Proper Generalised Decompositions: Theory and Applications

Thomas Lloyd David Croft

School of Mathematics

Cardiff University

A thesis submitted for the degree of

Doctor of Philosophy

9th April 2015

Summary

In this thesis a recently proposed method for the efficient approximation of solutions to high-dimensional partial differential equations has been investigated. This method, known as the Proper Generalised Decomposition (PGD), seeks a separated representation of the unknown field which leads to the solution of a series of low-dimensional problems instead of a single high-dimensional problem. This effectively bypasses the computational issue known as the 'curse of dimensionality'.

The PGD and its recent developments are reviewed and we present results for both the Poisson and Stokes problems. Furthermore, we investigate convergence of PGD algorithms by comparing them to greedy algorithms which have previously been studied in the non-linear approximation community. We highlight that convergence of PGD algorithms is not guaranteed when a Galerkin formulation of the problem is considered. Furthermore, it is shown that stability conditions related to weakly coercive problems (such as the Stokes problem) are not guaranteed to hold when employing a PGD approximation.

PGD algorithms based on rigorously derived least-squares formulations are developed and it is shown that convergence of associated greedy algorithms is guaranteed. These formulations also have the added benefit that they remove the requirement to satisfy stability conditions related to weakly coercive problems. A variety of least-squares formulations are derived based on different first-order reformulations of the problems and a thorough comparison is made. The least-squares PGD algorithms developed in this research are applied once again to the Poisson and Stokes problems as well as the non-symmetric convection-diffusion equation.

Finally, an application of the PGD to a deterministic approach to kinetic theory models in polymer rheology is considered. This involves solving the (potentially high-dimensional) Fokker-Planck equation. Results are provided for a spatially homogeneous form of the Fokker-Planck equation and streamline upwinding is employed to stabilise the numerical solutions. A method recently proposed for solving the fully non-homogeneous Fokker-Planck equation is investigated which uses an operator splitting technique. It is shown that this approach is not suitable to be applied in conjunction with the PGD and instead two different schemes for solving this problem are proposed.

Declaration

This work has not been submitted in substance for any other degree or award at this or any other university or place of learning, nor is being submitted concurrently in candidature for any degree or other award.

Signed Date

STATEMENT 1

This thesis is being submitted in partial fulfillment of the requirements for the degree of PhD.

Signed Date

STATEMENT 2

This thesis is the result of my own independent work/investigation, except where otherwise stated. Other sources are acknowledged by explicit references. The views expressed are my own.

Signed Date

STATEMENT 3

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed Date

Acknowledgments

Firstly I would like to thank my excellent supervisor Prof. Tim Phillips for his guidance and support throughout the duration of my studies. In particular I would like to thank him for taking me on as his first non-fluid mechanically inclined PhD student (although I seem to have been gently pushed in that direction in the end!) I would also like to thank him for always being able to find time to see me whilst balancing a number of other PhD students as well as all of his Head of School duties; your time management skills are astounding!

I would also like to gratefully acknowledge the studentship funding I received from the Engineering and Physical Sciences Research Council of the United Kingdom.

During the completion of this work I have had the great pleasure of discussing some of the mathematical ideas presented in this work with a number of esteemed academics from around the world, both in person and via email. Namely, I would like to thank: Amine Ammar, Pavel Bochev, Paco Chinesta, Virginie Ehrlacher, Antonio Falcó, Tony Lelièvre, Rob Owens, Anthony Patera, and Endre Süli. I would also like to thank my two examiners Marc Gerritsma and Karl Michael Schmidt for their very thorough review of this thesis.

I would also like to thank my family for their support and belief in me during the past three and a half years. I would especially like to thank my Grandad, John 'Stokes' Harris, for his impeccable proofreading ability. In particular, I would like to thank him for noticing my incorrect usage of an apostrophe when discussing the Reynolds number!

Special mention and thanks also needs to be given to the Cardiff University non-undergraduate team: Angelico, Brad, Cheryl, Chris, Iza, Jonny G, Julie, Leanne, Lizzie, Mike, Penny, and all the other friends I made in the mathematics department. You guys made this whole process a lot more enjoyable than it probably should be!

Lastly, but by no means least, I would like to thank my lovely girlfriend, and fellow PhD student, Lizzie who I met and fell in love with here in the Cardiff University mathematics department (thanks again Angelico for sorting that out). I would like to thank her for keeping me (mostly) sane during this process and for putting up with me constantly going to her office and distracting her. Sorry!

Contents

Chapter 1

Introduction

The numerical solution of partial differential equations (PDEs) defined in high-dimensional space has, up until recently, been a huge computational challenge. In some cases, when the dimension is sufficiently large, these sorts of problems were completely unsolvable with standard techniques as the required computer memory approaches astronomical orders that are comparable with the estimated total number of atoms in the universe [48]. This issue is commonly referred to by the rather dramatic name of 'the curse of dimensionality'. In order to illustrate the problem, consider a PDE defined in d-dimensional space with solution $u(x_1, \ldots, x_d)$. One would typically look for an approximate numerical solution $u_N(x_1, \ldots, x_d)$ as a linear combination of tensor products:

$$u_N(x_1, \ldots, x_d) = \sum_{i_1=1}^{N} \cdots \sum_{i_d=1}^{N} u_{i_1,\ldots,i_d} \prod_{k=1}^{d} h_{i_k}(x_k), \tag{1.1}$$

where $h_{i_k}(x_k)$ are some basis functions depending on the choice of discretisation. For problems defined in spaces of moderate dimension, d, this is a very effective and commonly used technique. However, when dealing with the numerical approximation of partial differential equations defined in high-dimensional space, if one were to naively try and use this standard tensor product approximation technique it would soon become clear that this is a very impractical task. Indeed, the number of unknowns in (1.1) are the elements of a tensor of order d ($u_{i_1,\ldots,i_d}$) which has N elements in each coordinate direction. This means that the total number of unknowns is N^d and hence the complexity of the problem increases exponentially with increasing dimension d. Clearly this can become very problematic in terms of computational cost when d is sufficiently large.

In this thesis we investigate a method proposed recently by Ammar et al. [8,9] which is now known as the Proper Generalised Decomposition (PGD). This method was designed to alleviate the 'curse of dimensionality' when solving problems defined in high-dimensional spaces. A previously derived method with the same goal in mind is that of sparse grids [34]. However, this latter method has been seen to be restricted to models in moderately high-dimensions ($d \leq 20$). In contrast, the PGD has already been used to solve a $d = 100$ dimensional Poisson equation by Ammar et al. [9]. This increasingly impressive and promising method is based on a separated representation of the unknown field. A separated representation can be thought of as the natural extension of separation of variables and their importance in the field of high-dimensional numerical analysis was first highlighted by Beylkin and Mohlenkamp [18]. Given a function $u(x_1, \dots, x_d)$, a rank-J separated representation of u is given by:

$$u(x_1, \dots, x_d) \approx \sum_{j=1}^{J} F_j^1(x_1) \times \cdots \times F_j^d(x_d). \tag{1.2}$$

The idea behind this approximation is that, as the rank $J \to \infty$, the separated representation approaches the true solution. In contrast with the standard approximation technique (1.1), if we now discretise the basis functions, $F_j^1(x_1), \dots, F_j^d(x_d)$, $j = 1, \dots, J$, in (1.2) then we obtain the following approximate solution:

$$u_N(x_1, \dots, x_d) \approx \sum_{j=1}^{J} \sum_{i_1=1}^{N} \cdots \sum_{i_d=1}^{N} \prod_{k=1}^{d} \alpha_{j,i_k} h_{i_k}(x_k).$$

The unknowns are now the elements of d tensors of order 2 (α_{j,i_k}, $k = 1, \dots, d$) which each have $J \times N$ elements. Therefore the total number of unknowns is simply $J \times N \times d$. This means that the complexity of the problem increases linearly as the dimension, d, is increased instead of the exponential growth found with the standard technique (1.1). This is clearly a great improvement and allows us to work efficiently with problems defined in much higher dimensions. Furthermore, due to the linearity and separability of the separated representation (1.2), we need only use one-dimensional operations. For example, when integrating a high-dimensional function that has this separated form we can apply Fubini's theorem to reduce the problem to a product of one-dimensional integrals rather than having to deal with a higher-dimensional integral.

This separated representation is not unique to the PGD. Indeed, it has been used previously in the large time increment (LATIN) solver of Ladevèze [89] in which his so called 'radial approximations' use a space-time separated representation of the form:

$$u(\mathbf{x}, t) \approx \sum_{j=1}^{J} X_j(\mathbf{x}) \times T_j(t).$$

This method can be thought of as a space-time PGD for $d = 2$. This method allows one to employ non-incremental time integration which can be a huge computational saving. The synergy between the LATIN solver and the PGD was highlighted in [91]. A second example of methods that had previously made use of a separated representation are post-Hartree-Fock methods such as the Configuration Interaction (CI) method [123]. This method is used in computational quantum chemistry and seeks an approximate separated representation of the wave function in Schrödinger's equation. This approximation is equivalent to the commonly used visualisation of electrons occupying orbitals. A final method that uses a separated representation is the Proper Orthogonal Decomposition (POD) (see e.g. Chatterjee [42]). The PGD's name originates as a generalisation of the POD and hence it is particularly relevant. For this reason we now present a more detailed description of the POD before describing the PGD itself.

1.1 The Proper Orthogonal Decomposition

Suppose we wish to approximate a function, f, by the rank-J separated representation

$$f(\mathbf{x},\mathbf{y}) \approx \sum_{j=0}^{J} \alpha_j X_j(\mathbf{x}) Y_j(\mathbf{y}), \tag{1.3}$$

where $\alpha_j \geq 0$, for $j = 1, \dots, J$, and where $\mathbf{x} \in \Omega_x \subset \mathbb{R}^{d_1}$ and $\mathbf{y} \in \Omega_y \subset \mathbb{R}^{d_2}$ are typically of some moderate dimension $d_1, d_2 \leq 3$. We assume that this approximation converges as $J \to \infty$. Note that this representation is not unique since different choices of the functions X_j yield different sets of functions Y_j for $j = 1, \dots, J$, and hence we need to enforce some criteria in order to define the decomposition. The POD enforces an orthonormality condition on the basis functions:

$$\langle X_i, X_j \rangle_x = \langle Y_i, Y_j \rangle_y = \delta_{i,j}, \quad i, j = 1, \dots, J, \tag{1.4}$$

where $\langle \cdot, \cdot \rangle_x$ and $\langle \cdot, \cdot \rangle_y$ denote the L^2-inner products on Ω_x and Ω_y, respectively.

It is then possible to derive an expression for the coefficients α_j, $j = 1, \dots, J$, by considering:

$$\begin{aligned} 0 = \left\| f - \sum_{i=0}^{\infty} \alpha_i X_i Y_i \right\|^2 &= \langle f - \sum_{i=0}^{\infty} \alpha_i X_i Y_i, f - \sum_{j=0}^{\infty} \alpha_j X_j Y_j \rangle \\ &= \sum_{i,j=0}^{\infty} \alpha_i \alpha_j \langle X_i, X_j \rangle_x \langle Y_i, Y_j \rangle_y - 2 \sum_{i=0}^{\infty} \alpha_i \langle f, X_i Y_i \rangle + \langle f, f \rangle, \end{aligned} \tag{1.5}$$

where $\| \cdot \|$ and $\langle \cdot, \cdot \rangle$ denote the L^2-norm and inner product on $\Omega_x \times \Omega_y$. Given that we have:

$$\langle f, f \rangle = \langle \sum_{i=0}^{\infty} \alpha_i X_i Y_i, \sum_{j=0}^{\infty} \alpha_j X_j Y_j \rangle = \sum_{i,j=0}^{\infty} \alpha_i \alpha_j \langle X_i, X_j \rangle_x \langle Y_i, Y_j \rangle_y,$$

then (1.5) can be written as:

$$0 = \ 2\sum_{i,j=0}^{\infty} \alpha_i\alpha_j \underbrace{\langle X_i, X_j\rangle_x}_{=\delta_{i,j}} \underbrace{\langle Y_i, Y_j\rangle_y}_{=\delta_{i,j}} -2\sum_{i=0}^{\infty} \alpha_i\langle f, X_iY_i\rangle = 2\sum_{i=0}^{\infty} \alpha_i^2 - 2\sum_{i=0}^{\infty} \alpha_i\langle f, X_iY_i\rangle.$$

This implies that $\alpha_j = \langle f, X_jY_j\rangle$ for $j = 1, \dots, J$. A similar expression can then be found for the basis functions by considering:

$$0 = \frac{1}{2}\left\| f - \sum_{i=0}^{\infty} \alpha_i X_i Y_i \right\|_x^2 = \sum_{i=0}^{\infty} (\alpha_i Y_i)^2 - \sum_{i=0}^{\infty} \alpha_i Y_i \langle f, X_i\rangle_x.$$

This implies that $\alpha_j Y_j = \langle f, X_j\rangle_x$ which is a desirable result since it means that the basis function Y_j depends only on X_j and not on any of the previous basis functions. Given a particular orthonormal basis $\{X_j\}$ we can then define our decomposition by evaluating the coefficients $\alpha_j \geq 0$ via:

$$\alpha_j^2 = \langle f, \alpha_j X_j Y_j\rangle = \langle f, X_j \langle f, X_j\rangle_x\rangle,$$

for $j = 1, \dots, J$, and then using $Y_j = \frac{1}{\alpha_j}\langle f, X_j\rangle_x$ to evaluate the basis functions Y_j, $j = 1, \dots, J$. In the POD we select the basis functions X_j, $j = 1, \dots, J$, so that the approximation of $f(\mathbf{x}, \mathbf{y})$ for each J is optimal in a least squares sense. The resulting orthonormal basis functions that are obtained are known as the proper orthogonal modes for the function $f(\mathbf{x}, \mathbf{y})$.

The POD is most generally applied in infinite dimensions. However, when considering the numerical approximation of PDEs, we will always discretise the solution and hence we need only apply the POD in finite dimensions. The finite equivalent of the POD can be viewed as the well known Singular Value Decomposition (SVD) (see Trefethen and Bau [128], for example). Indeed, let the discrete analogue of $f(\mathbf{x}, \mathbf{y})$ (which was obtained either by some sort of numerical method or experimental data) be given by the matrix A defined by:

$$A_{i,j} = f(\mathbf{x}_i, \mathbf{y}_j),$$

where $\mathbf{x}_i$ $(i = 1, \dots, n)$ and $\mathbf{y}_j$ $(j = 1, \dots, m)$ are some discrete points inside Ω_x and Ω_y respectively. The SVD of A is then defined by

$$A = U\Sigma V^T$$

where U is an $m \times m$ orthogonal matrix, V is an $n \times n$ orthogonal matrix and Σ is an $m \times n$ matrix with zero entries everywhere except the diagonal. These diagonal entries, $\sigma_i = \Sigma_{i,i}$, are the singular values of A which are all non-negative numbers arranged in decreasing order. So that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$ where

$r = \min(n, m)$. The rank of A is then given by the number of nonzero singular values.

We can then see how this corresponds to a discrete analogue of the POD by letting $\mathbf{u}_k$ and $\mathbf{v}_k$, $k = 1, \ldots, r$, denote the k^{th} columns of U and V, respectively. We write the matrix A as the following matrix product:

$$A = U\Sigma V^T = \sum_{k=1}^{r} \sigma_k \mathbf{u}_k \mathbf{v}_k^T. \tag{1.6}$$

One can think of the vectors $\mathbf{v}_k^T$ and $\mathbf{u}_k$ are discrete analogues of the functions $X_k(\mathbf{x})$ and $Y_k(\mathbf{y})$, respectively, with the singular values being the discrete analogues of the coefficients α_j, $j = 1, \ldots, J$. Therefore (1.6) can be viewed as the discrete (and hence finite) analogue of the full-rank equality:

$$f(\mathbf{x}, \mathbf{y}) = \sum_{j=0}^{\infty} \alpha_j X_j(\mathbf{x}) Y_j(\mathbf{y}), \tag{1.7}$$

where the orthonormality condition (1.4) comes directly from the fact that U and V are orthogonal matrices and hence have orthonormal columns. In this sense, for the continuous POD (1.7), the term full-rank means at most infinite rank whereas in the discrete case full-rank means at most rank-r. It is for this reason that the term 'rank' is used to describe the number of basis functions that appear in a separated representation. Furthermore, it means that we can describe the approximate rank-J approximation of $f(\mathbf{x}, \mathbf{y})$ given in (1.3) as a low-rank (or reduced basis) approximation. Analogously we can obtain a low-rank approximation for the discrete problem (1.6) by setting a number of the smallest singular values to zero. More formally, define the matrix Σ_k for $k < r$ to be the matrix obtained by setting $\sigma_{k+1} = \sigma_{k+2} = \ldots = \sigma_r = 0$ in Σ. We then define our rank-k reduced basis approximation of A to be given by

$$A_k = U\Sigma_k V^T. \tag{1.8}$$

In fact this approximation is the optimal rank-k approximation of A since no other rank-k matrix can be closer to A in the Frobenius norm or 2-norm as stated by the famous Theorem of Eckart and Young [61]. This means that the first k columns of U and V provide an optimal orthonormal basis for approximating A and hence the columns of U and V are the previously mentioned proper orthogonal modes for this decomposition.

The SVD of a matrix can be calculated by recasting the problem as an eigenvalue decomposition. Indeed, consider:

$$A^T A = V\Sigma^T U^T U\Sigma V^T.$$

Now since U is orthogonal we have that $U^T U = I$. Hence this becomes

$$A^T A = V\Sigma^T \Sigma V^T = V\Lambda V^{-1},$$

where $\Lambda = \Sigma^T \Sigma$ is an $n \times n$ diagonal matrix, the diagonal entries of which are the squares of the singular values of A. This is of the form of an eigenvalue decomposition for the matrix $A^T A$ where the eigenvalues of $A^T A$ are the squares of the singular values of A and the eigenvectors of $A^T A$ are the columns of V. Therefore the singular values and proper orthogonal modes can be found by solving the following eigenvalue problem:

$$A^T A\mathbf{v} = \sigma^2 \mathbf{v}.$$

Note that the columns of U can be found similarly by considering the matrix AA^T. A method for finding a reduced basis approximation using this eigenvalue decomposition was described in a paper by Chinesta et al. [50]. In this paper the authors demonstrated the effectiveness of such a reduced basis approximation by considering the example of the one-dimensional heat transfer problem discretized using a finite element method on a mesh with $n = 100$ nodes and $m = 300$ time steps. The rank of the associated full model is then less than $r = \min(n, m) = 100$ (but presumably still large). Using their chosen restriction on the choice of eigenvectors to use in the reduced basis approximation ($\geq 10^{-8}\sigma_1^2$) they then obtained a rank-3 reduced basis approximation to the problem which yielded a very accurate representation of the solution. This example highlights the power of reduced basis approximations via separated representations. Indeed, the number of degrees of freedom in a full-rank tensor product approximation of the problem would be 30,000 whereas in the rank-3 reduced basis approximation there are instead only 1,200 degrees of freedom.

While this may sound impressive the problem with the POD (and SVD) is that the full-rank problem needs to be solved initially before we can 'discard' a certain number of the proper orthogonal modes in order to obtain a low-rank approximation. In this sense there is no significant computational saving in applying the method outlined above. In other words the big disadvantage of the POD is that a priori knowledge of the solution is needed in order to obtain a low-rank approximation.

Of course it is not the case that the POD is completely useless as a reduced basis technique. One particular example is the use of snapshotting in which snapshots in time (or for different values of parameters) are solved to obtain full-rank approximations. Approximate low-rank approximations are then constructed for intermittent times (or parameter values) from the proper orthogonal modes of the snapshots by projection onto the reduced basis. These types of methods have been used extensively in fluid mechanics computations (see e.g. [35, 124]). Although it is also the case that the PGD can also be applied to these problems by including time (or the parameters) as additional independent variables. We will elaborate on this later.

A second disadvantage of the POD is that it is only defined as a separated representation in two variables (i.e. it does not make use of the more general, high dimensional separated representation (1.2)). In a paper by Kolda [85] higher dimensional tensor decompositions analogous to the POD were studied and a high dimensional generalisation of the Theorem of Eckart and Young [61] was conjectured. Unfortunately, de Silva and Lim [56] proved that tensors of order ≥ 3 can fail to have a best rank-r approximation for $r \geq 2$ and so, in general, no such analogous result exists. Therefore, while higher dimensional versions of the POD exist (see e.g. Kolda and Bader [86]), we do not have guaranteed optimality as in the standard POD. Furthermore, for very high dimensional problems this does not alleviate the 'curse of dimensionality' since full-rank approximations still need to be computationally viable to solve.

1.2 The Proper Generalised Decomposition

From the previous section we found that there were two disadvantages with the POD:

1. A priori knowledge of the solution is needed before a reduced basis approximation can be applied.
2. The standard definition of the POD is only applicable for a separated representation in two variables.

