of our theoretical work, we have focused on proving upper bounds for the error in the quantity of interest. Our work on efficiency has been restricted to the demonstration of effectivity indices, as we do not have any theoretical proof of lower bounds associated with our error estimation strategies. In fact, as evidenced by our second test problem from Section 4.7, and the surrounding discussion, there are some classes of problems that are difficult to estimate efficiently. Further, as noted in [72], we know that even in the deterministic setting, a goal functional may satisfy $Q(u) - Q(u_h) = 0$ when $u \neq u_h$. We similarly observe from inequality (4.34) that we can obtain $\mathcal{Q}(u) - \bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC}) = 0$ with $u \neq u_\bullet^{SC}$, so efficiency for goal-oriented procedures is likely to be very difficult to quantify precisely from a theoretical perspective.

Another area where work is sorely is needed on the establishing of convergence results for adaptive algorithms in the stochastic collocation FEM setting. As noted in the introduction, convergence is relatively well understood in the deterministic setting, and given the results of [42], it is expected that when multilevel adaptivity is used, the optimal rate of $\mathcal{O}(N^{-1})$ can be recovered in the goal-oriented setting for certain problems. Despite this, more general theoretical results remain elusive and will necessarily be a subject for future works in this area. On the other hand, the algorithm analysed in [43] represents a very close non-goal-oriented analogue to our procedures. Although this work does not reveal information about optimal rates, the general results that are demonstrated here can also be adapted to our work.

Bibliography

[1] A. Fick. Ueber diffusion. *Annalen der Physik*, 170(1):59–86, 1855.

[2] L. D. Landau and E. M. Lifshitz. *Fluid Mechanics: Volume 6*. Number v.6 in ABC. Elsevier Science, 1987.

[3] E. Schrödinger. An undulatory theory of the mechanics of atoms and molecules. *Phys. Rev.*, 28:1049–1070, 1926.

[4] P. Wilmott, S. Howison, and J. Dewynne. *The Mathematics of Financial Derivatives: A Student Introduction*. The Mathematics of Financial Derivatives: A Student Introduction. Cambridge University Press, 1995.

[5] D. Zhang. *Stochastic Methods for Flow in Porous Media: Coping with Uncertainties*. Elsevier Science, 2001.

[6] G. Lord, C. Powell, and T. Shardlow. *An Introduction to Computational Stochastic PDEs*. Cambridge University Press, 2014.

[7] N. Wiener. The homogeneous chaos. *American Journal of Mathematics*, 60:897, 1938.

[8] D. Xiu and G. Karniadakis. Modeling uncertainty in steady state diffusion problems via generalized polynomial chaos. *Computer Methods in Applied Mechanics and Engineering*, 191:4927–4948, 2002.

[9] M. Loeve. *Probability Theory II*. Graduate Texts in Mathematics. Springer, 1978.

[10] I. Graham, R. Scheichl, and E. Ullmann. Mixed finite element analysis of lognormal diffusion and multilevel Monte Carlo methods. *Stochastics and Partial Differential Equations Analysis and Computations*, 4:41–75, 2013.

[11] O. Ernst, B. Sprungk, and L. Tamellini. Convergence of sparse collocation for functions of countably many Gaussian random variables - with application to lognormal elliptic diffusion problems. *SIAM Journal on Numerical Analysis*, 56:877–905, 2016.

[12] P. G. Ciarlet. *The Finite Element Method for Elliptic Problems*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 2002.

[13] G. Strang and G. Fix. *An Analysis of the Finite Element Method*. Wellesley-Cambridge Press, 2008.

[14] S. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Texts in Applied Mathematics. Springer New York, 2002.

[15] W. Dörfler. A convergent adaptive algorithm for Poisson's equation. *SIAM Journal on Numerical Analysis*, 33(3):1106–1124, 1996.

[16] P. Morin, R. H. Nochetto, and K. G. Siebert. Data oscillation and convergence of adaptive FEM. *SIAM Journal on Numerical Analysis*, 38(2):466–488, 2000.

[17] R. Stevenson. An optimal adaptive finite element method. *SIAM Journal on Numerical Analysis*, 42(5):2188–2217, 2005.

[18] P. Binev, W. Dahmen, and R. DeVore. Adaptive finite element methods with convergence rates. *Numerische Mathematik*, 97:219–268, 2004.