The PGD addresses both of these issues by obtaining an approximate d-dimensional separated representation of the form:

$$u(x_1, \ldots, x_d) \approx \sum_{j=1}^{J} F_j^1(x_1) \times \cdots \times F_j^d(x_d), \tag{1.9}$$

where x_i, $i = 1, \ldots, d$ are of some moderate dimensions (≤ 3). Furthermore, this is constructed without any a priori knowledge of the solution. There are a number of ways this can be done since the PGD is actually a family of methods. A number of different PGD algorithms are described by Nouy [106]. Throughout this thesis we will concentrate on the simplest definition of the PGD: the progressive PGD. This version of the PGD seeks to find iteratively the 'best' rank-one separated representation (or rank-one tensor) $F_j^1(x_1) \times \cdots \times F_j^d(x_d)$ for each $j = 1, \ldots, J$. The reason we only seek a rank-one tensor at each iteration is once again due to the result of de Silva and Lim [56] in which they proved tensors of order ≥ 3 can fail to have a best rank-r approximation for $r \geq 2$. The basis functions $F_j^i(x_i)$,

$i = 1, \ldots, d$, $j = 1, \ldots, J$ are known as the PGD modes in analogy with the POD. Previously calculated PGD modes are simply moved to the right hand side of the PDE and the next rank-one tensor is sought. Unlike in the POD, the PGD modes, $\{F_j^1(x_1), \ldots, F_j^d(x_d) | j = 1, \ldots, J\}$, will not, in general, be orthonormal. Orthonormality was imposed in the POD in order to define a unique decomposition but this is not needed in the progressive PGD due to the iterative nature of the algorithm.

In Chapter 2, when we introduce Galerkin PGDs, we will specify what we mean by the 'best' rank-one tensor but we will point out that it does not necessarily mean the optimal choice. If we considered a separated representation in just two variables ($d = 2$), then choosing the optimal rank-one tensor at each iteration would be equivalent to constructing the POD dynamically with no a priori knowledge of the solution (due to the Theorem of Eckart and Young [61]). This was attempted by Leonenko and Phillips [96] by orthonormalising the PGD modes, $F_j^1(x_1)$ and $F_j^2(x_2)$, using a Gram-Schmidt procedure to obtain a new set of basis functions, $\hat{F}_j^1(x_1)$ and $\hat{F}_j^2(x_2)$, say, for each $j = 1, \ldots, J$. The solution was then projected on to this new basis to obtain an approximate separated representation of the solution of the form:

$$u(x_1, x_d) \approx \sum_{j=1}^{J} \alpha_j \hat{F}_j^1(x_1) \hat{F}_j^2(x_2), \tag{1.10}$$

where the coefficients α_j are calculated from the projection onto the new basis. This clearly resembles the POD where the coefficients α_j resemble the singular values. Unfortunately, this separated representation will not, in general, be optimal. Indeed, it is not known if it is possible to dynamically construct the POD a priori in this manner. However, Nouy [106] presented an optimal Galerkin PGD, which is optimal in the sense of the Galerkin projection, which displayed convergence rates that were very close to those of the POD. Unfortunately, this PGD algorithm is generally very expensive to implement and hence is not practical. A progressive PGD using the projective step as in (1.10) was also employed in the first publications on the PGD by Ammar et al. [8, 9] (although without the orthonormalisation of the basis functions). This projective step can sometimes provide better rates of convergence than a standard progressive PGD.

Other types of PGD algorithm introduced by Nouy [106] include the minimal residual PGD and the minimax PGD. The minimal residual PGD is closely related to the least-squares PGD which we will discuss further in Chapter 3. Very recently a PGD type algorithm called the ideal minimal residual was proposed by Billaud-Friess et al. [20] which displays impressive rates of convergence, close to the optimal POD when $d = 2$: much like the optimal Galerkin PGD for Galerkin type problems. The minimax PGD can be thought of as a PGD based on a Petrov-Galerkin formulation and has also displayed impressive rates of convergence [106].

In-depth details of how the progressive PGD is employed will be saved for Chapter 2. We now present a literature review of the theoretical developments of the PGD as well as a number of interesting applications.

1.3 PGD Literature Review

PGDs are a family of methods which are very much in their infancy. Indeed, the first paper on the PGD by Ammar et al. was published in 2006 (not including the earlier work on the LATIN method by Ladevèze [89]) and the first books on the topic have only very recently been released in 2013/14 by Chinesta et al. [51, 52]. As a result PGDs are not yet well understood. However, there has been a great deal of progress in the advancement of theoretical understanding of PGD algorithms which we shall summarise here.

One of the natural first questions one might have as a numerical analyst is whether or not PGDs converge. This question was first addressed by Le Bris et al. [95] in which progressive PGDs were compared with greedy algorithms, the likes of which had previously been studied by Temylakov [127]. The authors of [95] then went on to prove convergence of a greedy algorithm for the solution of the Poisson equation. This work was further extended to the more general case of elliptic nonlinear self-adjoint problems in tensor product Hilbert spaces by Cancès et al. [38] and to tensor product Banach spaces (as well as to a larger class of PGD algorithms) by Falcó and Nouy [67]. In an earlier work, Falcó and Nouy [66] also provided a novel proof of convergence for linear elliptic variational problems which draws an analogy to the classic Theorem of Eckart and Young [61]. Other convergence results for greedy algorithms include convergence for a class of linear systems by Ammar et al. [6], and convergence of a greedy algorithm for the Maxwellian transformed Fokker-Planck equation by Figueroa and Süli [69].

A number of other theoretic advancements have also been made including the paper of Nouy [106] in which he expressed PGD algorithms as pseudo-eigenproblems which led to some of the different definitions of PGDs that were mentioned in the previous section. There have also been improvements to numerical strategies for faster convergence of Galerkin PGDs [65] and minimal residual PGDs [20]. PGDs defined in more complex geometries and ways of imposing non-homogeneous Dirichlet boundary conditions were proposed by González et al. [71]. Finally, Ammar et al. [5] and Ladevèze and Chamoin [90] developed error estimators for PGD algorithms which can be used as stopping criteria or used in adaptive strategies.

Since the PGD's conception it has been applied to a large variety of applications. An excellent review of these, up to 2011, was compiled by Chinesta et al. [53]. The first application of the PGD was to the potentially high-dimensional Fokker-Planck equation governing the evolution of kinetic theory models in polymer rheology in the seminal papers of Ammar et al. [8, 9]. There have also been a number of other papers on this topic which can be found in the exhaustive review of this application by Chinesta et al. [50]. We will also be considering this application of the PGD in Chapter 4 of this thesis in which a more in depth review of the application can be found. Other multidimensional models that the PGD has been applied to include Schrödinger's equation in quantum chemistry [49], financial models for option pricing with a high number of risk factors [64], and the chemical master equation in [46] which has led to some very interesting applications in systems biology (e.g. see Chancellor et al. [40]).

The PGDs applicability is not just limited to multidimensional models. It can also be used to efficiently obtain solutions to moderate dimensional problems. For example, it has been applied to the 3D Navier-Stokes equations defined in a cube by Dumon et al. [60]. The PGD is also very effective for problems defined on plate-like geometries (see Bognet et al. [26]) in which one seeks a separated representation of the form:

$$u(x, y, z) \approx \sum_{j=1}^{J} X_j(x, y) Z_j(z),$$

where the plate surface is defined on the (x, y)-plane and may have complex geometry and the z coordinate is defined in an interval associated with the thickness of the plate. This proved to be a very successful technique in [26].

The PGD has also proved to be successful when applied to stochastic PDEs: a concept which was first proposed by Nouy [105] under the name 'generalized spectral decomposition' in which the stochastic variables are separated from the deterministic variables in the separated representation. One particularly interesting example of this is the solution of the stochastic steady Navier-Stokes equations by Tamellini et al. [125] in which the Reynolds number and forcing term are considered as random variables.

Multiscale problems are also well suited to be solved via a PGD algorithm. The reason for this being that the different scales can be accounted for by including extra independent variables. This increases the dimensionality of the problem which can be efficiently treated using the PGD. This has been applied to problems with local kinetic couplings by Chinesta et al. [47] and for problems involving different time scales by Ammar et al. [4].

Finally, one of the most popular and impressive applications of the PGD in the engineering sciences is that of parametric models. The main idea being that, given a problem defined in a space of moderate dimension with parameters $\lambda_1, \ldots, \lambda_d$, then we can seek a separated representation of the solution of the form:

$$u(\mathbf{x}; \lambda_1, \ldots, \lambda_d) \approx \sum_{j=1}^{J} X_j(\mathbf{x}) \Lambda_j^1(\lambda_1) \times \cdots \times \Lambda_j^d(\lambda_d).$$

In other words, the parameters are included as additional variables which increases the dimensionality of the problem. However, this can be handled efficiently by the PGD. Furthermore, the parameter space, $(\lambda_1, \ldots, \lambda_d) \in \Lambda$, is a d-orthotope (a d-dimensional hyper-rectangle) the likes of which PGDs are naturally applicable to. It is also possible to include boundary conditions or initial conditions as extra variables. The application of the PGD to parametric models of this type was considered by Pruliere et al. [114]. This can be used to obtain close to real-time simulation of complex problems whereby the PGD is solved in an offline stage to obtain the solution for all possible values of parameters in a given range providing what Chinesta et al. [54] refer to as a 'computational vademecum' (Latin for 'handbook') which can be used to look up solutions for certain parameters in the online stage in close to real time. One of the most impressive and pioneering applications of this is to the real-time simulation of surgery (see Niroomandi et al. [104]) which can be used to train surgeons without having to use live subjects. A more general study of this to the haptic collision of nonlinear solids (of which simulating surgery is an example) was recently conducted by González et al. [70]. Another application of the PGD for parametric models is that of shape optimisation (e.g. see Ammar et al. [7]) in which geometric parameters are included as additional variables.

Clearly there is a wide variety of potential applications for PGD algorithms as their ability to efficiently handle high-dimensional problems opens up whole new realms of possibility in the world of computational engineering and mathematics. This in turn means that there is a need for further numerical analysis in order to fully understand this increasingly employed method. In the next section we detail what this thesis hopes to contribute to the understanding and development of the PGD.

1.4 Objectives and Outline of Thesis

The main objectives of this thesis are:

- To review the progressive Galerkin PGD, in particular when we can and cannot prove convergence of greedy algorithms associated with these PGDs, and to further develop understanding and techniques for these types of algorithm.

- To develop progressive PGD algorithms based on rigorously defined least-squares methods providing a variety of results and proving convergence of associated greedy algorithms.

- To develop a method to apply a PGD algorithm to the solution of the fully non-homogeneous Fokker-Planck equation in polymer rheology and to couple this with macroscopic flow problems.

The thesis is structured as follows: In Chapter 2 we consider the Galerkin progressive PGD applying it to the Poisson equation and Stokes equations and reviewing two different proofs of convergence of associated greedy algorithms. We also make some comments on the application of a spectral element discretisation in the PGD which has not yet been considered in detail. In Chapter 3 we introduce the least-squares PGD and compare a variety of formulations for the Poisson, convection-diffusion and Stokes equation. We show that greedy algorithms associated with least-squares PGDs converge for all elliptic problems. In Chapter 4 we consider an application of the PGD to kinetic theory models in polymer rheology. This involves applying the PGD to the solution of the Fokker-Planck equation in purely configurational space as well as the more practical application of the PGD to the fully non-homogeneous Fokker-Planck equation in both configurational and physical space. This can then be coupled to the macroscopic Navier-Stokes equations to model the non-Newtonian flow of dilute polymers, for example. Finally, in Chapter 5 we provide conclusions on the whole thesis and suggest some areas of potential future interest.

Chapter 2

Galerkin Proper Generalised Decompositions

2.1 Introduction

In this chapter we consider the progressive Galerkin PGD. Galerkin PGDs are currently the most employed type of PGD algorithm and, in particular, were used in the first papers on the topic by Ammar et al. [8, 9]. In Section 1.2 we briefly described how a progressive PGD works: We iteratively find the 'best' rank-one tensor, where all the previously calculated tensors are included on the right-hand side of the equation. In Galerkin PGD algorithms what we mean by the 'best' rank-one tensor is the one which satisfies Galerkin orthogonality. More formally, assume we have the following problem in weak form: Find $u \in V(\Omega)$ such that:

$$a(u, u^*) = L(u^*), \qquad \forall u^* \in V(\Omega), \tag{2.1}$$

where $a(\cdot, \cdot)$ and $L(\cdot)$ are some bilinear and linear forms respectively with some suitable function space $V(\Omega)$ with homogeneous Dirichlet boundary conditions and where $\Omega \subset \mathbb{R}^d$. Further assume we are at the stage where we have already calculated the following rank-J approximate separated representation, u_J, of u:

$$u_J(x_1, \ldots, x_d) = \sum_{j=1}^{J} \prod_{i=1}^{d} F_j^i(x_i). \tag{2.2}$$

We then enrich our basis by including an additional rank-one tensor in the following way:

$$u_J^{(e)}(x_1, \ldots, x_d) = u_J(x_1, \ldots, x_d) + \prod_{i=1}^{d} r_i(x_i),$$

where the modes $r_i(x_i)$, $i = 1, \ldots, d$, are a priori unknown. These new modes are calculated by requiring that the enriched approximation satisfies Galerkin orthogonality. That is to say, given that from (2.1) we have:

$$a(u_J^{(e)}, u^*) = L(u^*) + a(\epsilon_J, u^*), \qquad \forall u^* \in V(\Omega), \tag{2.3}$$

where $\epsilon_J := u_J^{(e)} - u$ is the residual, Galerkin orthogonality requires that this residual satisfies $a(\epsilon_J, u^*) = 0$, $\forall u^* \in V^J(\Omega)$, where $V^J(\Omega) \subset V(\Omega)$ is a suitable approximation space associated with the low rank PGD approximation. Hence, from (2.3), we can see that the modes $r_i(x_i)$, $i = 1, \ldots, d$, are required to satisfy the following equation:

$$a\left(\prod_{i=1}^{d} r_i(x_i), u^*\right) = L(u^*) - a(u_J, u^*), \qquad \forall u^* \in V^J(\Omega), \tag{2.4}$$

where the known part of the enriched solution, u_J, has been moved to the RHS. The test function for this progressive PGD is chosen to be of the form:

$$u^*(x_1, \ldots, x_d) = \sum_{k=1}^{d} r_k^*(x_k) \prod_{\substack{i=1 \\ i \neq k}}^{d} r_i(x_i),$$

where $r_k^*(x_k)$ is a suitable test function in the x_k coordinate direction for $k = 1, \ldots, d$. This is the most natural choice of test function coming from the calculus of variations (see Ammar et al. [8]). The modes we obtain as the solution of (2.4) with this chosen test function are then selected as the next set of PGD modes:

$$F_{J+1}^i(x_i) = r_i(x_i), \qquad \text{for} \qquad i = 1, \ldots, d.$$

The progressive Galerkin PGD continues in this way until a desired rank approximation is reached or some stopping criterion is satisfied. Note that the equation (2.4) is nonlinear in the modes $r_i(x_i)$, $i = 1, \ldots, d$, and hence we need to employ a linearisation in order to solve it. We will use an alternating directions fixed point algorithm to linearise (2.4) which, while quite simple, has proven to be very efficient (e.g. see Chinesta et al. for detailed examples when $d = 2$ [48] and $d = 3$ [50]). We will give a more detailed description of this linearisation, as well as the other elements of the progressive Galerkin PGD, in the next section when we consider the example of the Poisson equation in 2D.

2.2 The Poisson Equation

Consider the following 2D Poisson equation with homogeneous boundary conditions:

$$\begin{aligned} -\nabla^2 u = f \quad &\text{in} \quad \Omega \subset \mathbb{R}^2, \\ u = 0 \quad &\text{on} \quad \partial\Omega, \end{aligned}$$

which can be reformulated as the following weak problem. Find $u \in H_0^1(\Omega)$ such that

$$a(u,v) = L(v), \qquad \forall v \in H_0^1(\Omega) \tag{2.5}$$

where

$$a(u,v) = \int_\Omega \nabla u \cdot \nabla v \; d\Omega, \qquad L(v) = \int_\Omega fv \; d\Omega. \tag{2.6}$$

The PGD is most commonly coupled with a finite element method (see e.g. Ammar et al. [3]). However, throughout this thesis we will instead employ a spectral element discretisation. This has been used with the PGD by Leonenko and Phillips [96] and our aim is to further investigate how this higher order discretisation performs in the PGD context.

2.2.1 Spectral Element Discretisation

Spectral elements methods were first introduced by Patera [110] and they combine the flexibility of linear finite element methods with the exponential convergence of spectral methods (see e.g. Canuto et al. [39]). They are closely related to the more recent higher-order finite element methods known as hp-FEM introduced by Babuška and Guo [13]. Spectral methods use a basis which is built up from orthogonal polynomials, for example, Legendre or Chebyshev polynomials. In this thesis we focus on Legendre spectral element methods and we shall now describe how these are incorporated into the PGD.

For simplicity consider the rectangular domain $\Omega = [a,b] \times [c,d]$. This domain is split into K rectangular elements by dividing the x-domain, $[a,b]$, into K_x elements, $[a_{k-1}, a_k]$, $k = 1, \ldots, K_x$, and the y-domain, $[c,d]$, into K_y elements $[c_{k-1}, c_k]$, $k = 1, \ldots, K_y$.

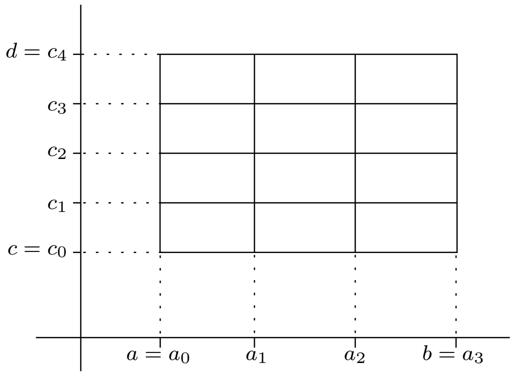


Figure 2.1: PGD Spectral Element Mesh for $K_x = 3$, $K_y = 4$

The K rectangular elements are then effectively provided by the application of these one dimensional meshes to the PGD basis functions in x and y respectively so that $K = K_x \times K_y$ (e.g. see Fig. 2.1). We wish to approximate the solution u of the weak problem (2.5) using the reduced basis separated form

$$u(x,y) \approx \sum_{j=1}^{J} X_j(x)Y_j(y) =: u_J(x,y) \tag{2.7}$$

where $X_j(x)$ and $Y_j(y)$ are piecewise polynomial basis functions given by:

$$X_j(x) = \begin{cases} \sum_{i=0}^{N} \alpha_{j,i,k} h_{i,k}(x), & x \in [a_{k-1}, a_k], \\ 0, & \text{otherwise}, \end{cases} \tag{2.8}$$

$$Y_j(y) = \begin{cases} \sum_{i=0}^{N} \beta_{j,i,k} h_{i,k}(y), & y \in [c_{k-1}, c_k] \\ 0, & \text{otherwise}. \end{cases} \tag{2.9}$$

The homogeneous Dirichlet boundary condition is then included explicitly by requiring that $\alpha_{j,0,1} = \alpha_{j,N,K_x} = \beta_{j,0,1} = \beta_{j,N,K_y} = 0$, for $j = 1, \ldots, J$.

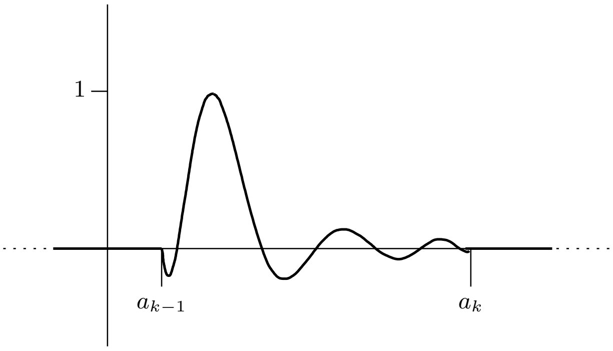


Figure 2.2: The Interpolating Polynomial $h_{3,k}(x)$ for $N = 8$

The interpolating polynomials $h_{i,k}(x)$ $(i = 0, \ldots, N,\ k = 1, \ldots, K_x)$ are defined to be the standard Legendre interpolating polynomials mapped to the element $[a_{k-1}, a_k]$ and zero outside this element. A particular example of one of these interpolating polynomials is shown in Figure 2.2. More formally we have:

$$h_{i,k}(x) = \begin{cases} h_i(\xi_k(x)) := \dfrac{(1-(\xi_k(x))^2)P_N'(\xi_k(x))}{N(N+1)P_N(x_i)(x_i - \xi_k(x))}, & x \in [a_{k-1}, a_k], \\ \\ 0, & \text{otherwise}, \end{cases}$$

where $P_N(x)$ denotes the Legendre polynomial of degree N, x_i $(i = 0, \ldots, N)$ are the Gauss-Lobatto-Legendre (GLL) points and $\xi_k(x)$ $(k = 1, \ldots, K_x)$ are the mappings given by:

$$\xi_k(x) = -1 + \frac{2}{\Delta a_k}(x - a_{k-1}), \tag{2.10}$$

where Δa_k is the size of the k^{th} element on the x-domain, $\Delta a_k = a_k - a_{k-1}$. Note that $h_{i,k}(y)$ is defined analogously with mappings $\eta_k(y)$ $(k = 1, \ldots, K_y)$ given by:

$$\eta_k(y) = -1 + \frac{2}{\Delta c_k}(y - c_{k-1}). \tag{2.11}$$

In the implementation of the spectral element discretisation we need to map several integrals over the K elements to the parent square $[-1, 1]^2$ in order to make use of GLL quadrature. For this reason we also need to define the inverse mappings $\xi_k^{-1}(x)$ $(k = 1, \ldots, K_x)$ and $\eta_k^{-1}(y)$ $(k = 1, \ldots, K_y)$:

$$\xi_k^{-1}(x) = a_{k-1} + \frac{\Delta a_k}{2}(x + 1), \quad \eta_k^{-1}(y) = c_{k-1} + \frac{\Delta c_k}{2}(y + 1),$$

as well as the GLL weights w_i $(i = 0, \ldots, N)$:

$$w_i = \frac{2}{N(N+1)(P_N(x_i))^2},$$

and also the Legendre collocation differentiation matrix, D, since it will appear in the resulting linear systems:

$$D_{i,j} = h_j'(x_i) = \begin{cases} \dfrac{1}{(x_i - x_j)} \cdot \dfrac{P_N(x_i)}{P_N(x_j)}, & i \neq j, \\ 0, & i = j, \ 1 \leq i \leq N-1, \\ \dfrac{-N(N+1)}{4}, & i = j = 0, \\ \dfrac{N(N+1)}{4}, & i = j = N. \end{cases}$$

2.2.2 The Progressive PGD Algorithm

A description of the progressive Galerkin PGD algorithm was given in Section 2.1. Indeed, if we assume we are at the $(J+1)^{\text{th}}$ iteration then the enriched approximate solution takes the form:

$$u_J^{(e)}(x, y) = u_J(x, y) + r(x)s(y) = \sum_{j=1}^{J} X_j(x)Y_j(y) + r(x)s(y),$$

or in the case of $J = 0$, i.e. at the beginning of the algorithm, we have just

$$u_0^{(e)}(x, y) = r(x)s(y).$$

We can then define the progressive Galerkin PGD algorithm by the following iterative procedure:

$$u_j(x, y) = u_{j-1}^{(e)}(x, y), \quad u_0(x, y) = 0. \tag{2.12}$$

This continues until either a desired rank, $j = J$, is reached or when some global convergence criterion is satisfied.