[19] J. M. Cascon, C. Kreuzer, R. H. Nochetto, and K. G. Siebert. Quasi-optimal convergence rate for an adaptive finite element method. *SIAM Journal on Numerical Analysis*, 46(5):2524–2550, 2008.

[20] A. Cohen, R. DeVore, and R. Nochetto. Convergence rates of AFEM with H^{-1} data. *Foundations of Computational Mathematics*, 12(5):671–718, 2012.

[21] R.G. Ghanem and P.D. Spanos. *Stochastic Finite Elements: A Spectral Approach*. Springer New York, 1991.

[22] M. Deb, I. Babuška, and J. Oden. Solution of stochastic partial differential equations using Galerkin finite element techniques. *Computer Methods in Applied Mechanics and Engineering*, 190:6359–6372, 2001.

[23] I. Babuska, R. Tempone, and G. Zouraris. Galerkin finite element approximations of stochastic elliptic partial differential equations. *SIAM J. Numerical Analysis*, 42:800–825, 2004.

[24] N. Metropolis. The beginning of the Monte Carlo method. In *Los Alamos Science Special Issue*, pages 125–130, 1987.

[25] R.Y. Rubinstein and D.P. Kroese. *Simulation and the Monte Carlo Method*. Wiley Series in Probability and Statistics. Wiley, 2016.

[26] R. E. Caflisch. Monte Carlo and quasi-Monte Carlo methods. *Acta Numerica*, 7:1–49, 1998.

[27] M. Giles. Multilevel Monte Carlo path simulation. *Operations Research*, 56:607–617, 2008.

[28] L. Mathelin, M. Hussaini, and T. Zang. Stochastic approaches to uncertainty quantification in CFD simulations. *Numerical Algorithms*, 38:209–236, 2005.

[29] D. Xiu and J. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM J. Scientific Computing*, 27:1118–1139, 2005.

[30] I. Babuška, F. Nobile, and R. Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 45(3):1005–1034, 2007.

[31] F. Nobile, R. Tempone, and C. G. Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 46(5):2309–2345, 2008.

[32] S. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Dokl. Akad. Nauk SSSR*, 148:1042–10455, 1963.

[33] F. Nobile, R. Tempone, and C. Webster. An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM J. Numerical Analysis*, 46:2411–2442, 2008.

[34] A. Chernov. Sparse polynomial approximation in positive order Sobolev spaces with bounded mixed derivatives and applications to elliptic problems with random loading. *Applied Numerical Mathematics*, 62(4):360–377, 2012. Third Chilean Workshop on Numerical Analysis of Partial Differential Equations (WONAPDE 2010).

[35] C. Schwab and R. Todor. Sparse finite elements for elliptic problems with stochastic loading. *Numerische Mathematik*, 95(4):707–734, 2003.

[36] P. Cioica, S. Dahlke, N. Döhring, S. Kinzel, F. Lindner, T. Raasch, K. Ritter, and R. Schilling. Adaptive wavelet methods for the stochastic Poisson equation. *BIT Numerical Mathematics*, 52(3):589– 614, 2012.

[37] D. Guignard and F. Nobile. A posteriori error estimation for the stochastic collocation finite element method. *SIAM Journal on Numerical Analysis*, 56(5):3121–3143, 2018.

[38] T. Gerstner and M. Griebel. Dimension-adaptive tensor-product quadrature. *Computing*, 71:65–87, 2003.

[39] M. Feischl and A. Scaglioni. Convergence of adaptive stochastic collocation with finite elements. *Computers & Mathematics with Applications*, 98:139–156, 2021.

[40] M. Eigel, O. Ernst, B. Sprungk, and L. Tamellini. On the convergence of adaptive stochastic collocation for elliptic partial differential equations with affine diffusion. *SIAM Journal on Numerical Analysis*, 60:659–687, 2022.

[41] A. Bespalov, D. Silvester, and F. Xu. Error estimation and adaptivity for stochastic collocation finite elements part I: Single-level approximation. *SIAM Journal on Scientific Computing*, 44(5):A3393–A3412, 2022.