Firstly, as with the basis functions $X_j(x)Y_j(y)$, the modes of the enrichment couple, $r(x)s(y)$, are discretised using a Legendre spectral element method:

$$r(x) = \begin{cases} \sum_{i=0}^{N} r_{i,k} h_{i,k}(x), & x \in [a_{k-1}, a_k], \\ 0, & \text{otherwise}, \end{cases}$$

$$s(y) = \begin{cases} \sum_{i=0}^{N} s_{i,k} h_{i,k}(y), & y \in [c_{k-1}, c_k] \\ 0, & \text{otherwise}, \end{cases}$$

where the homogeneous boundary conditions are included explicitly by requiring that $r_{0,1} = r_{N,K_x} = s_{0,1} = s_{N,K_y} = 0$. We then require that our enriched solution satisfies Galerkin orthogonality and hence we seek modes $r(x)$ and $s(y)$ which satisfy:

$$a(r(x)s(y), u^*) = L(u^*) - a(u_J, u^*), \tag{2.13}$$

where $a(\cdot, \cdot)$ and $L(\cdot)$ are given by (2.6) and where the test functions are given by:

$$u^*(x, y) = r(x)h_{l,k_y}(y) + s(y)h_{m,k_x}(x), \tag{2.14}$$

for $l, m = 0, \ldots, N$, $k_x = 1, \ldots, K_x$, and $k_y = 1, \ldots, K_y$.

As stated in Section 2.1, (2.13) is nonlinear in $r(x)$ and $s(y)$. This is linearised using an alternating directions fixed point algorithm (ADFPA) which we describe in the next section.

2.2.3 Alternating Directions Fixed Point Algorithm

The main idea of the ADFPA is to treat each coordinate direction separately, iteratively updating the modes in each coordinate direction by solving a series of linear systems. The algorithm begins by making an initial guess for one of the modes, $r(x)$, say, denoted by $r^{(0)}(x)$. This reduces the test function (2.14) to:

$$u_y^{*(0)}(x, y) = r^{(0)}(x)h_{l,k_y}(y), \quad l = 0, \ldots, N, \quad k_y = 1, \ldots, K_y.$$

We then solve the following problem, in the y variable only, to obtain an initial approximation $s^{(0)}(y)$ to $s(y)$:

$$a(r^{(0)}(x)s^{(0)}(y), u_y^{*(0)}) = L(u_y^{*(0)}) - a(u_J, u_y^{*(0)}).$$

We then use this calculated value of $s^{(0)}(y)$ to obtain an updated approximate value of $r(x)$, given by $r^{(1)}(x)$, by solving the following problem in the x variable only:

$$a(r^{(1)}(x)s^{(0)}(y), u_x^{*(0)}) = L(u_x^{*(0)}) - a(u_J, u_x^{*(0)}),$$

where the test function, $u_x^{*(0)}$, is defined by:

$$u_x^{*(0)}(x,y) = h_{m,k_x}(x)s^{(0)}(y), \quad m = 0,\dots,N, \quad k_x = 1,\dots,K_x.$$

The algorithm continues in this way until some convergence criterion is met with some chosen tolerance ϵ. This leads to the following definition of the ADFPA for this problem:

Algorithm 1 Alternating Directions Fixed Point Algorithm

```
**Input:** $(r,\ \epsilon)$
  $r^{(0)} = r$
  $s^{(0)} = A_y(r^{(0)})$
  $r^{(1)} = A_x(s^{(0)})$
  $s^{(1)} = A_y(r^{(1)})$
  $n = 1$
  **while** $\left\|r^{(n)}s^{(n)} - r^{(n-1)}s^{(n-1)}\right\| \geq \epsilon$ **do**
    $n \leftarrow n + 1$
    $r^{(n)} = A_x(s^{(n-1)})$
    $s^{(n)} = A_y(r^{(n)})$
  **end while**
```

where the procedures $A_x(s^{(n)})$ and $A_y(r^{(n)})$ denote the solution of the problems

$$a(r^{(n+1)}(x)s^{(n)}(y), u_x^{*(n)}) = L(u_x^{*(n)}) - a(u_J, u_x^{*(n)}), \tag{2.15}$$

and

$$a(r^{(n)}(x)s^{(n)}(y), u_y^{*(n)}) = L(u_y^{*(n)}) - a(u_J, u_y^{*(n)}), \tag{2.16}$$

respectively, with test functions defined by:

$$u_x^{*(n)}(x,y) = h_{m,k_x}(x)s^{(n)}(y), \quad m = 0,\dots,N, \quad k_x = 1,\dots,K_x,$$

and

$$u_y^{*(n)}(x,y) = r^{(n)}(x)h_{l,k_y}(y), \quad l = 0,\dots,N, \quad k_y = 1,\dots,K_y.$$

Once Algorithm 1 has converged then we can take the final values of $r^{(n)}(x)$ and $s^{(n)}(y)$ to be our approximate values of the solutions, $r(x)$ and $s(y)$, of the nonlinear problem (2.13) which in turn are then chosen to be the next two PGD modes $X_{J+1}(x) = r^{(n)}(x)$ and $Y_{J+1}(y) = s^{(n)}(y)$.

We now present the discrete linear systems which arise from this linearisation. For brevity we only consider the y-direction solve (2.16). In discrete form this leads to

K_y local linear systems:

$$A^{k_y}\mathbf{s}^{k_y} = \mathbf{b}^{k_y}, \qquad k_y = 1, \ldots, K_y,$$

where, for simplicity, we have dropped the superscript indicating the current iteration of the ADFPA and where $\mathbf{s}^{k_y}$ is the vector of unknowns s_{i,k_y}, $i = 1, \ldots, N$. The elements of the local matrix, A^{k_y}, and the RHS, $\mathbf{b}^{k_y}$, are found by mapping each of the $K = K_x \times K_y$ elements to the parent square $[-1, 1]^2$ using the mappings (2.10) and (2.11). This enables us to apply GLL quadrature in order to evaluate the integrals in (2.16) for each of the K_y local linear systems. For our specific spectral element discretisation this leads to the following expression for the elements of A^{k_y}:

$$A^{k_y}_{l,m} = \sum_{k_x=1}^{K_x} \left(\frac{\Delta a_{k_x}}{\Delta a_{k_y}} \sum_{i=0}^{N} w_i r^2_{i,k_x} \sum_{n=0}^{N} w_n D_{n,m} D_{n,l} + w_l \delta_{l,m} \frac{\Delta a_{k_y}}{\Delta a_{k_x}} \sum_{i=0}^{N} w_i \left(\sum_{n=0}^{N} r_{n,k_x} D_{i,n} \right)^2 \right),$$

and for the elements of $\mathbf{b}^{k_y}$:

$$\begin{aligned} b^{k_y}_l = & -\sum_{k_x=1}^{K_x} \sum_{j=1}^{J} \left[\frac{\Delta a_{k_x}}{\Delta a_{k_y}} \beta_{j,l,k_y} w_l \left(\sum_{i=0}^{N} w_i \sum_{n=0}^{N} \alpha_{j,n,k_x} D_{i,n} \sum_{m=0}^{N} r_{m,k_x} D_{i,m} \right) \right. \\ & \left. + \frac{\Delta a_{k_y}}{\Delta a_{k_x}} \sum_{i=0}^{N} w_i \alpha_{j,i,k_x} r_{i,k_x} \sum_{n=0}^{N} \beta_{j,n,k_y} \sum_{m=0}^{N} w_m D_{n,m} D_{n,l} \right] \\ & + \frac{1}{4} \Delta a_{k_y} w_l \sum_{k_x=1}^{K_x} \Delta a_{k_x} \sum_{i=0}^{N} w_i r_{i,k_x} f(\xi^{-1}_{k_x}(x_i), \eta^{-1}_{k_y}(y_l)). \end{aligned}$$

Note that, due to Fubini's Theorem, we only need to calculate the product of one dimensional integrals for all terms apart from the source term $f(x, y)$ which may not, in general, have a separated representation. For this moderate dimensional problem this is not an issue since a 2D quadrature rule for this term is still very cheap to employ. However, if one considered a high-dimensional problem then the integral of a general d-dimensional source term could be very expensive. For this reason one could first employ a low-rank, high-dimensional version of the POD (see e.g. Kolda and Bader [86]) to approximate the full-rank source term in order that Fubini's Theorem can be applied to all terms.

We can then use these K_y local systems to construct the global system (see Figure 2.3) where we ensure continuity of $s(y)$ by equating contributions at element boundaries (i.e. $s_{N,k} = s_{0,k+1}$ for $k = 1, \ldots, K_y - 1$). Before solving this global system we firstly remove the first and last rows and the first and last columns from the system since we have explicitly defined Dirichlet boundary conditions at both end points of the domain.

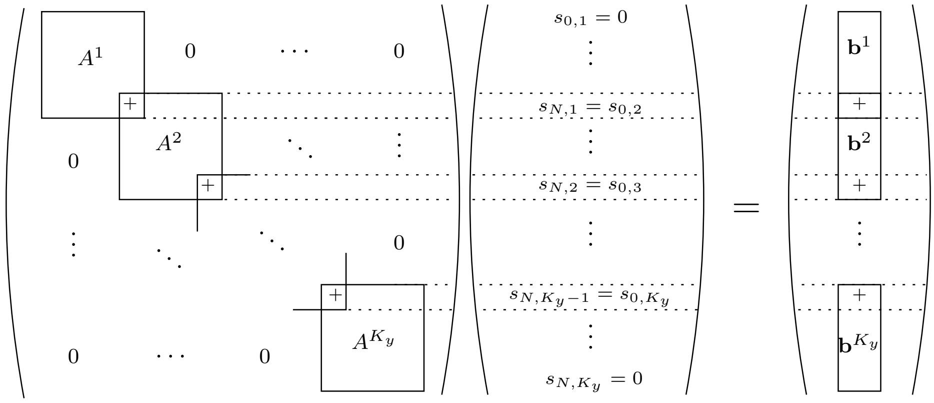


Figure 2.3: Construction of the Global System

2.2.4 Numerical Results

Example 1 (Infinite Rank Solution)**.**

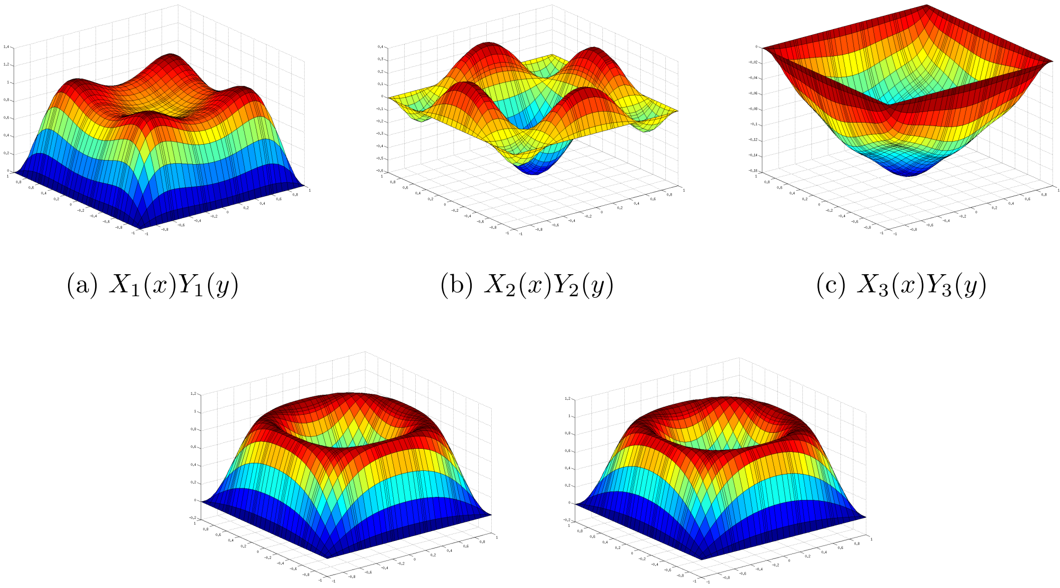
The first example we consider is the following:

$$\begin{aligned} -\nabla^2 u = f(x,y) \quad &\text{in} \quad \Omega = [-1,1]^2 \\ u = 0 \quad &\text{on} \quad \partial\Omega, \end{aligned}$$

with source term:

$$\begin{aligned} f(x,y) =& 4\pi^2(x^2(1-y^2)^2 + y^2(1-x^2)^2)\sin(\pi(1-x^2)(1-y^2)) \\ &+ 2\pi((1-x^2)+(1-y^2))\cos(\pi(1-x^2)(1-y^2)), \end{aligned}$$

which has known infinite-rank solution $u(x,y) = \sin(\pi(1-x^2)(1-y^2))$.



(a) $X_1(x)Y_1(y)$ (b) $X_2(x)Y_2(y)$ (c) $X_3(x)Y_3(y)$

(d) $u_3(x,y)$ (e) True Solution

Figure 2.4: Rank-3 PGD Modes and PGD Approximation for Example 1

Figure 2.4 shows the results for a rank-3 PGD approximation of the solution to the given Poisson equation. Figures 2.4(a)-(c) show the first three pairs of PGD modes and (d) shows the rank-3 PGD approximation which is the sum of the functions (a)-(c):

$$u_3(x,y) = \sum_{j=1}^{3} X_j(x)Y_j(y).$$

This is compared to a plot of the known true solution, $u(x,y) = \sin(\pi(1-x^2)(1-y^2))$, in Figure 2.4(e), which is practically indistinguishable from the PGD approximation. The spectral element discretisation used for these results used degree $N = 8$ polynomials on $K_x = K_y = 5$ elements in each coordinate direction.

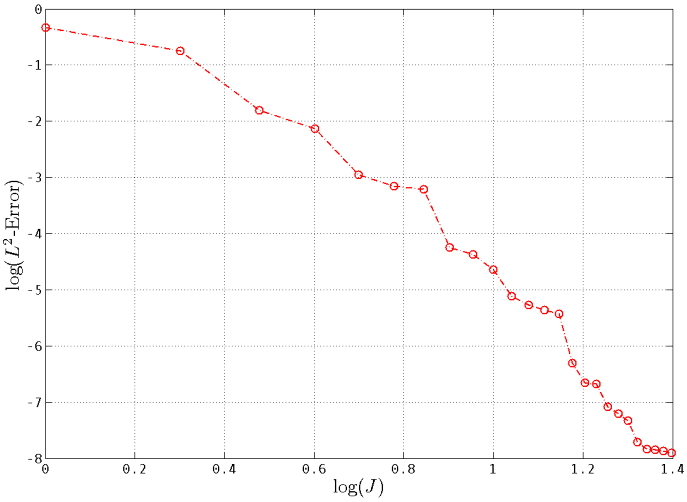
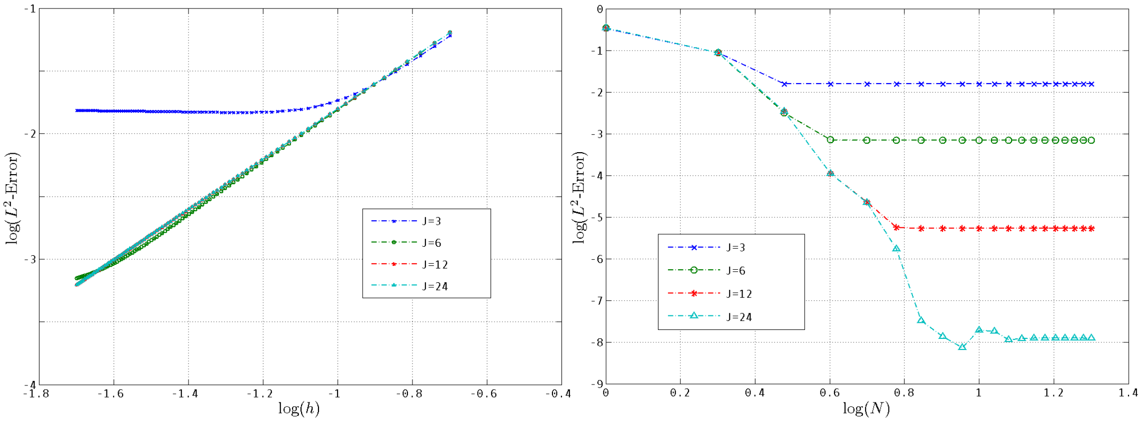


Figure 2.5: Convergence in the Rank for Example 1

Figure 2.5 shows the convergence of the PGD algorithm as the rank of our approximation is increased using the same fixed discretisation that was used in Figure 2.4. From this it is clear that the PGD converges monotonically at an exponential rate. Note that the rate of convergence does tail off near the end but this is due to the approximation being limited by the error in the discretisation.



(a) h-refinement

(b) p-refinement

Figure 2.6: Convergence of the Spectral Element Approximation

We use this simple Poisson problem to investigate how well spectral element methods work as a discretisation in the PGD. Figure 2.6 shows the h and p-convergence rates for Example 1 where we are using the common terminology in high-order methods that h-refinement refers to increasing the number of elements (where h denotes the mesh width) and where p-refinement refers to increasing polynomial degree on each element (i.e. increasing N). The h-convergence rates in Figure 2.6(a) are plotted for rank $J = 3, 6, 12, 24$ PGD approximations where $N = 1$ is fixed. Apart from the low-rank, $J = 3$, approximation we observe a convergence rate of order $O(h^2)$. This is the optimal rate of convergence one would expect for linear elements. When $J = 3$ we notice that the rate of convergence tails off for smaller values of h. The reason for this is that we are refining the discretisation of a low-rank approximation of the solution and not the solution itself. For this reason it is not worth over-refining the discretisation space if the rank of the PGD approximation is low since no benefit is gained from doing so.

The p-convergence rates in Figure 2.6(b) are also plotted for rank $J = 3, 6, 12, 24$ PGD approximations where we have a fixed mesh of $K_x = K_y = 5$ elements in each coordinate direction. One of the major benefits of high-order methods, such as spectral element methods, is that exponential convergence rates are obtained when p-refinement is used (see e.g. Canuto et al. [39]). In the case of $J = 24$ the exponential rate of convergence is clear up until $N = 9$ where the convergence rate tails off much like when $J = 3$ in Figure 2.6(a). For the lower-rank PGD approximations this tailing off occurs for even smaller values of N, in the case of $J = 3, 6$ this completely eradicates any indication of exponential convergence in N. The reason for this tailing off is the same as in Figure 2.6(a) and it is more noticeable for p-convergence due to the faster rate of convergence.

In summary, provided the rank of the PGD approximation is sufficiently high, we observe optimal rates of convergence in both h and p. On the other hand, one would typically want to have a particularly low-rank PGD approximation since the main benefit of the PGD is its computational saving. This means that it might not seem like high-order methods are well suited for use with the PGD. However, it could be possible to use high-order methods more optimally to recognise, on the fly, when it is no longer worth refining the approximation in either h or p. This would require error estimators for the PGD, the likes of which have been developed by Ladevèze and Chamoin [90]. While this could lead to some interesting adaptive strategies for PGD algorithms we will not pursue this idea in this thesis. Instead, we focus on convergence of PGD algorithms as the rank of the approximation is increased. We believe this will be a more interesting and informative approach since this type of refinement is quite unique. Furthermore, there have been theoretical results for convergence of PGD algorithms in the rank whereas currently theoretical results related to the discretisation do not exist.

Example 2 (Finite Rank Solutions).

Consider the following three Poisson problems:

$$\begin{aligned} -\nabla^2 u_i &= f_i(x,y) \quad &\text{in} \quad &\Omega = [-1,1]^2 \\ u_i &= 0 \quad &\text{on} \quad &\partial\Omega, \end{aligned}$$

for $i = 1, 2, 3$, where the source terms are given by:

$$\begin{aligned} f_1(x,y) &= 2\pi^2 \sin(\pi x)\sin(\pi y), \\ f_2(x,y) &= f_1(x,y) - 2(x^2-1) + 2(y^2-1), \\ f_3(x,y) &= f_2(x,y) - 2(1+2x^2)(e^{y^2-1}-1)e^{x^2-1} - 2(1+2y^2)(e^{x^2-1}-1)e^{y^2-1}. \end{aligned}$$

The true solutions of these problems are given by:

$$\begin{aligned} u_1(x,y) &= \sin(\pi x)\sin(\pi y), \\ u_2(x,y) &= u_1(x,y) + (x^2-1)(y^2-1), \\ u_3(x,y) &= u_2(x,y) + (e^{x^2-1}-1)(e^{y^2-1}-1). \end{aligned}$$

In contrast to Example 1 these solutions all have finite rank: 1, 2 and 3, respectively. The purpose of this example is to investigate whether or not the progressive PGD is able to capture the natural rank of the solutions.

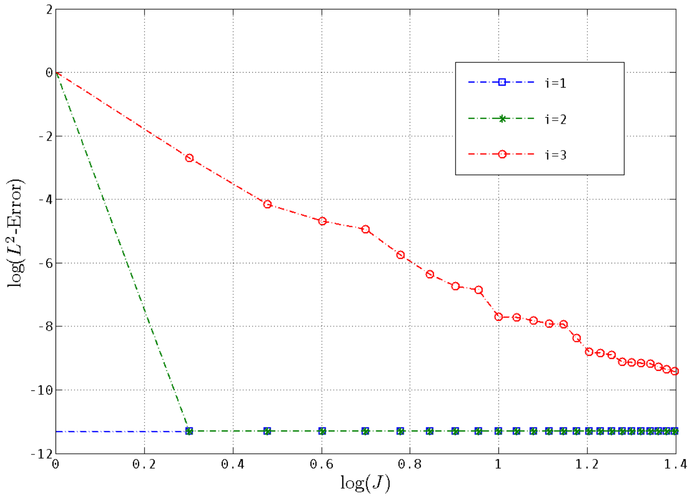
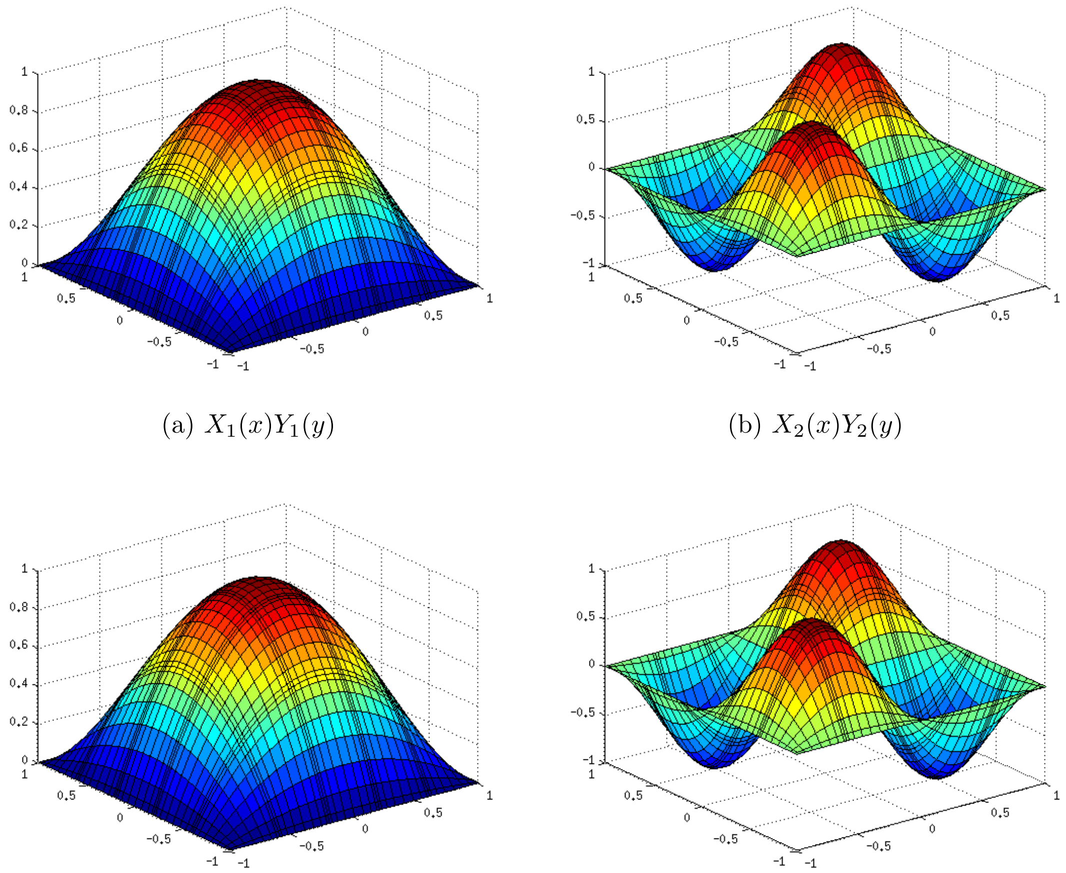


Figure 2.7: Convergence in the Rank for Example 2

Figure 2.7 shows the convergence of each of the three problems with increasing rank of the PGD approximation with a fixed spectral element discretisation of degree $N = 8$ polynomial basis functions over $K_x = K_y = 5$ elements in each coordinate direction. For the rank-1 and rank-2 solutions ($i = 1, 2$) we find that the PGD converges, up to discretisation error, in one and two iterations, respectively. This implies that the natural rank of the solutions of these two problems is captured by the PGD. On the other hand, the rank-3 solution ($i = 3$) does not converge in three iterations and instead we observe convergence behaviour similar to that of the infinite rank case in Example 1.

In Figure 2.8 we have plotted the two non-trivial PGD modes obtained when $i = 2$ and compared them with the two true modes of the solution $u_2(x, y)$. This makes it even clearer that the PGD has successfully captured the natural rank of the solution.



(a) $X_1(x)Y_1(y)$

(b) $X_2(x)Y_2(y)$

(c) $(x^2 - 1)(y^2 - 1)$

(d) $\sin(\pi x)\sin(\pi y)$

Figure 2.8: Comparison of PGD Modes with True Modes for $i = 2$ in Example 2

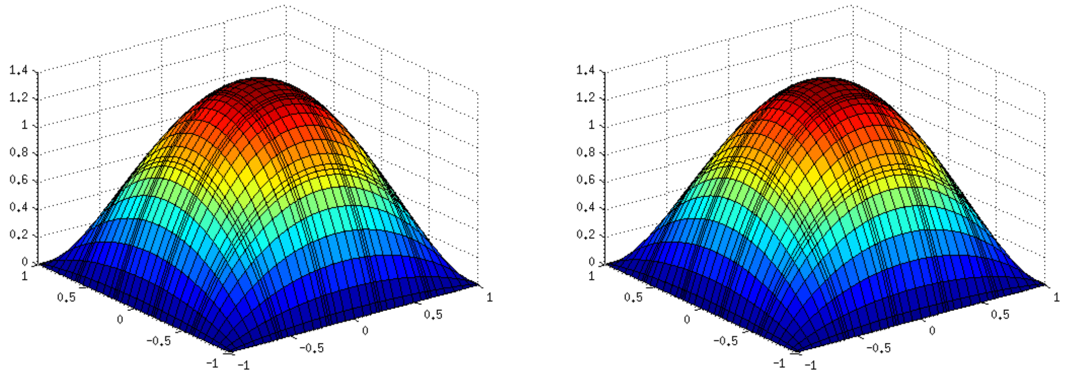
Furthermore, the L^2-norms of each of the true modes are given by:

$$\|(x^2 - 1)(y^2 - 1)\|_{L^2(\Omega)} = \frac{16}{15},$$
$$\|\sin(\pi x)\sin(\pi y)\|_{L^2(\Omega)} = 1.$$

Hence, the progressive PGD algorithm chose the first mode to be the one with the largest norm. In this sense the PGD is optimal for this particular example.

In order to investigate why the rank-3 ($i = 3$) solution was not also obtained optimally by the PGD we have plotted the first PGD mode obtained for this problem in Figure 2.9(a). This does not resemble any of the three true modes but it does happen to be a fairly accurate approximation of a sum of two of the true modes (see Figure 2.9(b)). It is for this reason that the natural rank of the solution could not be found by the PGD algorithm.

In summary, it is possible for the progressive Galerkin PGD algorithm to converge optimally and, in particular, capture the natural finite rank of solutions but it is not guaranteed to do so. This can happen if, for example, a combination of some of the true modes can be well approximated by a rank-one PGD mode.



(a) $X_1(x)Y_1(y)$

(b) $(x^2-1)(y^2-1)+(e^{x^2-1}-1)(e^{y^2-1}-1)$

Figure 2.9: Comparison of First PGD Mode (a) with a Combination of the True Modes (b) for $i=3$ in Example 2

2.3 Non-homogeneous Dirichlet Boundary Conditions

So far we have only considered problems with homogeneous Dirichlet boundary conditions. These are included explicitly in the PGD by imposing zero boundary conditions on each of the PGD modes. This ensures that the product of the modes has homogeneous Dirichlet boundary conditions on the whole of $\partial\Omega$. This only works because the boundary condition is zero. In general, we can only specify a non-zero value on the boundary at the corners of the domain (in the case of a rectangular domain) using this method. Note that non-Dirichlet boundary conditions such a Neumann or Robin boundary conditions are not a problem to implement since they are not imposed explicitly. It is the explicit imposition of non-homogeneous Dirichlet boundary conditions that is not straightforward in the PGD.

This issue was first addressed by González et al. [71] in which they suggested using a method called transfinite interpolation for constructing a function which explicitly satisfies the boundary conditions which can be used in order to impose general boundary conditions in the PGD. This function can then be used to recast the original problem in terms of one with homogeneous Dirichlet boundary conditions. In this section we introduce transfinite interpolation on a rectangle and then extend this result to the high-dimensional geometry of a d-orthotope (a d-dimensional hyper-rectangle). We also briefly discuss other geometries and a way of using transfinite interpolation in the PGD. We conclude with a numerical example.

2.3.1 Transfinite Interpolation on a Rectangle

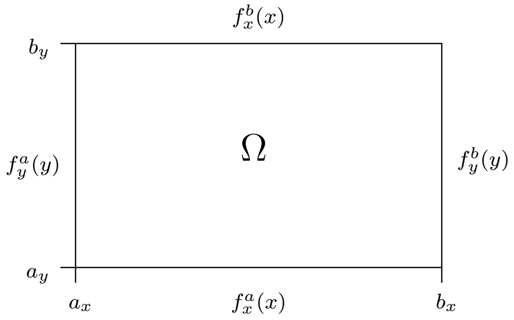


Figure 2.10: A Rectangular Domain

Consider the rectangular domain $\Omega = [a_x, b_x] \times [a_y, b_y]$ with prescribed boundary values $f_x^a(x), f_x^b(x), f_y^a(y)$ and $f_y^b(y)$ as shown in Figure 2.10. Using transfinite interpolation we are able to construct a continuous function $T(x, y)$ which satisfies these boundary values. It is given by (see Gordon and Hall [73]):

$$\begin{aligned} T(x,y) = \frac{-1}{(b_x-a_x)(b_y-a_y)} \Big[&(x-a_x)(b_y-y)f_x^a(b_x) + (x-a_x)(y-a_y)f_x^b(b_x) \\ &+(b_x-x)(b_y-y)f_x^a(a_x) + (b_x-x)(y-a_y)f_x^b(a_x) \Big] \\ &+\frac{1}{(b_x-a_x)}\Big((b_x-x)f_y^a(y) + (x-a_x)f_y^b(y) \Big) \\ &+\frac{1}{(b_y-a_y)}\Big((b_y-y)f_x^a(x) + (y-a_y)f_x^b(x) \Big) \qquad (2.17) \end{aligned}$$

Note that since we require that $T(x, y)$ is continuous we can assume that the boundary values agree on the corners of the domain Ω. In other words we have that:

$$f_y^a(a_y) = f_x^a(a_x),\ f_y^a(b_y) = f_x^b(a_x),\ f_y^b(b_y) = f_x^b(b_x) \text{ and } f_y^b(a_y) = f_x^a(b_x).$$

Armed with this knowledge the interpolating function is relatively simple to derive and it can easily be checked by evaluating $T(x, y)$ at the boundaries of Ω. For example:

$$\begin{aligned}T(x,a_y) =&\frac{-1}{(b_x-a_x)}\Bigg[(x-a_x)f_x^a(b_x)+(b_x-x)f_x^a(a_x)\Bigg]\\&+\frac{1}{(b_x-a_x)}\Bigg((b_x-x)f_y^a(a_y)+(x-a_x)f_y^b(a_y)\Bigg)+f_x^a(x)\\=&f_x^a(x),\end{aligned}$$

as required. This idea can also be extended to higher dimensional domains.

2.3.2 Transfinite Interpolation on a d-Orthotope

Consider the d-orthotope $\Omega = \prod_{i=1}^{d}[a_i,b_i]$ with prescribed boundary values $f_i^a(\mathbf{x}_i), f_i^b(\mathbf{x}_i), i=1,\dots,d$, where

$$\mathbf{x}_i = (x_1,\dots,x_{i-1},x_{i+1},\dots,x_d)^T.$$

We seek a function $T(\mathbf{x})$, $\mathbf{x}=(x_1,\dots,x_d)^T$, that attains these values at the boundary of Ω. Firstly, we define the following projections:

$$A_i\mathbf{x} := (x_1,\dots,x_{i-1},a_i,x_i,\dots,x_d)^T,\ \ i=1,\dots,d,$$

$$B_i\mathbf{x} := (x_1,\dots,x_{i-1},b_i,x_i,\dots,x_d)^T,\ \ i=1,\dots,d.$$

This enables us to express the boundary values in terms of T:

$$f_i^a(\mathbf{x}_i)=T(A_i\mathbf{x}),\ \ f_i^b(\mathbf{x}_i)=T(B_i\mathbf{x}).$$

To greatly simplify the notation we now only express the boundary values in this way. We present the following Theorem:

Theorem 1. *The transfinite interpolating function $T(\boldsymbol{x})$ on a d-orthotope is given by*

$$T(\boldsymbol{x}) = \sum_{I\in\mathcal{P}[d]\setminus\emptyset}(-1)^{|I|+1}\Bigg(\prod_{i\in I}C_iT\Bigg)(\boldsymbol{x}), \tag{2.18}$$

where C_i, $i=1,\dots,d$ are the degree 1 interpolation operators:

$$(C_iT)(\boldsymbol{x}) := \frac{1}{(b_i-a_i)}\Bigg((b_i-x_i)T(A_i\boldsymbol{x})+(x_i-a_i)T(B_i\boldsymbol{x})\Bigg).$$

Remark 1. *We have used set theory notation with $[d]=\{1,\dots,d\}$, $\mathcal{P}[d]$ denotes the power set of $[d]$ (i.e. the set of all subsets of $[d]$), $|I|$ denotes the cardinality of I (i.e. the number of elements in I) and $\emptyset$ denotes the empty set.*

Remark 2. *The operators C_i (and A_i, B_i), $i=1,\dots,d$, are commutative and hence the ordering of I is not an issue.*

Proof. To prove that (2.18) is indeed a transfinite interpolating function on the d-orthotope Ω we need only show that it satisfies an arbitrary choice of boundary value at a_n, $n \in [d]$, say. (The same proof holds for b_n by symmetry). Hence we need to show that:

$$T(A_n\mathbf{x}) = \sum_{I\in\mathcal{P}[d]\backslash\emptyset} (-1)^{|I|+1}\left(\prod_{i\in I} C_i T\right)(A_n\mathbf{x}).$$

First consider the case where $I = \{n\}$, then we have the term:

$$\begin{aligned}(C_nT)(A_n\mathbf{x}) &= \frac{1}{(b_n-a_n)}\Big((b_n-a_n)T(A_n^2\mathbf{x}) + (a_n-a_n)T(A_nB_n\mathbf{x})\Big)\\ &= T(A_n\mathbf{x}),\end{aligned}$$

since $A_n^2 = A_n$ as A_n is a projection. It now remains to show that:

$$\sum_{I\in\mathcal{P}[d]\backslash\{\emptyset,\{n\}\}} (-1)^{|I|+1}\left(\prod_{i\in I} C_i T\right)(A_n\mathbf{x}) = 0. \tag{2.19}$$

Consider a general index set $I_1 \in \mathcal{P}[d]\backslash\emptyset$ such that $n \notin I_1$. The related term in the above sum (2.19) is given by:

$$(-1)^{|I_1|+1}\left(\prod_{i\in I_1} C_i T\right)(A_n\mathbf{x}). \tag{2.20}$$

If we now consider the index set $I_2 = I_1 \cup \{n\}$ which has the following term in the sum (2.19):

$$\begin{aligned}(-1)^{|I_2|+1}&\left(\prod_{i\in I_2} C_i T\right)(A_n\mathbf{x}) = (-1)^{|I_2|+1}\left(C_n\prod_{i\in I_1} C_i T\right)(A_n\mathbf{x})\\ =&\frac{(-1)^{|I_2|+1}}{(b_n-a_n)}\left((b_n-a_n)\left(\prod_{i\in I_1} C_i T\right)(A_n^2\mathbf{x}) + (a_n-a_n)\left(\prod_{i\in I_1} C_i T\right)(A_nB_n\mathbf{x})\right)\\ =&(-1)^{|I_2|+1}\left(\prod_{i\in I_1} C_i T\right)(A_n\mathbf{x}) = -(-1)^{|I_1|+1}\left(\prod_{i\in I_1} C_i T\right)(A_n\mathbf{x}),\end{aligned}$$

since $A_n^2 = A_n$ and $|I_2| = |I_1| + 1$. This is exactly the negative of the term in the sum (2.19) associated with the index set I_1 (2.20). Hence for any choice of index set $I_1 \in \mathcal{P}[d]\backslash\{\emptyset,\{n\}\}$ in the sum (2.19) there is a index set $I_2 \in \mathcal{P}[d]\backslash\{\emptyset,\{n\}\}$ unique to the choice of I_1 given by

$$I_2 = \begin{cases} I_1 \cup \{n\}, & \text{if } n \notin I_1\\ I_1\backslash\{n\}, & \text{if } n \in I_1\end{cases}$$

such that

$$(-1)^{|I_1|+1}\bigg(\prod_{i\in I_1} C_i T\bigg)(A_n\mathbf{x}) + (-1)^{|I_2|+1}\bigg(\prod_{i\in I_2} C_i T\bigg)(A_n\mathbf{x}) = 0,$$

hence every term in the sum (2.19) cancels with another and so the whole sum is 0. Therefore the function $T(\mathbf{x})$ satisfies the general boundary value at a_n and hence satisfies all given boundary values and therefore is a transfinite interpolating function on the d-orthotope Ω. □

2.3.3 Other Geometries

The PGD is most naturally applied to cartesian product domains. Indeed, given a problem defined on $\Omega = \Omega_1 \times \cdots \times \Omega_d$ a PGD approximation is sought of the form:

$$u_J(x_1,\ldots,x_d) = \sum_{j=1}^{J} F_j^1(x_1) \times \cdots \times F_j^d(x_d),$$

where each of the separated components $x_i \in \Omega_i$ for $i = 1,\ldots,d$. In this case it is simple to define the separated representation. In particular when we consider a separation into just 1D components (as we considered in Section 2.2) then each of the domains Ω_i, $i = 1,\ldots,d$ is an interval and hence Ω is a d-orthotope. This is why we focused on deriving a transfinite interpolating function for this geometry in the previous section. When the geometry of the domain is not of this form it is less clear how one can define such a separated representation and hence apply a PGD approximation. This issue was once again considered by González et al. [71] in which a PGD approximation of the form:

$$u_J(x_1,\ldots,x_d) = \omega(x_1,\ldots,x_d)\sum_{j=1}^{J} F_j^1(x_1) \times \cdots \times F_j^d(x_d), \tag{2.21}$$

was considered, where $\omega : \Omega^{\blacksquare} \to \mathbb{R}$ is a function satisfying:

$$\begin{aligned} \omega(x_1,\ldots,x_d) &> 0, \quad (x_1,\ldots,x_d) \in \Omega, \\ \omega(x_1,\ldots,x_d) &= 0, \quad (x_1,\ldots,x_d) \in \Omega^{\blacksquare}\backslash\Omega, \end{aligned}$$

where $\Omega^{\blacksquare}$ is the smallest hypercube such that $\Omega \subset \Omega^{\blacksquare}$. This effectively enforces homogeneous Dirichlet boundary conditions on $\partial\Omega$ while ensuring the solution is zero outside the original domain Ω. Furthermore, the problem is now defined on a hypercube (and hence a cartesian product domain) and so the separated representation in (2.21) is well defined.

To construct such a function ω González et al. [71] suggested using R-functions. R-functions were developed by Rvachev (e.g. see [116]) and are essentially functions whose sign is completely governed by the signs of its arguments. They can then be used to define boolean operators such as AND and OR. Domains which can then be expressed as a series of inequalities can be effectively described by a suitable combination of R-functions acting as boolean operators. We direct the interested reader for several examples in [116] and for a specific example of an application to a PGD approximation of the Poisson equation in [71]. Transfinite interpolation and R-functions can also be combined to impose non-homogeneous Dirichlet boundary conditions on non-rectangular domains although this can be a difficult task if the geometry is particularly complicated. (e.g. see Rvachev et al. [117]).

There are a number of difficulties with using the method proposed by González et al. [71]. Firstly, the geometry of Ω needs to be sufficiently simple that an R-function can be calculated and once we have found a suitable function ω then a POD or higher dimensional equivalent needs to be applied in order to obtain an approximation in separated form. Furthermore, ω will not generally be differentiable on $\partial\Omega$ and hence a discontinuous basis method needs to be employed such as the eXtended Finite Element Method (XFEM) [102]. Additionally, high-dimensional problems are not generally going to be defined in non-cartesian product domains as these sorts of geometries typically arise in physical space where the dimension is moderate (≤ 3). Intrinsically high-dimensional problems, such as the Fokker-Planck equation, have components which are defined in their own configuration spaces (e.g. in the Fokker-Planck equation these are the spaces of all possible configurations of a spring in a bead-spring chain model of a polymer) and hence the whole problem is defined in a cartesian product of these individual configuration spaces. Similarly, parametric problems (which are made high-dimensional by including parameters as additional variables) are also defined in a cartesian product space since parameter space is naturally an orthotope as the parameters are chosen to vary over certain intervals.