[42] A. Bespalov and D. Silvester. Error estimation and adaptivity for stochastic collocation finite elements part II: Multilevel approximation. *SIAM Journal on Scientific Computing*, 45(2):A781–A797, 2023.

[43] A. Bespalov and A. Savinov. Convergence analysis of the adaptive stochastic collocation finite element method. *Preprint, arXiv: 2401.14894*, 2024.

[44] R. Becker and R. Rannacher. Weighted a posteriori error control in FE methods. In *Proc. ENUMATH'97*, 1997.

[45] R. Rannacher and F.-T. Suttmeier. A feed-back approach to error control in finite element methods: application to linear elasticity. *Computational Mechanics*, 19:434–446, 1997.

[46] S. Prudhomme and J.T. Oden. On goal-oriented error estimation for elliptic problems: application to the control of pointwise errors. *Computer Methods in Applied Mechanics and Engineering*, 176(1):313–331, 1999.

[47] J.T. Oden and S. Prudhomme. Goal-oriented error estimation and adaptivity for the finite element method. *Computers & Mathematics with Applications*, 41(5):735 – 756, 2001.

[48] W. Dahmen, A. Kunoth, and J. Vorloeper. Convergence of adaptive wavelet methods for goal-oriented error estimation. In *Numerical Mathematics and Advanced Applications*, pages 39–61. Springer Berlin Heidelberg, 2006.

[49] K. Moon, E. von Schwerin, A. Szepessy, and R. Tempone. Convergence rates for an adaptive dual weighted residual finite element algorithm. *BIT*, 46:367–407, 2006.

[50] M. Mommer and R. Stevenson. A goal-oriented adaptive finite element method with convergence rates. *SIAM J. Numerical Analysis*, 47:861–886, 2009.

[51] R. Becker, M. Brunner, M. Innerberger, J. Melenk, and D. Praetorius. Rate-optimal goal-oriented adaptive FEM for semilinear elliptic PDEs. *Computers & Mathematics with Applications*, 118:18–35, 2022.

[52] L. Mathelin and O. Le Maître. Dual-based a posteriori error estimate for stochastic finite element methods. *Comm. Appl. Math. Comp. Sci.*, 2:83–115, 2007.

[53] T. Butler, C. Dawson, and T. Wildey. A posteriori error analysis of stochastic differential equations using polynomial chaos expansions. *SIAM Journal on Scientific Computing*, 33:1267–1291, 2011.

[54] A. Bespalov, D. Praetorius, L. Rocchi, and M. Ruggeri. Goal-oriented error estimation and adaptivity for elliptic PDEs with parametric or uncertain inputs. *Computer Methods in Applied Mechanics and Engineering*, 345:951–982, Mar 2019.

[55] M. Eigel, C. Merdon, and J. Neumann. An adaptive multilevel Monte Carlo method with stochastic bounds for quantities of interest with uncertain data. *SIAM/ASA Journal on Uncertainty Quantification*, 4:1219–1245, 2016.

[56] R. C. Almeida and J. T. Oden. Solution verification, goal-oriented adaptive methods for stochastic advection–diffusion problems. *Computer Methods in Applied Mechanics and Engineering*, 199(37):2472–2486, 2010.

[57] M. Giles and E. Süli. Adjoint methods for PDEs: A posteriori error analysis and postprocessing by duality. *Acta Numerica*, 11:145–236, 01 2002.

[58] N. Pierce and M. Giles. Adjoint recovery of superconvergent functionals from PDE approximations. siam rev., in press. *SIAM Review*, 42:247–264, 2000.

[59] J. N. Reddy. *An Introduction to the Finite Element Method*. McGraw-Hill series in mechanical engineering. McGraw-Hill, 2006.

[60] R.A. Adams and J.J.F. Fournier. *Sobolev Spaces*. ISSN. Elsevier Science, 2003.

[61] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2009.

[62] W. Rudin. *Real and Complex Analysis*. Mathematics series. McGraw-Hill, 1987.

[63] P. Knabner and L. Angerman. *Numerical Methods for Elliptic and Parabolic Partial Differential Equations*. Texts in Applied Mathematics. Springer New York, 2006.

[64] H. Brezis. *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Universitext. Springer New York, 2010.

[65] B. Dacorogna. *Introduction to the Calculus of Variations*. Information and Interdisciplinary Subjects Series. Imperial College Press, 2004.