For all the reasons given above we will only consider the PGD defined on cartesian product spaces in this thesis. However, it is certainly very promising that the PGD can be extended to more complicated geometries if needed.

2.3.4 PGD Implementation of Transfinite Interpolation

To describe how the transfinite interpolating function, $T(\mathbf{x})$, can be used to impose non-homogeneous Dirichlet boundary conditions consider the following boundary value problem:

$$\begin{aligned} \mathcal{L}u = f \quad &\text{in} \quad \Omega = \prod_{i=1}^{d}[a_i, b_i], \\ u = g \not\equiv 0 \quad &\text{on} \quad \partial\Omega, \end{aligned}$$

where $\mathcal{L}$ is some differential operator. We can then employ the following change of variables $\tilde{u} = u - T$, where T is the transfinite interpolating function that satisfies $T = g$ on $\partial\Omega$. This leads to the following problem:

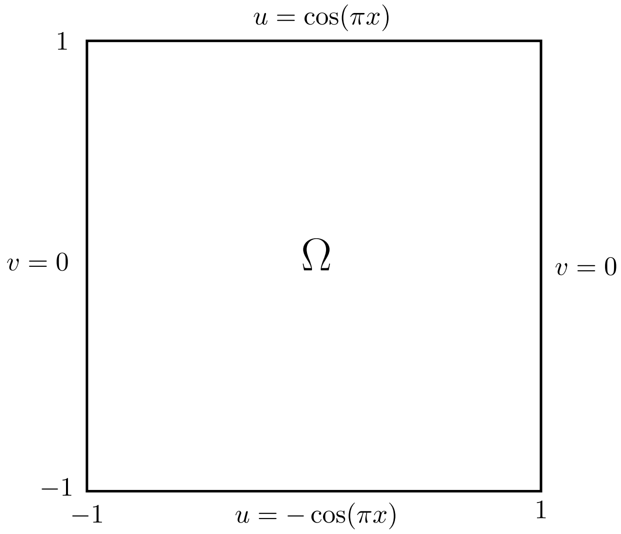
$$\mathcal{L}\tilde{u} = f + \mathcal{L}T \quad \text{in} \quad \Omega = \prod_{i=1}^{d} [a_i, b_i],$$
$$\tilde{u} = 0 \quad \text{on} \quad \partial\Omega.$$

This is now a problem with homogeneous Dirichlet boundary conditions and a separated representation of $\tilde{u}$ can be calculated in the same way as in Section 2.2. Note that, as with the source term f, a POD or higher dimensional equivalent will need to be employed on the transfinite interpolating function T if it does not already possess a finite rank separated representation. In the case of a 2D problem the transfinite interpolating function $T(x, y)$ given by (2.17) conveniently possesses a rank-4 (or lower) separated representation. Once a rank-r (approximate or exact) separated representation T_r of T has been found then we could also consider the set of the first r PGD modes to be the modes of T_r. This effectively increases the rank of our PGD approximation by r but at no real extra cost. This leads to the following generalisation of our progressive PGD algorithm defined by the iterative procedure (2.12):

$$u_j(\mathbf{x}) = u_{j-1}^{(e)}(\mathbf{x}), \quad u_0(x, y) = T_r(\mathbf{x}),$$

for $j = 0, \dots, J$.

In Chapter 3 we will encounter some boundary conditions of the form $\mathbf{n} \times \mathbf{u} = \mathbf{g}$ on $\partial\Omega$, where $\mathbf{n}$ denotes the outward normal unit vector to $\partial\Omega$. To give an example of this, consider a general problem with the following boundary conditions in the square domain $[-1, 1]^2$:



In this case boundary conditions for the components of $\mathbf{u}$ are only defined on certain parts of $\partial\Omega$ when Ω is a d-orthotope. This means that the transfinite interpolating polynomial (2.18) is not defined since we need to know $T(A_i\mathbf{x})$ and $T(B_i\mathbf{x})$ for $i = 1, \dots, d$ and not just where the boundary conditions are defined. We remedy this by simply linearly interpolating between vertices of known boundary conditions to ensure continuity of $T(\mathbf{x})$. In the particular case of the problem above we have boundary condition of the type $\mathbf{n} \times \mathbf{u} = \mathbf{g}$ on $\partial\Omega$, where $\mathbf{u} = (u, v)^T$. The boundary conditions on v are homogeneous and so a transfinite interpolating function does not need to be constructed for this. However, u has non-homogeneous Dirichlet boundary conditions on the top and bottom parts of $\partial\Omega$ but is not defined on the left or right parts of $\partial\Omega$. In order to construct a transfinite interpolating function for u we therefore linearly interpolate between the values of u at the vertices $(-1, -1)$ and $(-1, 1)$, and between vertices $(1, -1)$ and $(1, 1)$, which in this case leads to:

$$T(-1, y) = T(1, y) = -y, \tag{2.22}$$

and hence the transfinite interpolating function (2.17) in 2D is calculated to be:

$$T(x, y) = y\cos(\pi x),$$

(see Figure 2.11).

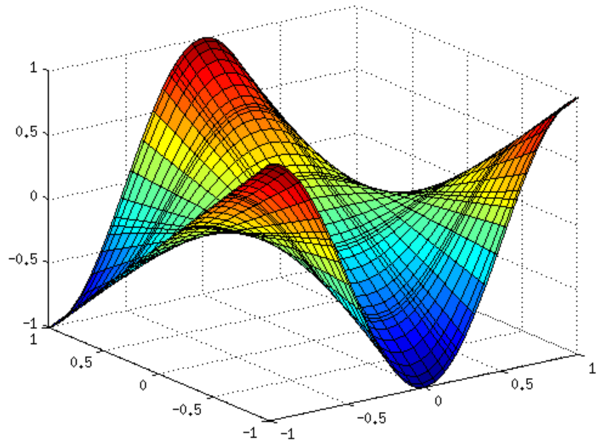


Figure 2.11: The Transfinite Interpolating Function $T(x, y) = y\cos(\pi x)$

Applying a discretisation of the form (2.8)-(2.9) to the PGD modes of u we then explicitly impose the boundary conditions by setting $\beta_{j,0,1} = \beta_{j,N,K_y} = 0$, $j = 1, \dots, J$. By not enforcing $\alpha_{j,0,1} = \alpha_{j,N,K_x} = 0$, $j = 1, \dots, J$, we allow the algorithm to update the solution on the the parts of the boundary where we imposed the linear artificial boundary conditions (2.22) in order to define our transfinite interpolating function.

Applications of transfinite interpolation to boundary conditions of this type can be found in Chapter 3 where they arise in certain first order reformulations of the

Poisson and Stokes problems. We will now demonstrate how transfinite interpolation works for standard Dirichlet boundary conditions on the whole boundary by considering a specific example.

2.3.5 Numerical Example

To give an example of transfinite interpolation being utilised to impose non-homogeneous Dirichlet boundary conditions we revisit the Poisson problem in 2D.

Example 3 (Non-homogeneous Dirichlet Boundary Conditions)**.**

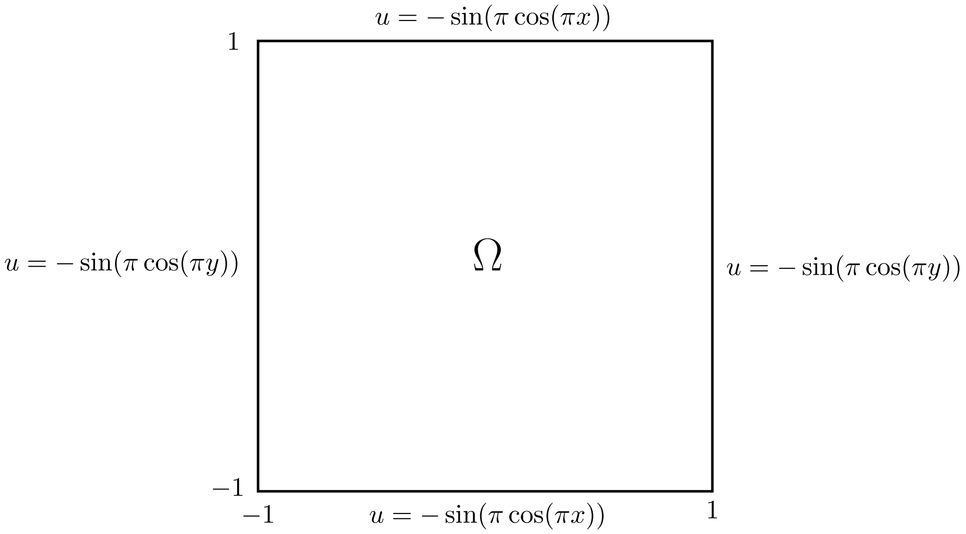
Consider the Poisson problem:

$$-\nabla^2 u = f(x,y) \quad \text{in} \quad \Omega = [-1,1]^2$$

with source term:

$$\begin{aligned} f(x,y) = {} & \pi^4(\sin^2(\pi x)\cos^2(\pi y) + \cos^2(\pi x)\sin^2(\pi y))\sin(\pi\cos(\pi x)\cos(\pi y)) \\ & + 2\pi^3\cos(\pi x)\cos(\pi y)\cos(\pi\cos(\pi x)\cos(\pi y)), \end{aligned}$$

and with boundary conditions as pictured below:


$u = -\sin(\pi\cos(\pi x))$
1
$u = -\sin(\pi\cos(\pi y))$
Ω
$u = -\sin(\pi\cos(\pi y))$
-1
-1
$u = -\sin(\pi\cos(\pi x))$
1

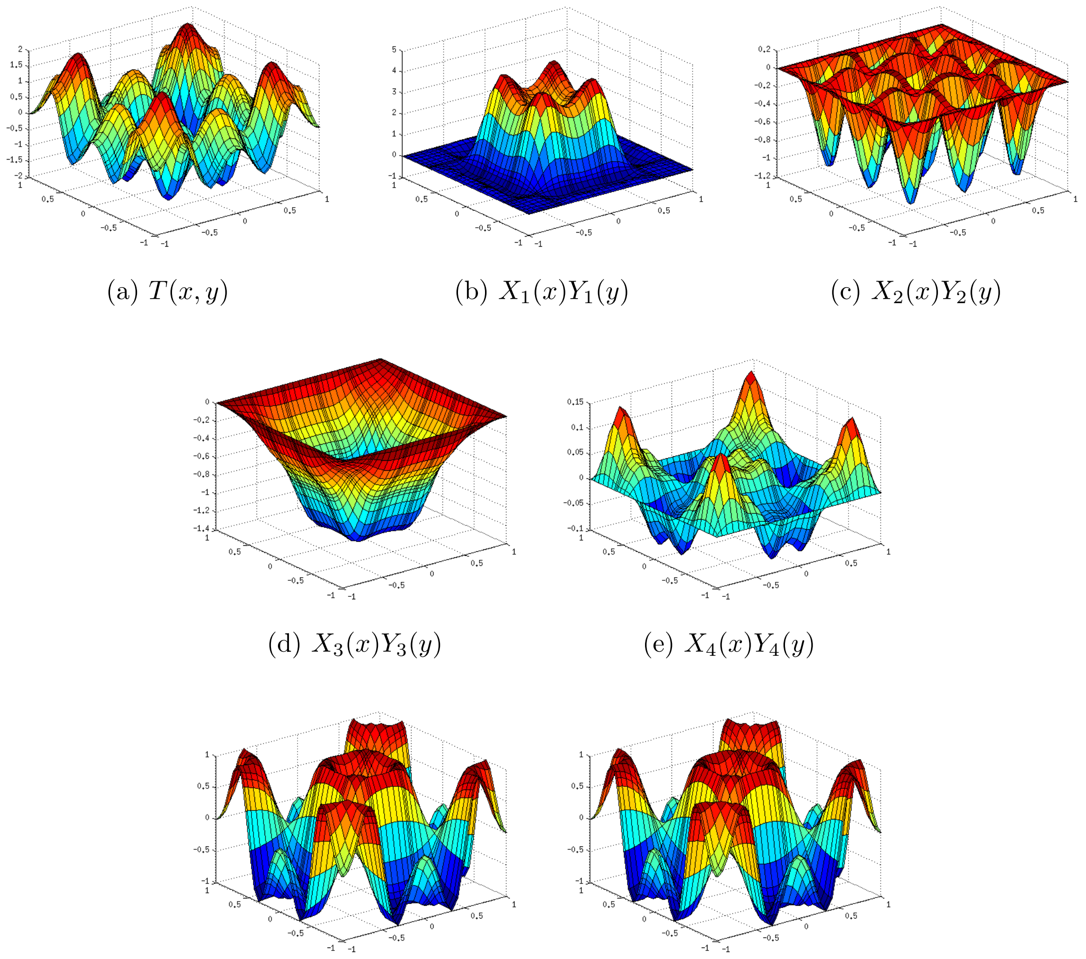
This problem has the exact solution $u(x,y) = \sin(\pi\cos(\pi x)\cos(\pi y))$. We begin by constructing a transfinite interpolating function which satisfies the given boundary conditions. Using equation (2.17) we obtain the following for this example:

$$T(x,y) = -(\sin(\pi\cos(\pi x)) + \sin(\pi\cos(\pi y)).$$

Figure 2.12(a) shows this transfinite interpolating function as well as the first 4 PGD modes in Figures 2.12(b)-(e). These are summed to obtain the rank-4 (or rank-6 if you include $T(x, y)$) approximate PGD solution:

$$u_4(x, y) = T(x, y) + \sum_{j=1}^{4} X_j(x)Y_j(y),$$

which is plotted in Figure 2.12(f) and which is indistinguishable from the true solution in Figure 2.12(g). The discretisation used for this was the same as in Examples 1 and 2.



(a) $T(x, y)$

(b) $X_1(x)Y_1(y)$

(c) $X_2(x)Y_2(y)$

(d) $X_3(x)Y_3(y)$

(e) $X_4(x)Y_4(y)$

(f) $u_4(x, y)$

(g) True Solution

Figure 2.12: Rank-4 PGD Modes and PGD Approximation for Example 3

To verify that the progressive PGD algorithm converges when using transfinite interpolation to impose non-homogeneous Dirichlet boundary conditions we have plotted the error in increasing rank of the PGD approximation in Figure 2.13.

As with Example 1 (since this example also has infinite rank solution) we find that we obtain monotonic convergence at an exponential rate until the error tails off near the end due to the limiting error in the discretisation.

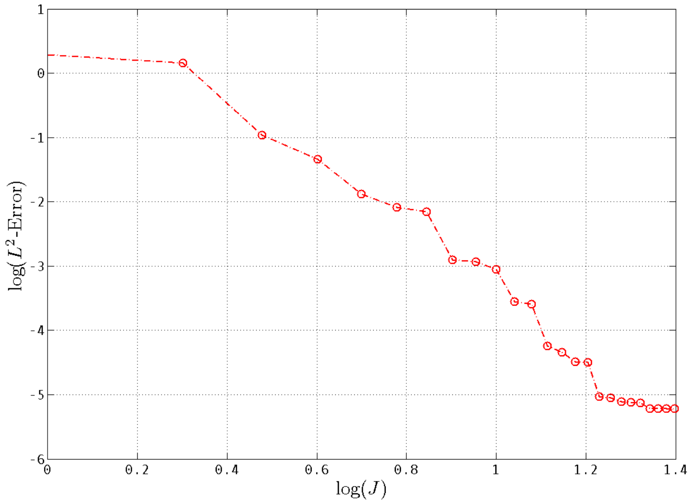


Figure 2.13: Convergence in the Rank for Example 3

2.3.6 Concluding Remarks

In Section 2.2 we introduced the progressive Galerkin PGD in the context of solving the Poisson equation in 2D with homogeneous Dirichlet boundary conditions. We observed that this PGD algorithm converged monotonically and exponentially as the rank of the PGD approximation was increased. Furthermore, in Section 2.3, we showed how the algorithm could be extended to treat non-homogeneous Dirichlet boundary conditions and the same observations of convergence were made.

In the next section we review some of the theoretical results for progressive PGD algorithms with the objective of proving convergence of the progressive Galerkin PGD algorithm thereby verifying our observations.

2.4 Convergence Analysis

In order to prove convergence of the progressive PGD we need to consider it in a theoretical setting. To do this we express separated representations as tensor decompositions. Note that we have already been using the term tensor when describing the PGD and we now introduce this more formally. We begin by defining the following tensor product:

$$\bigotimes_{i=1}^{d} v_i : (x_1, \ldots, x_d) \mapsto \prod_{i=1}^{d} v_i(x_i). \tag{2.23}$$

This notation can be used to rewrite the separated representation (2.2) as the following tensor decomposition:

$$\sum_{j=1}^{J} \prod_{i=1}^{d} F_j^i(x_i) = \sum_{j=1}^{J} \bigotimes_{i=1}^{d} F_j^i, \tag{2.24}$$

where $F^i_j \in V_i$ $(i = 1, \ldots, d)$ for all $j = 1, \ldots, J$. For a tensor, U, of this form we define the tensor rank, $\operatorname{rank}_\otimes U$, to be the smallest value of J such that U can be expressed in the form given in (2.24). We now present the theoretical setting for progressive PGD algorithms.

2.4.1 Theoretical Setting

Let $\Omega_1, \ldots, \Omega_d$ be open sets of $\mathbb{R}^{n_1}, \ldots, \mathbb{R}^{n_d}$, respectively, and let $V_1, \ldots, V_d$ be Hilbert spaces of functions on $\Omega_1, \ldots, \Omega_d$, with inner products $\langle \cdot, \cdot \rangle_1, \ldots, \langle \cdot, \cdot \rangle_d$ and associated norms $\| \cdot \|_1, \ldots, \| \cdot \|_d$, respectively. If we let $\mathcal{S}_1$ denote the set of all rank-one tensors under the tensor product (2.23) i.e.

$$\mathcal{S}_1 = \left\{ \bigotimes_{i=1}^{d} v_i \mid (v_1, \ldots, v_d) \in \prod_{i=1}^{d} V_i \right\},$$

then we can define the tensor product Hilbert space, $V_\otimes := \bigotimes_{i=1}^{d} V_i$, to be the closure of $\operatorname{Span}(\mathcal{S}_1)$ under the cross-norm $\| \cdot \|_\otimes$ (see Cancès et al. [37]) which is defined by:

$$\left\| \bigotimes_{i=1}^{d} v_i \right\|_\otimes = \prod_{i=1}^{d} \|v_i\|_i.$$

The formal definition of the tensor product Hilbert space is then given by

$$V_\otimes = \overline{\operatorname{Span}(\mathcal{S}_1)}^{\| \cdot \|_\otimes}.$$

We further define the subsets $\mathcal{S}_n \subset V_\otimes$, for $n \geq 2$, which were first introduced by de Silva and Lim [56]:

$$\mathcal{S}_n = \left\{ u \in V_\otimes : u = \sum_{j=1}^{J} u^{(j)},\ u^{(j)} \in \mathcal{S}_1,\ J \leq n \right\}.$$

Note that $\mathcal{S}_n \subset \mathcal{S}_{n+1}$ and we say that $\operatorname{rank}_\otimes U = n$ if and only if $U \in \mathcal{S}_n \backslash \mathcal{S}_{n-1}$. This means that the rank-n PGD approximations we obtain can be thought of as elements of the tensor product Hilbert space $V_\otimes \supset \mathcal{S}_n$.

In order to prove convergence of the progressive PGD we need to think of it as a greedy algorithm. Greedy algorithms essentially work by finding iteratively the member of a given set (known as the dictionary) which minimises some quantity. Recall that in the progressive PGD we seek, in some way, the ‘best’ rank-one tensor at each iteration. If we define the ‘best’ rank-one tensor to be the one which minimises some quantity which is equivalent to solving the weak formulation of the PDE then it is clear that the progressive PGD can be thought of as a greedy algorithm where the dictionary is the set of all rank-one tensors, $\mathcal{S}_1$.

For elliptic symmetric problems it is possible to recast the weak formulation of a problem given by (2.1) as the following minimisation problem:

$$u = \arg \min_{v \in V(\Omega)} \left(\frac{1}{2} a(v, v) - L(v) \right). \tag{2.25}$$

This minimisation problem is known as a Rayleigh-Ritz setting and we will give a more detailed explanation of this in the upcoming section. The Galerkin orthogonality criterion is then equivalent to solving the Euler-Lagrange equations associated with (2.25). Provided that the functional we are minimising is convex and we seek solutions in a linear space $V(\Omega)$ then this Euler-Lagrange equation is equivalent to solving the minimisation problem itself. Unfortunately, in the PGD we iteratively seek solutions which are in a nonlinear manifold, $\mathcal{S}_1$, which is embedded in a linear space, $V_\otimes$, the main point being that $\mathcal{S}_1$ is not a linear subspace of $V_\otimes$. In this sense the minimisation problem is not equivalent to solving its Euler-Lagrange equations and hence the greedy algorithm is not the same as the Galerkin progressive PGD in practice. The reason we do not simply employ the greedy algorithm in practice by solving the minimisation problem directly is that this is a very computationally expensive task. Solving the Euler-Lagrange equations (i.e. Galerkin orthogonality) should be thought of as a computational strategy only and to progress theoretically with the PGD one should think of the progressive PGD as a greedy algorithm in which the minimisation is solved. Throughout this thesis we may use phrases similar to 'proving convergence of the progressive PGD' by which we, strictly speaking, mean 'proving convergence of a greedy algorithm associated with the progressive PGD'.