[66] W.A. Strauss. *Partial Differential Equations: An Introduction*. Wiley, 2007.

[67] P. D. Lax and A. N. Milgram. Parabolic equations. *Contributions to the Theory of Partial Differential Equations. (AM-33)*, pages 167–190, 1954.

[68] K. Inayatnoor and M. Aslam Noor. A generalization of the Lax-Milgram lemma. *Canadian Journal of Mathematics*, 23(2):179–184, 1980.

[69] B. Szabó, A. Düster, and E. Rank. The p-version of the finite element method. *Siam Journal on Numerical Analysis*, 18:119–139, 11 2004.

[70] I. Babuška and M. Dorr. Error estimates for the combined h and p versions of the finite element method. *Numerische Mathematik*, 37:257–277, 1981.

[71] J. Céa. Approximation variationnelle des problèmes aux limites. *Annales de l'institut Fourier*, 14(2):345–444, 1964.

[72] R. Verfürth. *A Posteriori Error Estimation Techniques for Finite Element Methods*. Numerical Mathematics and Scientific Computation. OUP Oxford, 2013.

[73] R. H. Nochetto and A. Veeser. Primer of adaptive finite element methods. *Multiscale and Adaptivity: Modeling, Numerics and Applications: C.I.M.E. Summer School, Cetraro, Italy 2009*, pages 125 –225, 2012.

[74] V. Eijkhout and P. Vassilevski. The role of the strengthened Cauchy–Buniakowskii –Schwarz inequality in multilevel methods. *SIAM review, 33 (3)*, pages 405–419, 1991.

[75] R. Bank and A. Weiser. Some a posteriori error estimators for elliptic partial differential equations. *Mathematics of Computation*, 44:283–301, 1985.

[76] P. Mund, E. P. Stephan, and J. Weiße. Two-level methods for the single layer potential in R3. *Computing*, 60(3):243–266, 1998.

[77] I. Babuška and M. Vogelius. Feedback and adaptive finite element solution of one-dimensional boundary value problems. *Numerische Mathematik*, 44:75–102, 1984.

[78] W. F. Mitchell. A comparison of adaptive refinement techniques for elliptic problems. *ACM Trans. Math. Softw.*, 15(4):326–347, December 1989.

[79] E. Bänsch. Local mesh refinement in 2 and 3 dimensions. *IMPACT of Computing in Science and Engineering*, 3(3):181–191, 1991.

[80] W. Gong, H. Liu, and N. Yan. Adaptive finite element method for parabolic equations with Dirac measure. *Computer Methods in Applied Mechanics and Engineering*, 328:217–241, 2018.

[81] Z. Chen and F. Jia. An adaptive finite element algorithm with reliable and efficient error control for linear parabolic problems. *Math. Comput.*, 73:1167–1193, 2004.

[82] A. Bespalov, L. Rocchi, and D. Silvester. T-IFISS: a toolbox for adaptive FEM computation. *Computers & Mathematics with Applications*, 2020.

[83] M. Ainsworth and J. T. Oden. *A Posteriori Error Estimation in Finite Element Analysis*. John Wiley & Sons, Ltd, 2000.

[84] P. Mund and E. P. Stephan. An adaptive two-level method for the coupling of nonlinear FEM-BEM equations. *SIAM Journal on Numerical Analysis*, 36(4):1001–1021, 1999.

[85] M. Feischl, D. Praetorius, and K. van der Zee. An abstract analysis of optimal goal-oriented adaptivity. *SIAM Journal on Numerical Analysis*, 2015.

[86] W. Bangerth and R. Rannacher. *Adaptive Finite Element Methods for Differential Equations*. Lectures in Mathematics. ETH Zürich. Birkhäuser Basel, 2013.

[87] R. Durrett. *Probability: Theory and Examples*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2010.

[88] M. Gunzburger, C. Webster, and G. Zhang. Stochastic finite element methods for partial differential equations with random input data. *Acta Numerica*, 23:521–650, 2014.

[89] T. Hytönen, J. van Neerven, M. Veraar, and L. Weis. *Analysis in Banach Spaces: Volume I: Martingales and Littlewood-Paley Theory*. Ergebnisse der Mathematik und ihrer Grenzgebiete. 3. Folge / A Series of Modern Surveys in Mathematics. Springer International Publishing, 2016.

[90] G. Fishman. *Monte Carlo: Concepts, Algorithms, and Applications*. Springer Series in Operations Research and Financial Engineering. Springer New York, 2013.

[91] W.H. Press, S.A. Teukolsky, B.P. Flannery, and W.T. Vetterling. *Numerical Recipes in FORTRAN 77: Volume 1, Volume 1 of Fortran Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 1992.

[92] P. Glasserman. *Monte Carlo Methods in Financial Engineering*. Applications of mathematics: stochastic modelling and applied probability. Springer, 2004.

[93] I. Sobol. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967.

[94] J. H. Halton. Algorithm 247: Radical-inverse quasi-random point sequence. *Commun. ACM*, 7(12):701–702, 1964.

[95] A. Barth, C. Schwab, and N. Zollinger. Multi-level Monte Carlo finite element method for elliptic PDEs with stochastic coefficients. *Numerische Mathematik*, 119:123–161, 2011.

[96] M. Giles. Multilevel Monte Carlo path simulation. *Operations Research*, 56:607–617, 06 2008.

[97] A. Bespalov, D. Loghin, and R. Youngnoi. Truncation preconditioners for stochastic Galerkin finite element discretizations. *SIAM Journal on Scientific Computing*, 43(5):S92–S116, 2021.

[98] M. Eigel, M. Marschall, M. Pfeffer, and R. Schneider. Adaptive stochastic Galerkin FEM for lognormal coefficients in hierarchical tensor representations. *Numerische Mathematik*, 145:765–803, 2020.

[99] A. Bespalov and F. Xu. A posteriori error estimation and adaptivity in stochastic Galerkin FEM for parametric elliptic PDEs: Beyond the affine case. *Computers & Mathematics with Applications*, 80:1084–1103, 2020.

[100] V. Hoang and C. Schwab. High-dimensional finite elements for elliptic problems with multiple scales. *Multiscale Modeling & Simulation*, 3(1):168–194, 2005.

[101] V. Barthelmann, E. Novak, and K. Ritter. High dimensional polynomial interpolation on sparse grids. *Adv. Comput. Math.*, 12:273–288, 2000.

[102] F. Leja. Sur certaines suites liées aux ensembles plans et leur application à la représentation conforme. *Annales Polonici Mathematici*, 4:8–13, 1957.

[103] F. Nobile, L. Tamellini, and R. Tempone. Comparison of Clenshaw–Curtis and Leja quasi-optimal sparse grids for the approximation of random PDEs. In R. Kirby, M. Berzins, and J. Hesthaven, editors, *Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2014*, pages 475–482, Cham, 2015. Springer International Publishing.

[104] A. Narayan and J. Jakeman. Adaptive Leja sparse grid constructions for stochastic collocation and high-dimensional approximation. *SIAM Journal on Scientific Computing*, 36(6):A2952–A2983, 2014.

[105] C. Clenshaw and A. Curtis. A method for numerical integration on an automatic computer. *Numerische Mathematik*, 2:197–205, 1960.

[106] D. Loukrezis, U. Römer, and H. De Gersem. Numerical comparison of Leja and Clenshaw–Curtis dimension-adaptive collocation for stochastic parametric electromagnetic field problems. *CoRR*, abs/1712.07223, 2017.

[107] M. Eigel, C. Gittelson, C. Schwab, and E. Zander. Adaptive stochastic Galerkin FEM. *Computer Methods in Applied Mechanics and Engineering*, 270:247–269, 2014.

[108] F. Nobile, L. Tamellini, F. Tesei, and R. Tempone. An adaptive sparse grid algorithm for elliptic PDEs with lognormal diffusion coefficient. In *Sparse Grids and Applications - Stuttgart 2014*, pages 191–220, 2016.

[109] A. Cohen and R. DeVore. Approximation of high-dimensional parametric PDEs. *Acta Numerica*, 24:1–159, 2015.

[110] A. Bespalov, D. Praetorius, L. Rocchi, and M. Ruggeri. Convergence of adaptive stochastic Galerkin FEM. *SIAM Journal on Numerical Analysis*, 57:2359–2382, 01 2019.