Greedy algorithms of this type have been explored previously in nonlinear approximation theory (see Temylakov [127], for example). A number of convergence results for these greedy algorithms has been provided in the context of the PGD, a brief review of which was provided in Section 1.3. In this section we will provide a more in depth description of two very different approaches to proving convergence of progressive PGD algorithms. The first proof is specific to the Poisson equation and was provided by Le Bris et al. [95] (we also describe a generalisation of this result to linear and nonlinear elliptic self-adjoint problems by Cancès et al. [37]). The second proof we consider is the more abstract proof of Falcó and Nouy [66] which is based on a generalisation of the classic result of Eckart and Young [61].

2.4.2 Proof A: Energy Minimisation

Consider the weak formulation: Find $u \in V(\Omega)$ such that:

$$a(u, u^*) = L(u^*), \quad \forall u^* \in V(\Omega), \tag{2.26}$$

where $a(\cdot,\cdot)$ and $L(\cdot)$ are bilinear and linear forms, respectively. In the previous section we stated that, provided $a(\cdot,\cdot)$ is symmetric (i.e. $a(u,v)=a(v,u)$, $\forall u,v \in V(\Omega)$), then this is equivalent to solving the following energy minimisation problem:

$$u = \arg \min_{v\in V(\Omega)} \left(\frac{1}{2} a(v,v) - L(v) \right).$$

This well known result can be easily derived by considering the Euler-Lagrange equations for the above energy functional:

$$\begin{aligned} 0 &= \lim_{\varepsilon\to 0} \frac{d}{d\varepsilon} \left(\frac{1}{2} a(u+\varepsilon u^*, u+\varepsilon u^*) - L(u+\varepsilon u^*) \right) \\ &= \lim_{\varepsilon\to 0} \left(a(u,u^*) + \varepsilon a(u^*,u^*) - L(u^*) \right) \\ &= a(u,u^*) - L(u^*), \quad \forall u^* \in V(\Omega), \end{aligned}$$

which is exactly the weak problem (2.26). The proof we present here was provided by Le Bris et al. [95] for the specific case of the Poisson equation. In this case our energy minimisation problem is the minimisation of Dirichlet energies in $H_0^1(\Omega)$:

$$u = \arg \min_{v\in H_0^1(\Omega)} \int_\Omega \left(\frac{1}{2} |\nabla v|^2 - fu \right) d\Omega. \tag{2.27}$$

We begin by defining a greedy algorithm based on this minimisation which is most relevant to the application of the progressive PGD.

There are two main greedy algorithms which were investigated by Temylakov [127]: the pure and orthogonal greedy algorithms. The orthogonal greedy algorithm includes a projective step the likes of which were employed in progressive PGD algorithms by Ammar et al. [8, 9] and Leonenko and Phillips [96], for example. The pure greedy algorithm is the same but without this projective step. Since we do not consider the PGD with a projective step in this thesis we will only concentrate on results for the pure greedy algorithm. In 2D it is defined as follows:

Algorithm 2 Pure Greedy Algorithm

```
Input: $(f,\ \epsilon)$
  $f_0 = f$
  $n = 0$
  while $\|f_n\| \geq \epsilon$ do
    $n \leftarrow n+1$
    $(X_n, Y_n) =$ procedure(minEnergy)
    $f_n = f_{n-1} + \nabla^2(X_n \otimes Y_n)$
  end while
```

This algorithm generalises to higher dimensions analogously. For brevity we only consider the proof of convergence in two dimensions in this section.

Algorithm 2 effectively describes our Galerkin progressive PGD algorithm where instead of solving Galerkin orthogonality in order to define the next PGD modes we instead employ the *minEnergy* procedure which is defined as the minimisation of the energy functional:

$$(X_n, Y_n) = \arg \min_{(r,s)\in H_0^1(\Omega_x)\times H_0^1(\Omega_y)} \int_\Omega \left\{ \frac{1}{2} |\nabla(r\otimes s)|^2 - f_{n-1} r \otimes s \right\} d\Omega, \tag{2.28}$$

where $\Omega = \Omega_x \times \Omega_y$. A Lemma in [95] states that:

$$r\otimes s \in H_0^1(\Omega) \Longleftrightarrow r \in H_0^1(\Omega_x) \quad \text{and} \quad s \in H_0^1(\Omega_y),$$

and since we have that:

$$r \otimes s \in \mathcal{S}_1 \subset V_\otimes = H_0^1(\Omega_x) \otimes H_0^1(\Omega_y) \subset H_0^1(\Omega),$$

this justifies the notation used in (2.28). Furthermore, Le Bris et al. [95] proved that the iterations in Algorithm 2 are well defined. This enables us to ask the question of whether the pure greedy algorithm converges to the true solution of the Poisson equation.

To answer this we begin by defining the sequence of functions u_n which satisfy the Dirichlet problems:

$$-\nabla^2 u_n = f_n \quad \text{in} \quad \Omega, \tag{2.29}$$

$$u_n = 0 \quad \text{on} \quad \partial\Omega, \tag{2.30}$$

where f_n is as defined in Algorithm 2. Notice that $u_n = u_{n-1} - X_n \otimes Y_n$ and hence:

$$u_n = u - \sum_{i=1}^{n} X_i \otimes Y_i. \tag{2.31}$$

Therefore proving that pure greedy algorithm converges amounts to proving that u_n converges to 0.

In this section we shall endow the functional space $H_0^1(\Omega)$ with the inner product:

$$\langle u, v\rangle = \int_\Omega \nabla u \cdot \nabla v \; d\Omega$$

and the associated H^1 seminorm (although convergence can also be proven in the full H^1 norm):

$$\|u\|^2 = \langle u, u\rangle = \int_\Omega |\nabla u|^2 \; d\Omega.$$

We now also define the Euler-Lagrange equation associated with the minimisation (2.28) which is given by (see Le Bris et al. [95]): Find $(X_n, Y_n) \in H_0^1(\Omega_x) \times H_0^1(\Omega_y)$ such that:

$$\int_\Omega \nabla(X_n \otimes Y_n) \cdot \nabla(X_n \otimes s + r \otimes Y_n) \; d\Omega = \int_\Omega f_{n-1}(X_n \otimes s + r \otimes Y_n) \; d\Omega, \quad (2.32)$$

for all $(r, s) \in H_0^1(\Omega_x) \times H_0^1(\Omega_y)$. We can also write this in terms of the sequence u_n defined in (2.31) by considering that:

$$\begin{aligned}
&\langle u_n, X_n \otimes s + r \otimes Y_n \rangle = \langle u_{n-1} - X_n \otimes Y_n, X_n \otimes s + r \otimes Y_n \rangle \\
&= \int_\Omega \nabla(u_{n-1} - X_n \otimes Y_n) \cdot \nabla(X_n \otimes s + r \otimes Y_n) \; d\Omega \\
&= -\int_\Omega \nabla^2 u_{n-1}(X_n \otimes s + r \otimes Y_n) \; d\Omega - \int_\Omega \nabla(X_n \otimes Y_n) \cdot \nabla(X_n \otimes s + r \otimes Y_n) \; d\Omega \\
&= \int_\Omega f_{n-1}(X_n \otimes s + r \otimes Y_n) \; d\Omega - \int_\Omega \nabla(X_n \otimes Y_n) \cdot \nabla(X_n \otimes s + r \otimes Y_n) \; d\Omega \\
&= 0,
\end{aligned}$$

by using the Euler-Lagrange equation (2.32) and the fact that u_{n-1} satisfies the Dirichlet problem (2.29)-(2.30). This leads to the following equivalent expression of (2.32) which will be of use in the proof of Theorem 2:

$$\langle u_n, X_n \otimes s + r \otimes Y_n \rangle = 0. \quad (2.33)$$

Before we give an outline of the proof of convergence given by Le Bris et al. [95] we first present the following two Lemmas:

Lemma 1. *Let $h : \Omega \longrightarrow \mathbb{R}$ be a locally integrable function with corresponding distribution $T_h \in \mathcal{D}'(\Omega)$ such that, for any functions $(\phi, \psi) \in \mathcal{D}(\Omega_x) \times \mathcal{D}(\Omega_y)$:*

$$\langle T_h, \phi \otimes \psi \rangle_{(\mathcal{D}'(\Omega), \mathcal{D}(\Omega))} := \int_\Omega h(\phi \otimes \psi) \; d\Omega = 0$$

then $h = 0$ almost everywhere in Ω.

The proof of this Lemma is well known in distribution theory and hence is not reproduced here.

Lemma 2. *The functions (X_n, Y_n) that minimise (2.28) are such that: $\forall (r, s) \in H_0^1(\Omega_x) \times H_0^1(\Omega_y)$*

$$\|X_n \otimes Y_n\| = \frac{\langle X_n \otimes Y_n, u_{n-1} \rangle}{\|X_n \otimes Y_n\|} \geq \frac{\langle r \otimes s, u_{n-1} \rangle}{\|r \otimes s\|},$$

where u_{n-1} is as defined in (2.31).

Proof. See Le Bris et al. [95]. □

Theorem 2 (Convergence of the Pure Greedy Algorithm)**.** *In Algorithm 2 assume that* (X_n, Y_n) *satisfies the Euler-Lagrange equation* (2.32)*. Let us denote the energy at iteration* n *as*

$$E_n = \frac{1}{2}\int_\Omega \left|\nabla(X_n \otimes Y_n)\right|^2 \; d\Omega - \int_\Omega f_{n-1} X_n \otimes Y_n \; d\Omega$$

Then we have that

$$\sum_{n=1}^{\infty} \int_\Omega \left|\nabla(X_n \otimes Y_n)\right|^2 = -2\sum_{n=1}^{\infty} E_n < \infty.$$

Assume also that (X_n, Y_n) *is a minimiser of* (2.28)*. Then we have that*

$$\lim_{n\to\infty} u_n = 0 \quad in \;\; H_0^1(\Omega)$$

and so the pure greedy algorithm converges.

Proof. We present here an outline of the proof given by Le Bris et al. [95] by separating it into three main steps:

Step 1 (The sequence converges):
Assuming that (X_n, Y_n) satisfies the Euler-Lagrange equation (2.32) notice that

$$\left\|u_{n-1}\right\|^2 = \left\|u_n + X_n \otimes Y_n\right\|^2 = \left\|u_n\right\|^2 + \left\|X_n \otimes Y_n\right\|^2,$$

since $\langle u_n, X_n \otimes Y_n\rangle = 0$ by taking $r = X_n$ and $s = 0$ in the Euler-Lagrange equation (2.33). This tells us that $\left\|u_n\right\|^2$ is a convergent sequence which implies that

$$\sum_{n=1}^{\infty} \left\|X_n \otimes Y_n\right\|^2 = \sum_{n=1}^{\infty} \int_\Omega \left|\nabla(X_n \otimes Y_n)\right|^2 \; d\Omega < \infty$$

We also have that

$$\begin{aligned} E_n =& \frac{1}{2}\int_\Omega \left|\nabla(X_n \otimes Y_n)\right|^2 \; d\Omega - \int_\Omega f_{n-1} X_n \otimes Y_n \; d\Omega \\ =& \frac{1}{2}\int_\Omega \left|\nabla(X_n \otimes Y_n)\right|^2 \; d\Omega - \int_\Omega \nabla u_{n-1} \cdot \nabla(X_n \otimes Y_n) \; d\Omega \\ =& -\frac{1}{2}\int_\Omega \left|\nabla(X_n \otimes Y_n)\right|^2 + \int_\Omega \nabla u_n \cdot \nabla(X_n \otimes Y_n) \; d\Omega \\ =& -\frac{1}{2}\int_\Omega \left|\nabla(X_n \otimes Y_n)\right|^2, \end{aligned}$$

since $\langle u_n, X_n \otimes Y_n\rangle = 0$. This proves the first part of the Theorem.

Step 2 (The sequence weakly converges to 0):
Assume now that (X_n, Y_n) also satisfies the minimisation problem (2.28). We know from Step 1 that $\left\|u_n\right\|^2$ is a bounded sequence and therefore there exists a subse-

quence of u_n which converges weakly in $H_0^1(\Omega)$ to some $u_\infty \in H_0^1(\Omega)$. Using the fact that $\lim\limits_{n\to\infty} E_n = 0$ and that (X_n, Y_n) minimises E_n we find that for any functions $(r, s) \in H_0^1(\Omega_x) \times H_0^1(\Omega_y)$:

$$\int_\Omega \nabla u_\infty \cdot \nabla(r \otimes s)\ d\Omega = 0.$$

Lemma 1 then implies that $-\Delta u_\infty = 0$ almost everywhere. Therefore, since $u_\infty \in H_0^1(\Omega)$, we must have that $u_\infty = 0$. This means there is only one possible limit of a subsequence of u_n and hence the sequence itself converges weakly to 0.

Step 3 (The sequence strongly converges to 0):
Firstly, we find that for any $n \geq m \geq 0$:

$$\begin{aligned}\|u_n - u_m\|^2 &= \|u_n\|^2 + \|u_m\|^2 - 2\left\langle u_n, \left(u_n + \sum_{k=m+1}^{n} X_k \otimes Y_k\right)\right\rangle \\ &= \|u_n\|^2 + \|u_m\|^2 - 2\|u_n\|^2 - 2\sum_{k=m+1}^{n} \langle u_n, X_k \otimes Y_k\rangle \\ &\leq -\|u_n\|^2 + \|u_m\|^2 + 2\sum_{k=m+1}^{n} \|X_k \otimes Y_k\| \, \|X_{n+1} \otimes Y_{n+1}\|,\end{aligned}$$

by using Lemma 2 and (2.31). If we now define

$$\phi(k+1) = \begin{cases} 1, & \text{if } k = 0, \\ \arg\min\limits_{n>\phi(k)} \left\{ \|X_n \otimes Y_n\| \leq \left\|X_{\phi(k)} \otimes Y_{\phi(k)}\right\| \right\}, & \text{otherwise,}\end{cases}$$

we have that $\lim\limits_{k\to\infty} \phi(k) = \infty$ since $\lim\limits_{k\to\infty} \|X_k \otimes Y_k\| = 0$ from the first part of the proof. Now using the previous inequality, we have that for any $l \geq k \geq 0$:

$$\left\|u_{\phi(l)-1} - u_{\phi(k)-1}\right\|^2 \leq -\left\|u_{\phi(l)-1}\right\|^2 + \left\|u_{\phi(k)-1}\right\|^2 + 2\sum_{i=\phi(k)}^{\phi(l)-1} \|X_i \otimes Y_i\|^2$$

This shows that the subsequence $(u_{\phi(k)-1})_{k\geq 0}$ is a Cauchy sequence and hence strongly converges to 0 (since we know u_n weakly converges to 0). Since $\|u_n\|$ is itself a converging sequence we have that

$$\lim_{n\to\infty} \|u_n\| = 0.$$

□

This concludes this proof of convergence for the pure greedy algorithm applied to the Poisson equation. A more general result which is applicable to a wider class of problems was proven by Cancès et al. The proof is in the same vein as Le Bris et al. [95]. Consider a weak problem with equivalent energy minimisation problem:

$$u = \arg \min_{v \in V(\Omega)} \mathcal{J}(v). \tag{2.34}$$

Convergence of a pure greedy algorithm associated with this minimisation can be proved using the main result of Cancès et al. (Theorem 2.1 in [37]) provided that the following assumptions on the energy functional are satisfied:

(A1) $\mathcal{J}$ is strongly convex for $\|\cdot\|_V$ so that there exists a constant $\alpha > 0$ such that for $t \in [0,1]$:

$$\mathcal{J}(tu + (1-t)v) \leq t\mathcal{J}(u) + (1-t)\mathcal{J}(v) - \frac{\alpha}{2}t(1-t)\|u-v\|_V^2, \quad \forall u, v \in V.$$

We then say that $\mathcal{J}$ is α-convex [77].

(A2) $\mathcal{J}$ is differentiable and its Fréchet derivative is Lipschitz continuous so that there exists a constant $L \geq 0$ such that

$$\|\mathcal{J}'(u) - \mathcal{J}'(v)\|_V \leq L\|u-v\|_V, \quad \forall u, v \in V,$$

where $\mathcal{J}'$ denotes the Fréchet derivative of $\mathcal{J}$.

Furthermore, we also require the following two assumptions on the functional spaces:

(A3) Span$(\mathcal{S}_1)$ is a dense subset of $(V, \|\cdot\|_V)$.

(A4) $\mathcal{S}_1$ is weakly closed in $(V, \|\cdot\|_V)$.

We now present the second of our proofs, by Falcó and Nouy [66], which adopts a very different approach.

2.4.3 Proof B: Generalised Eckart-Young Approach

We begin by presenting the following Lemma from [66]:

Lemma 3. *$\mathcal{S}_1$ is weakly closed in $(V_\otimes, \|\cdot\|_\otimes)$ and in particular if the norm $\|\cdot\|$ is equivalent to $\|\cdot\|_\otimes$ then $\mathcal{S}_1$ is weakly closed in $(V_\otimes, \|\cdot\|)$ since equivalent norms induce the same weak topology on $V_\otimes$.*

Proof. See [66]. □

From now on we assume the inner product $\langle\cdot,\cdot\rangle$ is such that the associated norm $\|\cdot\|$ is equivalent to $\|\cdot\|_\otimes$ then we define the multivariate mapping $\Pi : z \in V_\otimes \mapsto \Pi(z) \subset \mathcal{S}_1$ called the tensor rank-one projection by:

$$\Pi(z) = \arg \min_{v \in \mathcal{S}_1} \|z - v\|^2. \tag{2.35}$$

Lemma 3 ensures that this mapping is well defined (see [66]). This rank-one projection can be thought of as an abstract form of the energy minimisation (2.34) in Proof A. The pure greedy algorithm (Algorithm 2) is then defined as before but with the *minEnergy* procedure defined by (2.35).

In order to provide a generalisation of the Eckart-Young theorem we first need to introduce generalisations of dominant singular values and dominant singular vectors. We define the dominant singular value, $\sigma : V_\otimes \mapsto \mathbb{R}^+$, by:

$$\sigma(z) = \max_{w \in \mathcal{S}_1 : \|w\|=1} \langle z, w \rangle,$$

and dominant singular vectors, $\mathcal{V} : z \in V_\otimes \mapsto \mathcal{V}(z) \subset \mathcal{S}_1$, by:

$$\mathcal{V}(z) = \{w \in \mathcal{S}_1 : \|w\| = 1, \sigma(z) = \langle z, w \rangle\} = \arg \max_{w \in \mathcal{S}_1 : \|w\|=1} \langle z, w \rangle.$$

We can write the rank-one projector (2.35) in terms on the dominant singular value and vectors by considering:

$$\Pi(z) = \left(\arg\min_{\lambda \in \mathbb{R}} \left(\min_{w \in \mathcal{S}_1 : \|w\|=1} \|z - \lambda w\|^2 \right) \right) \times \left(\arg \min_{w \in \mathcal{S}_1 : \|w\|=1} \left(\min_{\lambda \in \mathbb{R}} \|z - \lambda w\|^2 \right) \right)$$

where

$$\begin{aligned} \arg\min_{\lambda \in \mathbb{R}} \left(\min_{w \in \mathcal{S}_1 : \|w\|=1} \|z - \lambda w\|^2 \right) &= \arg\min_{\lambda \in \mathbb{R}} \left(\min_{w \in \mathcal{S}_1 : \|w\|=1} (\|z\|^2 - 2\lambda \langle z, w \rangle + \lambda^2) \right) \\ &= \arg\min_{\lambda \in \mathbb{R}} \left(\lambda^2 - 2\lambda \max_{w \in \mathcal{S}_1 : \|w\|=1} \langle z, w \rangle \right) \\ &= \max_{w \in \mathcal{S}_1 : \|w\|=1} \langle z, w \rangle = \sigma(z), \end{aligned}$$

by differentiating with respect to λ and equating to zero. Similarly we have:

$$\begin{aligned} \arg \min_{w \in \mathcal{S}_1 : \|w\|=1} \left(\min_{\lambda \in \mathbb{R}} \|z - \lambda w\|^2 \right) &= \arg \min_{w \in \mathcal{S}_1 : \|w\|=1} \left(\min_{\lambda \in \mathbb{R}} (\|z\|^2 - 2\lambda \langle z, w \rangle + \lambda^2) \right) \\ &= \arg \min_{w \in \mathcal{S}_1 : \|w\|=1} \left(- \langle z, w \rangle^2 \right) \\ &= \pm \arg \max_{w \in \mathcal{S}_1 : \|w\|=1} \langle z, w \rangle = \pm \mathcal{V}(z). \end{aligned}$$

Therefore we have:

$$\Pi(z) = \sigma(z)\mathcal{V}(z),$$

where we have taken the positive root since $\sigma(z) \geq 0$.