[111] R. Becker, M. Innerberger, and D. Praetorius. Optimal convergence rates for goal-oriented FEM with quadratic goal functional. *Computational Methods in Applied Mathematics*, 21:267–288, 2020.

[112] A. L. Teckentrup, R. Scheichl, M. Giles, and E. Ullmann. Further analysis of multilevel Monte Carlo methods for elliptic PDEs with random coefficients. *Numerische Mathematik*, 125:569–600, 2013.

[113] A. Bespalov, D. Praetorius, and M. Ruggeri. Goal-oriented adaptivity for multilevel stochastic Galerkin FEM with nonlinear goal functionals. *Preprint, arXiv:2208.09388*, 2023.

[114] B. Kent, C. Powell, D. Silvester, and M. Zimoń. Efficient adaptive stochastic collocation strategies for advection–diffusion problems with uncertain inputs. *Journal of Scientific Computing*, 96:64–96, 2023.

[115] R. Hiptmair, L. Scarabosio, C. Schillings, and C. Schwab. Large deformation shape uncertainty quantification in acoustic scattering. *Advances in Computational Mathematics*, 44:1475–1518, 2018.

[116] J. Castrillón-Candás, F. Nobile, and R. Tempone. Analytic regularity and collocation approximation for elliptic PDEs with random domain deformations. *Computers & Mathematics with Applications*, 71(6):1173–1197, March 2016.

[117] D. Xiu and D. M. Tartakovsky. Numerical methods for differential equations in random domains. *SIAM J. Sci. Comput.*, 28(3):1167–1185, 2006.

[118] R. Kornhuber and E. Youett. Adaptive multilevel Monte Carlo methods for stochastic variational inequalities, 2017.

Appendix A

Glossary of Mathematical Notation

In this appendix, we present a list of notation in alphabetical order, with definitions and the section in which each symbol is introduced. For ease of use, and brevity, we omit some symbols which are only introduced temporarily, or are otherwise easy to recognise using rules and conventions established for other symbols and notation.

Symbol	Definition	First Introduced
$\mathbb{1}_{(\cdot)}$	Boolean quantity	Section 4.5
$\lvert\cdot\rvert$	Absolute value (real numbers), component sum (vectors), area (domains), or cardinality (sets)	Section 2.3
$a.e.$	Almost everywhere (with respect to a given measure)	Section 3.1
$*$	Final quantities from an adaptive algorithm (e.g. the final approximation $u_{\mathcal{T}}^* \in V_{\mathcal{T}}^*$)	Section 2.3
$B(\cdot,\cdot)$	Bilinear form associated with deterministic weak formulations (see also: $F(\cdot)$, $Q(\cdot)$)	Section 2.2
$B_{\bar{\boldsymbol{y}}}(\cdot,\cdot)$	Sampled bilinear form taken with sampled diffusion coefficient $a(x,\bar{\boldsymbol{y}})$	Section 2.2
$\mathcal{B}(\cdot,\cdot)$	Bilinear form associated with complete weak formulations (see also $\mathcal{F}(\cdot)$, $\mathcal{Q}(\cdot)$)	Section 3.1
$\mathcal{B}(\Gamma)$	Borel σ-algebra of Γ (event space)	Section 3.1
$[\![\cdot]\!]$	Jump of a function in $\mathcal{S}_0^\infty(\mathcal{T})$	Section 2.4

$\bar{(\cdot)}$	Quantities produced by modifications of other quantities	Section 2.2
$\bullet$	Discrete quantities[1] (reference to the underlying triangulation (e.g. $\mathcal{T}$) is used in Chapter 2)	Section 3.4
$C^\infty(D)$	Space of continuously infinitely continuously differentiable functions	Section 2.1
$\underline{C}^\infty(D)$	Subspace of functions in $C^\infty(D)$ with compact support	Section 2.1
$\chi_{(\cdot)}$	Characteristic function	Section 4.7
D	Spatial domain, typically Lipschitz (see also: ∂D)	Section 2.1
$\mathcal{D}(\boldsymbol{y})$	Parametrised spatial domain	Section 6.0
$\mathrm{d}x$	Lebesgue measure	Section 2.1
$\Delta^{\boldsymbol{i}}$	Surplus operator constructed from 1D Lagrange interpolation operators (see also: $\mathcal{U}^{\boldsymbol{i}}$)	Section 3.3
δ_{ij}	Kronecker delta corresponding to indexed quantities i, j	Section 2.3
E	Edges contained within a triangulation (see also $\mathcal{E}_\mathcal{T}$)	Section 2.4
$\mathcal{E}_\mathcal{T}$	Skeleton, or set of edges in a triangulation $\mathcal{T}$	Section 2.4
$\operatorname*{ess\,sup}_P (\cdot)$	Essential supremum (supremum a.e. where 'P' is true, also $\operatorname*{ess\,inf}_P (\cdot)$ for the essential infimum)	Section 2.1
η	Global (spatial) error estimate corresponding to a dual problem (see also: μ_K)	Section 2.8
ε	A small quantity (esp. an error tolerance), $\varepsilon \ll 1$	Section 2.3
$F(\cdot)$	Right-hand side source/forcing functional associated with deterministic weak formulations (see also: $B(\cdot,\cdot)$)	Section 2.2
$\mathcal{F}(\cdot)$	Right-hand side source/forcing functional associated with complete weak formulations (see also: $\mathcal{B}(\cdot,\cdot)$)	Section 3.1
Γ	Parameter domain	Section 3.1
$H^1(D)$	Sobolev space of L^2-functions with square integrable weak derivatives (see also: V)	Section 2.1
$H^1_0(D)$	Subspace of functions in $H^1_0(D)$ with compact support in D	Section 2.1
$\widehat{(\cdot)}$	Quantities related to e.g. spatial refinement (similarly, $\widetilde{(\cdot)}$ for coarsening)	Section 2.4
I	Sparse grid index set (also $I_\bullet$ et al.)	Section 3.3
K	Elements of a conforming spatial triangulation, $\mathcal{T}$	Section 2.2

[1] sometimes omitted for brevity; this is noted where necessary

$L_{\boldsymbol{m}}$	Lagrange basis functions associated with with collocation points (later denoted $L_{\bar{\boldsymbol{y}}}$)	Section 3.3
$L^2(D)$	Lebesgue space of square integrable functions	Section 2.1
$L^p_\rho(\Gamma; V)$	Bochner space of ρ-measurable functions $v : \Gamma \to V$	Section 3.1
$\mathbb{N}$	Set of natural numbers (by convention, excluding 0	Section 2.2
$\mathbb{N}_0$	Set $\mathbb{N} \cup \{0\}$ (the natural numbers and 0)	Section 2.2
M	Parameter space dimension (see also: Γ)	Section 3.1
$\mathcal{M}$	Marked sets generated using local indicators (also $\mathcal{M}_\bullet$)	Section 2.3
$\mathcal{M}_I$	Margin of an index set I (see also: $\mathcal{R}_I$)	Section 3.3
μ	Global (spatial) error estimate corresponding to a primal problem (also $\mu_\bullet$ et al.)	Section 2.3
μ_K	Local spatial error indicators corresponding to a primal problem (all other indicators defined analogously with relevant subscripts)	Section 2.3
$\mathcal{N}_{(\cdot)}$	Set of nodes associated with an underlying spatial object (e.g. a triangulation, $\mathcal{T}$, or an edge, E)	Section 2.3
$N_{\mathcal{T}}$	Total number of degrees of freedom associated with a triangulation, $\mathcal{T}$	Section 2.3
$\boldsymbol{n}$	Outward unit vector (also $\boldsymbol{n}_E$, $\boldsymbol{n}_K$)	Section 2.4
∇	Gradient operator	Section 2.2
$\|\cdot\|_D$	L^2 norm where D is a spatial domain	Section 2.1
$\|\cdot\|_{\Gamma;V}$	Bochner norm for the space $L^p_\rho(\Gamma; V)$ (alternatively, e.g. $\|\cdot\|_{\mathcal{V}}$ for the Bochner space $\mathcal{V}$)	Section 3.1
$\|\|\|\cdot\|\|\|$	Energy norm induced by a bilinear form	Section 2.2
ω_K	Weights for dual-weighted residuals (see also: μ_K)	Section 2.9
$\omega_{\bar{\boldsymbol{y}}}$	Weights associated with a quadrature-based calculation	Section 4.4
φ_j	Finite element basis function corresponding to a node j for a given spatial discretisation	Section 2.3
$\psi_{\mathcal{T}}$	Arbitrary function, $\psi_{\mathcal{T}} \in V_{\mathcal{T}}$ (also $\psi_\bullet \in V_\bullet$)	Section 2.9
∂D	Boundary of a spatial domain, D	Section 2.2
$Q(\cdot)$	Goal functional associated with a dual problem (see also $(B(\cdot,\cdot))$ and $F(\cdot)$)	Section 2.8