Furthermore, we introduce two more definitions that feature explicitly in the generalised Eckart-Young theorem: that of the progressive separated representation of an element in $V_\otimes$ and the notion of progressive rank. Given $z \in V_\otimes$ we define the sequence $\{z_n\}_{n\geq 0}$ for $z_n \in \mathcal{S}_n$ by:

$$z_n = \sum_{i=1}^{n} \sigma_i w^{(i)}, \quad \sigma_i = \sigma(z - z_{i-1}), \quad w^{(i)} \in \mathcal{V}(z - z_{i-1}), \tag{2.36}$$

for $n \geq 1$ and with $z_0 = 0$. The element z_n is then called an optimal rank-n progressive separated representation of z with respect to $\|\cdot\|$. The progressive rank ($\mathrm{rank}_{\sigma(z)}$) is then defined by:

$$\mathrm{rank}_{\sigma(z)} = \inf\{n : \sigma(z - z_n) = 0\}$$

We are now in a position to state the generalised Eckart-Young theorem as presented by Falcò and Nouy [66]. In order to put this into context we shall first present the original Eckart-Young theorem as first proved by Carl Eckart and Gale Young in 1936 [61]:

Theorem 3 (Eckart-Young)**.** *Let* $V_\otimes = \mathbb{R}^n \otimes \mathbb{R}^m$ *be endowed with the Frobenius norm* $\|\cdot\|_F$*. For each* $z \in V_\otimes$ *with* $n \leq \mathrm{rank}\, z$*, where* rank *refers to the matrix rank, there exists a nonunique minimizer of*

$$\min_{w\in\mathcal{S}_n} \|z - w\|_F \tag{2.37}$$

given by

$$z_n = \sum_{i=1}^{n} \sigma_i v_i \otimes w_i$$

where $\sigma_i > 0$ *and* $\|v_i \otimes w_i\|_F = 1$ *for* $i = 1, ..., n$*, such that*

$$\|z - z_n\|_F^2 = \|z\|_F^2 - \sum_{i=1}^{n} \sigma_i^2 = \sum_{i=n+1}^{\mathrm{rank}\, z} \sigma_i^2.$$

Remark 3. *Note that the tensor product over the matrix space* $\mathbb{R}^n \otimes \mathbb{R}^m$ *is defined by* $u \otimes v = u \cdot\ v^T$ *which makes its clearer how this Theorem relates to the error in the truncated SVD.*

Theorem 4 (Generalised Eckart-Young)**.** *For* $z \in V_\otimes$*, the sequence* $\{z_n\}_{n\geq 0}$ *given by* (2.36) *satisfies:*

$$z = \lim_{n\to\infty} z_n = z_{\mathrm{rank}_{\sigma(z)}} = \sum_{i=1}^{\mathrm{rank}_{\sigma(z)}} \sigma_i w^{(i)},$$

and

$$\|z - z_n\|^2 = \|z\|^2 - \sum_{i=1}^{n} \sigma_i^2 = \sum_{i=n+1}^{\mathrm{rank}_{\sigma(z)}} \sigma_i^2.$$

Proof. See [66]. □

Remark 4. *Theorem 4 tells us that if z has a finite rank then the algorithm should converge in finitely many steps. This is behaviour that we noted for the rank-*1 *and rank-*2 *solutions in Example 2 in Section 2.2.4. However, this was not observed in the rank-*3 *case which highlights the fact that the actual application of the PGD is not completely equivalent to this idealised greedy algorithm.*

To make it clear how we can use this result to prove convergence of progressive PGD algorithms, consider a weak problem defined on a tensor product Hilbert space $(V_\otimes, \|\cdot\|_\otimes)$ of the form: Find $u \in V_\otimes$ such that

$$\mathcal{A}(u, v) = L(v), \quad \forall v \in V_\otimes. \tag{2.38}$$

The results in [66] tell us that we can apply Theorem 4 to prove convergence of a pure greedy algorithm for this problem provided that $\mathcal{A}(\cdot, \cdot) : V_\otimes \times V_\otimes \to \mathbb{R}$ is a continuous, symmetric and coercive bilinear form. In other words, provided that, for all $u, v \in V_\otimes$:

$$\begin{aligned}
|\mathcal{A}(u, v)| &\leq \alpha \|u\|_\otimes \|v\|_\otimes, \\
\mathcal{A}(u, v) &= \mathcal{A}(v, u), \\
\mathcal{A}(v, v) &\geq \beta \|v\|_\otimes^2,
\end{aligned}$$

for constants $\alpha, \beta > 0$.

We then define the following operator $A : V_\otimes \to V_\otimes$ associated with $\mathcal{A}(\cdot, \cdot)$ by:

$$\mathcal{A}(u, v) = \langle Au, v \rangle_\otimes, \quad \forall u, v \in V_\otimes,$$

and the element $l \in V_\otimes$ associated with L by:

$$L(v) = \langle l, v \rangle_\otimes, \quad \forall u \in V_\otimes.$$

Existence of A and l is guaranteed by the Riesz representation theorem. Hence the weak problem (2.38) can be written in operator form as $Au = l$. From the previous assumptions on $\mathcal{A}(\cdot, \cdot)$ we know that A is bounded, self-adjoint and positive definite. In other words, if we have that, for all $u, v \in V_\otimes$:

$$\begin{aligned}
\|Av\|_\otimes &\leq \alpha \|v\|_\otimes, \\
\langle Au, v \rangle_\otimes &= \langle u, Av \rangle_\otimes, \\
\langle Av, v \rangle_\otimes &\geq \beta \|v\|_\otimes^2,
\end{aligned}$$

then we can define the following inner product and norm induced by the operator A:

$$\langle u, v \rangle_A = \langle Au, v \rangle_\otimes, \qquad \|u\|_A = \sqrt{\langle u, u \rangle_A}.$$

From the properties of A we know that the norm $\|\cdot\|_A$ is equivalent to $\|\cdot\|_\otimes$ and hence by Lemma 3 we have that $\mathcal{S}_1$ is weakly closed in $(V_\otimes, \|\cdot\|_A)$. Therefore we can define a rank-one projector, $\Pi_A(z)$, based on the operator norm $\|\cdot\|_A$ and we can define the associated progressive separated representation:

$$u_n = \sum_{i=1}^{n} u^{(i)}, \quad u^{(i)} \in \Pi_A(u - u_{i-1}), \tag{2.39}$$

where

$$\begin{aligned}
\Pi_A(u - u_{i-1}) =& \arg\min_{v \in \mathcal{S}_1} \|u - u_{i-1} - v\|_A^2 = \arg\min_{v \in \mathcal{S}_1} \mathcal{A}(u - u_{i-1} - v, u - u_{i-1} - v) \\
=& \arg\min_{v \in \mathcal{S}_1} (\mathcal{A}(u - u_{i-1}, u - u_{i-1}) - 2\mathcal{A}(u - u_{i-1}, v) + \mathcal{A}(v, v)) \\
=& \arg\min_{v \in \mathcal{S}_1} \left(\frac{1}{2}\mathcal{A}(v, v) - \mathcal{A}(u, v) + \mathcal{A}(u_{i-1}, v) \right) \\
=& \arg\min_{v \in \mathcal{S}_1} \left(\frac{1}{2}\mathcal{A}(v, v) - L(v) + \mathcal{A}(u_{i-1}, v) \right),
\end{aligned}$$

which is equivalent to the energy minimisation step in Proof A. Therefore the progressive separated representation (2.39) can be thought of as a pure greedy algorithm, convergence of which to the true solution $u = A^{-1}l$ is guaranteed by the generalised Eckart-Young Theorem (Theorem 4).

For the specific case of the Poisson problem considered earlier there is a small issue in applying this proof of convergence. The problem is that this proof is only valid for problems defined in a tensor product Hilbert space, $V_\otimes$. Recall that the definition of a tensor product Hilbert space was given by:

$$V_\otimes = \overline{\operatorname{Span}(\mathcal{S}_1)}^{\|\cdot\|_\otimes}.$$

Therefore, for a general Hilbert space V, $V = V_\otimes$ only if $\|\cdot\|_V$ and $\|\cdot\|_\otimes$ are equivalent norms which will not generally be the case but we do always have the inclusion $V_\otimes \subset V$ [37]. Consider a problem defined on the 2-dimensional Sobolev space $V = H^1(\Omega)$, $\Omega = \Omega_x \times \Omega_y$. For example, the norm on V of a tensor product is given by:

$$\begin{aligned}
\|r \otimes s\|_V^2 =& \int_{\Omega_x} \int_{\Omega_y} (r(x)s(y))^2 + (r'(x)s(y))^2 + (r(x)s'(y))^2 \, dy \, dx \\
=& \|r\|_{L^2(\Omega_x)}^2 \|s\|_{L^2(\Omega_y)}^2 + \|r'\|_{L^2(\Omega_x)}^2 \|s\|_{L^2(\Omega_y)}^2 + \|r\|_{L^2(\Omega_x)}^2 \|s'\|_{L^2(\Omega_y)}^2,
\end{aligned}$$

whereas the associated cross-norm of a tensor product is given by:

$$\begin{aligned}\|r \otimes s\|_{\otimes}^2 =& \|r\|_{H^1(\Omega_x)}^2 \|s\|_{H^1(\Omega_y)}^2 \\ =& \|r\|_{L^2(\Omega_x)}^2 \|s\|_{L^2(\Omega_y)}^2 + \|r'\|_{L^2(\Omega_x)}^2 \|s\|_{L^2(\Omega_y)}^2 \\ &+ \|r\|_{L^2(\Omega_x)}^2 \|s'\|_{L^2(\Omega_y)}^2 + \|r'\|_{L^2(\Omega_x)}^2 \|s'\|_{L^2(\Omega_y)}^2.\end{aligned}$$

Hence $\| \cdot \|_V$ is not equivalent to $\| \cdot \|_{\otimes}$ and so:

$$V = H^1(\Omega) \neq H^1(\Omega_x) \otimes H^1(\Omega_y) = V_{\otimes}.$$

Considering that $V = H^1(\Omega)$ is the natural space one associates with the Poisson problem, then the proof of Falcó and Nouy [66] does not apply in this case. This is surprising since convergence of a progressive PGD algorithm for the Poisson problem was proven by Le Bris et al. [95] (i.e. Proof A in Section 2.4.2). However, it is simple to remedy this by including assumptions (A3) and (A4) used by Cancès et al. [37] which we listed at the end of Section 2.4.2. Indeed, notice that Lemma 3 essentially proves that assumption (A4) holds when $V = V_{\otimes}$. Therefore to extend this theory to cover problems that are not defined in tensor product Hilbert spaces we simply replace Lemma 3 by assumption (A4).

The proof of Falcó and Nouy [66] unfolds as before, but now for problems defined in more general Hilbert spaces, until the point that it claims that the sequence defining the progressive separated representation (2.39) will converge to the solution $u = A^{-1}l$. This is certainly true when $V = V_{\otimes}$ but if we were in the situation where $V \neq V_{\otimes}$ and the solution $u \in V \backslash V_{\otimes}$ then we cannot guarantee this sequence will converge to the solution. All that we can guarantee is that it would converge to the element of the closure of Span($\mathcal{S}_1$) under the norm induced by the operator A, $\| \cdot \|_A$, which minimises the residual in the same norm. Under the assumptions that $\mathcal{A}(\cdot, \cdot)$ defines a continuous, symmetric and coercive bilinear form then $\| \cdot \|_A$ is equivalent to $\| \cdot \|_V$ Hence if we include assumption (A3) then the density of Span($\mathcal{S}_1$) in V ensures that the sequence converges to the true solution.

In the case of the Poisson problem we have

$$\mathcal{A}(u, v) = \int_{\Omega} \nabla u \cdot \nabla v \, d\Omega,$$

which is well known to be continuous, symmetric and coercive in $H_0^1(\Omega)$. Furthermore, the assumptions (A3) and (A4) have been shown to be satisfied when $V = H^1(\Omega)$ in the context of a high-dimensional Poisson equation by Cancès et al. [37] and hence an associated pure greedy algorithm (i.e. the progressive separated representation (2.39)) can be proven to converge for the Poisson equation by Theorem 4.

2.4.4 Rates of Convergence

In order to gain some understanding into the convergence rate of the pure greedy algorithm we refer to some early results on greedy algorithms given by DeVore and Temylakov [57]. Firstly, we need to define the following functional spaces for all $M > 0$:

$$\mathcal{L}_1^o(M) := \left\{ u \in V \ : \ u = \sum_{k=0}^{K} c_k w_k, \ w_k \in \mathcal{S}_1, \ \|w_k\|_V = 1, \ K < \infty, \ \sum_{k=0}^{K} |c_k| \leq M \right\},$$

we then define the following space:

$$\mathcal{L}_1 = \bigcup_{M>0} \overline{\mathcal{L}_1^o(M)},$$

with norm:

$$\|u\|_{\mathcal{L}_1} = \inf\{M > 0 \ : \ u \in \overline{\mathcal{L}_1^o(M)}\},$$

for $u \in \mathcal{L}_1$. DeVore and Temylakov [57] then proved that the following estimate holds for the pure greedy algorithm: For $u \in \mathcal{L}_1$:

$$\|u_n\|_V \leq \|u\|_{\mathcal{L}_1} n^{-1/6},$$

which was later very slightly improved by Konyagin and Temlyakov [88] to

$$\|u_n\|_V \leq \|u\|_{\mathcal{L}_1} n^{-11/62},$$

where the sequence u_n is defined as in Proof A (2.31). Unfortunately, both these estimates require that the true solution $u \in \mathcal{L}_1$. The problem with this is that it is not clear how to characterise elements of $\mathcal{L}_1$ and furthermore Le Bris et al. [95] showed that this requirement becomes even more restrictive in higher-dimensions where it appears that increased regularity of the solution is required.

On the other hand, Cancès et al. [37] showed that the pure greedy algorithm for problems defined in finite dimensional space converge exponentially. This result might seem more relevant to the progressive PGD since we apply a discretisation and hence solve a finite dimensional problem in the discretisation space V^h where h denotes the mesh width. Indeed, our convergence results in the rank for the Poisson problem (Figures 2.5, 2.7 and 2.13) appear to display exponential rates of convergence up to the stagnation due to error in the discretisation. Unfortunately, the result of Cancès et al. [37] is still not completely relevant to our application of the progressive PGD. What we really require is an estimate which is based on the mesh width, h, as well as the rank of the approximation. Unfortunately, it is unclear how and if it is possible to derive such an estimate and this is still an open problem.

2.4.5 Concluding Remarks

In this section we have placed the progressive PGD in a theoretical context by treating the separated representation as a tensor decomposition and by treating the progressive PGD itself as a pure greedy algorithm. Convergence of this pure greedy algorithm was proven using two distinct proofs A and B.

Proof A, which was based on energy minimisation, provided a clear comparison with the actual implementation of the progressive PGD. In particular, part 1 of the proof of Theorem 2 proves that the algorithm converges when we only assume that the rank-one tensors selected at each iteration satisfy the Euler-Lagrange equations (i.e. Galerkin orthogonality) which is how the PGD is applied in practice. Unfortunately, one can only proceed and say that it converges to the solution if we assume that the rank-one tensors also solve the energy minimisation problem.

On the other hand, Proof B gave a much more abstract approach to proving convergence of pure greedy algorithms. Indeed, it is not initially clear how this method can be applied to the solution of PDEs and technically some assumptions from Cancès et al. [37] need to be included in order to prove convergence in spaces which are not tensor product Hilbert spaces. However, the desirable aspect of this proof is that it draws a very interesting comparison with the famous result of Eckart and Young [61] for the error in the truncated SVD.

There are still a number of questions that need to be answered about convergence of progressive PGDs. Firstly, as we noted in Section 2.4.4, the proofs of convergence (and their rate of convergence) do not take into account the discretisation we need to apply in order to compute the PGD. It also does not take into account the alternating directions linearisation we need to employ. Furthermore, as we have already explained in Section 2.4.1, the pure greedy algorithm is not actually equivalent to the application of the progressive PGD in practice and it is unclear how or if the proof can be extended to cover this. Finally, the proofs only hold under certain assumptions on the original problem. Therefore it is completely unclear how the progressive PGD should behave for a problem which cannot be expressed as a minimisation of some energy functional.

It is this last point that we want to investigate next. So far we have only considered the progressive PGD applied to the Poisson equation for which Proofs A and B are both applicable. We will now investigate the Stokes problem which is a weakly coercive problem and hence cannot be expressed as the minimisation of some energy functional. This means that we cannot even define a pure greedy algorithm for this problem.

2.5 The Stokes Problem

The Stokes problem is the linear, steady version of the Navier-Stokes equations [126] which is the governing equation in fluid dynamics. In particular, the Stokes problem can be derived from the Navier-Stokes equations as the limit in the Reynolds number approaches zero. The Reynolds number is the ratio of inertial forces to viscous forces and hence a zero Reynolds number can be associated with extremely viscous flows. The Stokes problem is then an appropriate model for flows in materials such as lava or paint.

We begin by introducing the so called primal Stokes problem (e.g. see Brezzi and Fortin [31]) which is of the form of the following constrained optimisation problem: Find $\mathbf{u} \in (H_0^1(\Omega))^d$, $(d = 2, 3)$ such that:

$$\mathbf{u} = \arg \min_{\substack{\mathbf{v} \in (H_0^1(\Omega))^d \\ \nabla \cdot \mathbf{v} = 0}} \left(\frac{1}{2} \int_\Omega |\nabla \mathbf{v}|^2 \, d\Omega - \int_\Omega \mathbf{f} \cdot \mathbf{v} \, d\Omega \right), \tag{2.40}$$

where we have assumed we have homogeneous Dirichlet boundary conditions $\mathbf{u} = \mathbf{0}$ on $\partial\Omega$. Here $\mathbf{u}$ denotes the vector of velocity components of the flow and $\mathbf{f}$ denotes some known source term. This problem should look familiar as it is essentially a vector Dirichlet energy (similar to the scalar Dirichlet energy for the Poisson equation (2.27)) except with the additional constraint that $\nabla \cdot \mathbf{u} = 0$ which represents the assumption of incompressibility of the fluid. Theoretically we could define an associated pure greedy algorithm to this problem whereby the dictionary we use is no longer just the set of all rank-one tensors, $\mathcal{S}_1$, but the set of all rank-one tensors which satisfy incompressibility. Unfortunately, such a dictionary would be very difficult to construct and would not lead to a practical setting in which a progressive PGD algorithm could be applied.

Constrained optimisation problems such as (2.40) are most commonly solved by the introduction of a Lagrangian multiplier. Following Brezzi and Fortin [31] we define the following characteristic function:

$$\delta(x) := \begin{cases} 0, & \text{if } x = 0, \\ \infty, & \text{otherwise.} \end{cases}$$

We can then write the constrained optimisation problem (2.40) as:

$$\mathbf{u} = \arg \min_{\mathbf{v} \in (H_0^1(\Omega))^d} \left(\frac{1}{2} \int_\Omega |\nabla \mathbf{v}|^2 \, d\Omega - \int_\Omega \mathbf{f} \cdot \mathbf{v} \, d\Omega + \delta(\nabla \cdot \mathbf{v}) \right),$$

since minimising $\delta(\nabla \cdot \mathbf{v})$ enforces the incompressibility constraint $\nabla \cdot \mathbf{u} = 0$ due to the definition of the characteristic function δ. While we do now have an unconstrained optimisation problem this definition of the characteristic function is not at all practical. Instead, we express the characteristic function in the following way:

$$\delta(\nabla\cdot\mathbf{v}) = \sup_{q\in L^2(\Omega)}\left(-\int_\Omega q\nabla\cdot\mathbf{v}\;d\Omega\right),$$

which leads to the following saddle-point problem: Find $(\mathbf{u},p)\in(H_0^1(\Omega))^d\times L^2(\Omega)$ such that:

$$(\mathbf{u},p) = \arg\min_{\mathbf{v}\in(H_0^1(\Omega))^d}\max_{q\in L^2(\Omega)}\left(\frac{1}{2}\int_\Omega|\nabla\mathbf{v}|^2\;d\Omega - \int_\Omega\mathbf{f}\cdot\mathbf{v}\;d\Omega - \int_\Omega q\nabla\cdot\mathbf{v}\;d\Omega\right),$$

where we have substituted the supremum for the maximum since from the existence result for the Stokes problem (e.g. [126]) we know there exists a unique solution $p\in L^2(\Omega)$ (up to an additive constant). Note that the Lagrangian multiplier, p, can be physically interpreted as the pressure of the fluid.