$\mathcal{Q}(\cdot)$	Goal functional associated with a complete dual formulation	Section 4.1
$\bar{\mathcal{Q}}(u_\bullet^{SC}, z_\bullet^{SC})$	Corrected goal functional	Section 4.3
$\mathbb{R}$	Set of real numbers (also $\mathbb{R}^n$ for n-dimensional vectors)	Section 2.1
$\mathfrak{R}$	Residual-based quantities	Section 2.4
$\mathcal{R}_I$	Reduced margin of an index set I (see also: $\mathcal{M}_I$)	Section 3.3
$\rho(\boldsymbol{y})$	Probability density function	Section 2.4
ρ_K	Localised per-element residual indicators (also ρ_K^u, ρ_K^z; see also: μ_K)	Section 2.9
S_I	Smolyak sparse grid interpolant	Section 3.3
$\mathcal{S}_0^p(\mathcal{T})$	Space of piecewise polynomials degree p (piecewise on pieces $K \in \mathcal{T}$)	Section 2.2
$\mathcal{S}_0^\infty(\mathcal{T})$	Space of piecewise continuous functions in $H_0^1(D)$ (piecewise on pieces $K \in \mathcal{T}$)	Section 2.2
$\sigma_\bullet$	Global parametric error estimate corresponding to a dual problem (see also: μ_K)	Section 4.5
$\sigma_\cdot$	Standard deviation (subscript denotes the relevant quantity)	Section 3.7
$\mathcal{T}$	Spatial mesh triangulations (also $\mathcal{T}_\bullet$)	Section 2.2
$\tau_\bullet$	Global parametric error estimate corresponding to a primal problem (see also: μ_K)	Section 3.5
Θ	Effectivity indices (also $\Theta_\bullet$)	Section 2.4
θ	Marking parameters (also θ_x, $\theta_{\boldsymbol{y}}$)	Section 2.5
$\widetilde{(\cdot)}$	Quantities related to e.g. parametric refinements	Section 3.5
$\otimes$	Tensor product	Section 3.2
V	Linear vector space (associated with spatial domains)	Section 2.1
u	Solution to (primal) problem formulation	Section 2.2
$u_\bullet^{SC}$	Stochastic collocation FEM approximation for a primal problem (also u^{SC} et al.)	Section 3.2
$u_\star^{SC}$	Enhanced stochastic collocation FEM approximations (use of $\star$ not to be confused with $*$)	Section 3.5
$u_{\bullet\bar{\boldsymbol{y}}}$	Sampled finite element approximations $u_{\bullet\bar{\boldsymbol{y}}} \in V_{\bullet\bar{\boldsymbol{y}}} \subset V$	Section 3.4

$\mathcal{U}^{\boldsymbol{i}}$	One-dimensional Lagrange interpolant corresponding to a given index $\boldsymbol{i} \in \mathbb{N}$	Section 3.3
V'	Dual space of linear vector space, V	Section 2.1
$V_{\mathcal{T}}$	Discrete finite element space (deterministic setting)	Section 2.2
$\mathcal{V}$	Finite space associated with the complete weak formulation (see also $L^p_\rho(\Gamma; V)$)	Section 3.1
x	Spatial variable (also $\boldsymbol{x}$, in Chapter 2 only)	Section 2.1
Y	Collocation point set (also $Y_\bullet$ et al.)	Section 3.2
$\boldsymbol{y}$	Random variable in Γ	Section 3.1
$\bar{\boldsymbol{y}}$	Random sample or collocation point (from $Y, Y_\bullet$ et al.)	Section 3.1
z	Solution to dual problem (see also: u; replacing u by z in most definitions yields a dual-related quantity, e.g. $z_\bullet^{SC}$)	Section 2.8