One could attempt to define a greedy algorithm which iteratively seeks the rank-one solution which satisfies this saddle point problem but this type of algorithm is not covered by the theory developed by Temylakov [127] and hence the convergence of such an algorithm is unclear. In this section we develop a progressive PGD algorithm regardless of this point to investigate the convergence behaviour. To this end we find the Euler-Lagrange equations associated with the above saddle-point problem which are given by: Find $(\mathbf{u},p)\in(H_0^1(\Omega))^d\times L^2(\Omega)$ such that:

$$\int_\Omega\nabla\mathbf{u}:\nabla\mathbf{u}^*\;d\Omega - \int_\Omega p\nabla\cdot\mathbf{u}^*\;d\Omega = \int_\Omega\mathbf{f}\cdot\mathbf{u}^*\;d\Omega,\quad\forall\mathbf{u}^*\in(H_0^1(\Omega))^d,\tag{2.41}$$

$$\int_\Omega p^*\nabla\cdot\mathbf{u}\;d\Omega = 0,\quad\forall p^*\in L^2(\Omega),\tag{2.42}$$

which is exactly the weak formulation of the classic form of the Stokes problem:

$$-\nabla^2\mathbf{u}+\nabla p = \mathbf{f}\quad\text{in }\ \Omega,\tag{2.43}$$

$$\nabla\cdot\mathbf{u} = 0\quad\text{in }\ \Omega,\tag{2.44}$$

$$\mathbf{u} = \mathbf{0}\quad\text{on }\ \partial\Omega.\tag{2.45}$$

2.5.1 Zero Mean Pressure

We previously stated that the pressure is only unique up to an additive constant. For this reason we need to apply a constraint on the pressure in order to ensure uniqueness. This is done by enforcing zero mean pressure:

$$\int_\Omega p\;d\Omega = 0.$$

One of the most common ways of imposing this is by the method of setting the pressure datum (see e.g. Yeckel and Derby [130]) which amount to removing one degree of freedom from the pressure space. This can be thought of as fixing the pressure at a specific point which ensures it is uniquely defined. Unfortunately, in the PGD it is not possible to fix a value at a specific point due to the separated representation of the unknown field. In other words the degrees of freedom in the PGD do not represent points in space. An alternative method is to impose the zero mean pressure implicitly using the method used by Gwynllyw and Phillips [76], for example. This method considers the following alternative statement of the Stokes problem (2.43)-(2.45):

$$-\nabla^2 \mathbf{u} + \nabla p = \mathbf{f} \qquad \text{in } \Omega, \tag{2.46}$$

$$-\nabla \cdot \mathbf{u} = \mu \int_\Omega p \, d\Omega \qquad \text{in } \Omega, \tag{2.47}$$

$$\mathbf{u} = \mathbf{0} \qquad \text{on } \partial\Omega, \tag{2.48}$$

for some adjustable parameter $\mu > 0$. We can see how this includes the zero mean pressure implicitly by integrating (2.47) over Ω yielding:

$$-\int_\Omega \nabla \cdot \mathbf{u} \, d\Omega = \mu \cdot \text{meas}(\Omega) \int_\Omega p \, d\Omega. \tag{2.49}$$

Applying the Divergence theorem to the LHS we see that

$$\int_\Omega \nabla \cdot \mathbf{u} \, d\Omega = \int_{\partial\Omega} \mathbf{u} \cdot \mathbf{n} \, d\partial\Omega = 0$$

since $\mathbf{u} = \mathbf{0}$ on $\partial\Omega$. Then since we have that $\mu > 0$ and meas$(\Omega) > 0$ in (2.49) we must have that

$$\int_\Omega p \, d\Omega = 0.$$

Therefore the zero mean pressure constraint is imposed implicitly in this system. We can then derive the weak formulation of this alternative Stokes problem by taking the dot product of (2.46) with $\mathbf{u}^* \in (H_0^1(\Omega))^d$, integrating over Ω, and applying Green's first integral identity yielding:

$$\int_\Omega \nabla \mathbf{u} : \nabla \mathbf{u}^* \, d\Omega - \int_\Omega p(\nabla \cdot \mathbf{u}^*) \, d\Omega = \int_\Omega \mathbf{f} \cdot \mathbf{u}^* \, d\Omega$$

Similarly, for the pressure, we multiply (2.44) through by $p^* \in L^2(\Omega)$ and integrate over Ω to obtain

$$-\int_\Omega p^*(\nabla \cdot \mathbf{u}) \, d\Omega + \mu \int_\Omega p \, d\Omega \int_\Omega p^* \, d\Omega = 0.$$

If we now define the following bilinear forms

$$a(\mathbf{u}, \mathbf{u}^*) = \int_\Omega \nabla \mathbf{u} : \nabla \mathbf{u}^* \ d\Omega, \quad b(\mathbf{u}^*, p) = -\int_\Omega p(\nabla \cdot \mathbf{u}^*) d\Omega,$$

$$c(p, p^*) = \mu \int_\Omega p \ d\Omega \int_\Omega p^* \ d\Omega,$$

and the linear functional

$$l(\mathbf{u}^*) = \int_\Omega \mathbf{f} \cdot \mathbf{u}^* \ d\Omega.$$

Then the weak form of the Stokes problem with homogeneous boundary conditions may be written: Find $(\mathbf{u}, p) \in (H_0^1(\Omega))^d \times L^2(\Omega)$ such that:

$$a(\mathbf{u}, \mathbf{u}^*) + b(\mathbf{u}^*, p) = l(\mathbf{u}^*) \tag{2.50}$$

$$b(\mathbf{u}, p^*) + c(p, p^*) = 0 \qquad \forall (\mathbf{u}^*, p^*) \in (H_0^1(\Omega))^d \times L^2(\Omega), \tag{2.51}$$

which is exactly the weak formulation (2.41)-(2.42) derived as the Euler-Lagrange equations of the saddle-point problem with the additional term, $c(p, p^*)$, associated with the implicit imposition of the zero mean pressure constraint.

Note that, upon discretisation, the additional term, $c(p, p^*)$, will contribute a full matrix to the part of the linear system associated with the pressure. This can make finite/spectral element methods cumbersome when employing this method. However, it has been shown (see e.g. [130]) that when using iterative solvers such as GMRES then a solution to the Stokes problem can be found without imposing the zero mean pressure constraint and we can obtain the desired value of the pressure by adding/subtracting a suitable constant from the solution obtained. Considering that the progressive PGD algorithm can be thought of as an iterative method it may be the case that we do not need to impose zero mean pressure at all. This is something we shall investigate in our numerical experiments.

2.5.2 The LBB Condition

Before developing a progressive PGD algorithm for approximating the solution to the Stokes problem we need to address an issue which can effect the stability of numerical solutions. In order to explain this we begin by expressing the weak Stokes problem (2.50)-(2.51) as an equation involving a single bilinear form $Q : U \times U \mapsto \mathbb{R}$: Find $\{\mathbf{u}, p\} \in U$ such that:

$$Q(\{\mathbf{u}, p\}, \{\mathbf{u}^*, p^*\}) = F(\{\mathbf{u}^*, p^*\}), \quad \forall \{\mathbf{u}^*, p^*\} \in U,$$

where $U = (H_0^1(\Omega))^d \times L^2(\Omega)$ and

$$Q(\{\mathbf{u}, p\}, \{\mathbf{u}^*, p^*\}) := a(\mathbf{u}, \mathbf{u}^*) + b(\mathbf{u}, p^*) + b(\mathbf{u}^*, p), \qquad F(\{\mathbf{u}^*, p^*\}) := l(\mathbf{u}^*),$$

where for the time being we leave out the bilinear form, $c(\cdot,\cdot)$, associated with the implicit imposition of the zero mean pressure condition. It can be shown that this bilinear form is continuous, i.e. for all $u, v \in U$ we have:

$$|Q(u,v)| \leq \alpha \|u\|_U \|v\|_U,$$

for some constant $\alpha > 0$. However, $Q(\cdot,\cdot)$ is not (strongly) coercive but instead satisfies weak coercivity:

$$\inf_{v \in U} \sup_{u \in U} \frac{Q(u,v)}{\|u\|_U \|v\|_U} \geq \beta, \qquad \inf_{u \in U} \sup_{v \in U} \frac{Q(u,v)}{\|u\|_U \|v\|_U} \geq \gamma, \tag{2.52}$$

for some constants $\beta, \gamma > 0$. Note that $Q(\cdot,\cdot)$ is also symmetric:

$$Q(u,v) = Q(v,u), \qquad \forall u, v \in U,$$

and hence each of the inequalities in (2.52) implies the other. Note that it is the weak coercivity of this problem which prevents us from proving convergence of an associated progressive PGD algorithm. Indeed, recall that the required assumptions on the bilinear form in the generalised Eckart-Young approach in Section 2.4.3 was that it needed to be continuous, symmetric and strongly coercive and it is only the last point that the Stokes problem fails due to its weak coercivity.

Well-posedness of continuous, strongly coercive problems can be guaranteed by the famous Lax-Milgram Theorem [94]. For continuous, weakly coercive problems, such as the Stokes problem, this result can be generalised by the Babuška-Lax-Milgram Theorem [12]. The Brezzi Theorem [30] also provides well-posedness for the particular example of saddle-point problems, again such as the Stokes problem. This Theorem gives the assumptions not in terms of the bilinear form, $Q(\cdot,\cdot)$, but in terms of the original bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$. In the particular case of the Stokes problem, the Brezzi Theorem states that the problem (2.50)-(2.51) is well posed provided that $a(\cdot,\cdot)$ is weakly coercive on the null space of the operator induced by $b(\cdot,\cdot)$:

$$\inf_{\mathbf{u} \in Z} \sup_{\mathbf{v} \in Z} \frac{a(\mathbf{u},\mathbf{v})}{\|\mathbf{u}\|_{H^1} \|\mathbf{v}\|_{H^1}} \geq \beta_a, \tag{2.53}$$

for some constant $\beta_a > 0$, where:

$$Z = \{\mathbf{u} \in (H_0^1(\Omega))^d \mid b(\mathbf{u},p) = 0, \quad \forall p \in L^2(\Omega)\},$$

and provided that $b(\cdot,\cdot)$ satisfies the inf-sup condition:

$$\inf_{p \in L^2(\Omega)} \sup_{\mathbf{u} \in (H_0^1(\Omega))^d} \frac{b(\mathbf{u},p)}{\|\mathbf{u}\|_{H^1} \|p\|_{L^2}} \geq \beta_b, \tag{2.54}$$

for some constant $\beta_b > 0$. For the Stokes problem, (2.53) is satisfied since $a(\cdot,\cdot)$ is in fact strongly coercive on Z. The inf-sup condition (2.54) was also proven to hold for the Stokes problem by Ladyzhenskaya [92].

Consider a strongly coercive problem: Find $u \in W$ such that:

$$C(u, v) = G(v), \quad \forall v \in W,$$

where the bilinear form $C(\cdot, \cdot)$ satisfies:

$$C(u, u) \geq \beta \|u\|_W^2, \quad \forall u \in W,$$

for some constant $\beta > 0$. The discrete problem is then given by: Find $u^h \in W^h$

$$C(u^h, v^h) = G(v^h), \quad \forall v^h \in W^h,$$

for some conforming discretisation subspace $V^h \subset V$, where h denotes the mesh width of the discretisation. In this case, the strong coercivity is inherited by the discretisation subspace and hence we have the inequality:

$$C(u^h, u^h) \geq \beta^h \|u^h\|_{W^h}^2, \quad \forall u^h \in W^h,$$

for some constant $\beta^h > 0$. However, when you only have weak coercivity, the discrete analogues of the inequalities (2.52) will not, in general, hold even when using conforming discretisation subspaces. Therefore these inequalities need to be included as additional assumptions to ensure the discrete problem is well defined. In the case of the Stokes problem we need to include the additional assumption that the discrete analogue of the inf-sup condition (2.54) holds:

$$\inf_{p^h \in Q^h} \sup_{\mathbf{u}^h \in V^h} \frac{b(\mathbf{u}^h, p^h)}{\|\mathbf{u}^h\|_{V^h} \|p^h\|_{Q^h}} \geq \beta_b^h, \tag{2.55}$$

for some constant $\beta_b^h > 0$ and conforming discretisation subspaces $V^h \subset (H_0^1(\Omega))^d$ and $Q^h \subset L^2(\Omega)$. The additional assumption (2.55) is what is known as the LBB condition (named after Ladyzhenskaya, Babuška and Brezzi for their individual contributions on this topic [12, 30, 92]). If one is not careful with the selection of the discretisation subspaces V^h and Q^h then this can lead to an ill-posed discrete problem which may lead to spurious oscillations in the solution which are known as LBB stability issues. The LBB condition has been well studied and a number of ways of carefully selecting the discretisation subspaces V^h and Q^h has been derived. For example, when using a spectral element discretisation one can employ the $P_N - P_{N-2}$ method of Maday et al. [101] in which the pressure space involves polynomial basis functions which are two degrees lower than those in the velocity space. This method then ensures the LBB condition (2.55) is satisfied and hence we have LBB stability of our solution.

However, in the case of the progressive PGD the issue does not end here. Recall that we do not seek solutions $(\mathbf{u}, p) \in (H_0^1(\Omega))^d \times L^2(\Omega)$ but instead iteratively seek rank-one solutions $(\mathbf{u}, p) \in \mathcal{S}_1^{\mathbf{u}} \times \mathcal{S}_1^p$ where:

$$\mathcal{S}_1^{\mathbf{u}} = \left\{ \bigotimes_{i=1}^d \mathbf{u}_i \mid (\mathbf{u}_1, \ldots, \mathbf{u}_d) \in \prod_{i=1}^d (H_0^1(\Omega_i))^d \right\},$$

and

$$\mathcal{S}_1^p = \left\{ \bigotimes_{i=1}^d p_i \mid (p_1, \ldots, p_d) \in \prod_{i=1}^d L^2(\Omega_i) \right\},$$

where $\Omega = \prod_{i=1}^d \Omega_i$. Therefore we also need to satisfy the inf-sup condition (2.54) over these nonlinear manifolds of rank-one tensors (and furthermore discretisation subsets thereof), i.e. we require that:

$$\inf_{p \in \mathcal{S}_1^p} \sup_{\mathbf{u} \in \mathcal{S}_1^{\mathbf{u}}} \frac{b(\mathbf{u}, p)}{\|\mathbf{u}\|_{H^1} \|p\|_{L^2}} \geq \beta_b^*, \tag{2.56}$$

for some constant $\beta_b^* > 0$. Again, we cannot guarantee that this inequality will be inherited from the continuous full-rank inf-sup condition (2.54) but, unlike the LBB condition (2.55), it is not at all clear how to verify or ensure that this condition is satisfied. This potentially means that Galerkin PGD algorithms for the Stokes problem may experience unpredictable LBB-like stability issues.

So far we have seen that the theoretical setting for the Galerkin PGD applied to the Stokes problem is not particularly robust. Not only can we not prove convergence of progressive Galerkin PGD algorithms due to the saddle-point/weakly coercive nature of the problem but we also cannot guarantee stability of approximations. However, we will continue to formulate PGD algorithms for the Stokes problem and perform numerical experiments in order to try to establish whether these issues appear to have any detrimental effects on the algorithm.

2.5.3 PGD Formulation

Consider the weak Stokes problem (2.50)-(2.51) in 2D defined on a square domain $\Omega = [a, b] \times [c, d]$, where $\mathbf{u} = (u, v)^T$. We seek rank-J separated representations of the three scalar dependent variables (u, v, p):

$$u(x, y) \approx \sum_{j=1}^J X_j^u(x) Y_j^u(y) =: u_J(x, y), \qquad v(x, y) \approx \sum_{j=1}^J X_j^v(x) Y_j^v(y) =: v_J(x, y),$$

$$p(x, y) \approx \sum_{j=1}^J X_j^p(x) Y_j^p(y) =: p_J(x, y)$$

The PGD modes for the velocity components $(X_j^u(x), Y_j^u(y), X_j^v(x), Y_j^v(y))$ are discretised using a Legendre spectral element method in a completely analogous way to the Poisson equation (2.8)-(2.9), where once again the homogeneous Dirichlet boundary conditions are included explicitly in each of the PGD modes. The functions $X_j^p(x)$ and $Y_j^p(y)$ are also discretised using a Legendre spectral element method but this time, following the $P_N - P_{N-2}$ method of Maday et al. [101], we employ degree $N-2$ polynomial basis functions for the pressure given by

$$X_j^p(x) = \begin{cases} \sum_{i=1}^{N-1} \alpha_{j,i,k}^p \tilde{h}_{i,k}(x), & x \in [a_{k-1}, a_k], \\ 0, & \text{otherwise,} \end{cases}$$

$$Y_j^p(y) = \begin{cases} \sum_{i=1}^{N-1} \beta_{j,i,k}^p \tilde{h}_{i,k}(y), & y \in [c_{k-1}, c_k] \\ 0, & \text{otherwise.} \end{cases}$$

where $\tilde{h}_{i,k}(x)$ $i = 1, \dots, N-1$, $k = 1, \dots, K_x$ is chosen to be the interior Legendre interpolating polynomial on the k^{th} element:

$$\tilde{h}_{i,k}(x) = \begin{cases} \tilde{h}_i(\xi_k(x)) := \dfrac{(1-x_i^2)P_N'(\xi_k(x))}{N(N+1)P_N(x_i)(x_i - \xi_k(x))}, & x \in [a_{k-1}, a_k], \\ \\ 0, & \text{otherwise,} \end{cases}$$

where $\xi_k(x)$, $k = 1, \dots, K_x$ are the maps defined by (2.10). We also have analogous definitions for $\tilde{h}_{i,k}(y)$, $i = 1, \dots, N-1$, $k = 1, \dots, K_y$. These basis functions interpolate the PGD modes at the interior Gauss-Lobatto points so that we have

$$\alpha_{j,i}^p = X_j^p(x_i), \qquad \beta_{j,i}^p = Y_j^p(y_i),$$

for $j = 1, \dots, J$ and $i = 1, \dots, N-1$. Note that, even though we cannot guaranteed LBB-like stability in the PGD, we still employ this $P_N - P_{N-2}$ method since it is still desirable to use discretisation subspaces which satisfy the LBB condition for the full-rank problem.

The progressive PGD algorithm for the Stokes problem can then be described as the following iterative procedure in each of the scalar dependent variables:

$$\begin{aligned} u_J(x,y) =& u_{J-1}^{(e)}(x,y), \qquad u_0(x,y) = 0, \\ v_J(x,y) =& v_{J-1}^{(e)}(x,y), \qquad v_0(x,y) = 0, \\ p_J(x,y) =& p_{J-1}^{(e)}(x,y), \qquad p_0(x,y) = 0. \end{aligned}$$

where $u_{J-1}^{(e)}(x,y)$ denotes the enriched solution as described in Section 2.1, and similarly for the other dependent variables. The resulting nonlinear systems are once again solved using the ADFPA (Algorithm 1).

2.5.4 Numerical Results

Example 4 (Infinite Rank Pressure Solution)**.**

Consider the Stokes problem (2.46)-(2.48) with source term:

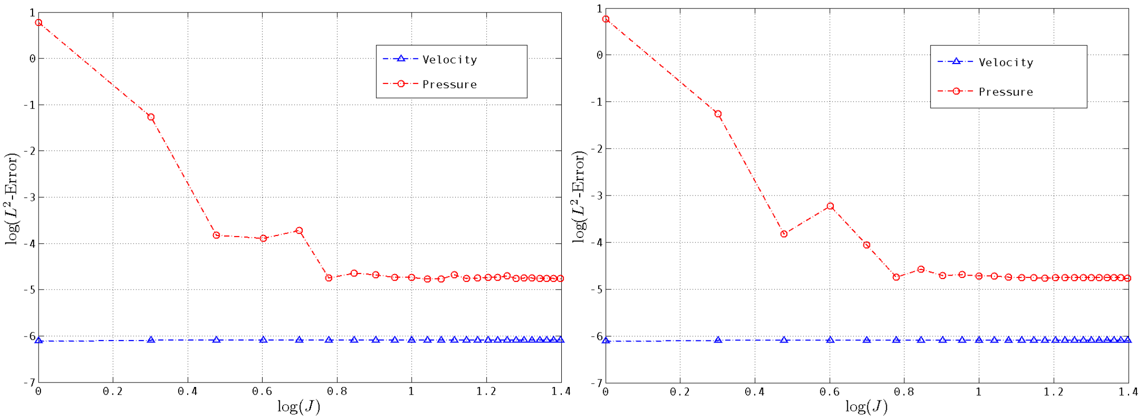
$$\mathbf{f}(x,y) = \begin{pmatrix} \pi y\cos(\pi xy) + 4\pi^2\sin(2\pi y)(2\cos(2\pi x) - 1) \\ \pi x\cos(\pi xy) - 4\pi^2\sin(2\pi x)(2\cos(2\pi y) - 1) \end{pmatrix},$$

This Stokes problem has the following exact solution:

$$\mathbf{u} = \begin{pmatrix} -\sin(2\pi y)(\cos(2\pi x) - 1) \\ \sin(2\pi x)(\cos(2\pi y) - 1) \end{pmatrix},$$

$$p = \sin(\pi xy).$$

Note that the velocity components both possess a natural rank-one separated representation whereas the the pressure possesses and infinite rank separated representation.



(a) Implicit Zero Mean Pressure ($\mu = 1$) (b) No Implicit Zero Mean Pressure ($\mu = 0$)

Figure 2.14: Convergence in the Rank for Example 4

Figure 2.14 shows the convergence with increasing rank of the PGD approximation both with and without the zero mean pressure included implicitly. It is clear from this that there is no significant difference in the levels of convergence obtained in either case and for this reason we believe it is better to not include zero mean pressure implicitly as it increases the complexity of the linear systems by introducing a full matrix into the pressure system.

We also note that the rank-one separated representation of the velocity has successfully been captured by this algorithm despite the coupling of the velocity with an infinite-rank pressure solution. We also find that the error in the pressure rapidly decreases until it reaches stagnation due to the discretisation error. Note that the pressure does not reach a comparable level of convergence to the velocity since the approximation space for this involves polynomial basis functions which are two degrees lower. We do however notice in both cases that the convergence in the pressure is not monotonic. In the results for the Poisson equation we also observed monotonic convergence. This behaviour can most likely be attributed to the fact that monotonic convergence cannot be proven in this case because the Stokes problem is only weakly coercive. Despite this the overall convergence rate is very promising.

So far this progressive Galerkin PGD algorithm for the Stokes problem appears to work perfectly well. Unfortunately, this is not the case. The results in Figure 2.14 were run for a choice of discretisation using degree $N = 15$ polynomial basis functions for the velocity and $N = 13$ for the pressure on a single element ($K_x = K_y = 1$) so essentially this is a high-order spectral method rather than a spectral element method. The reason for this choice of discretisation is that is was one of very few choices that worked for both $\mu = 0$ and $\mu = 1$. What we mean by this is that for the majority of choices of N, K_x and K_y we either experienced singular behaviour of the linear systems or the ADFPA (Algorithm 1) failed to converge. This is a significant problem since we do not have any indication a priori which choices of discretisation will yield a working algorithm. This would be a serious disadvantage for more complex fully three dimensional problems where we cannot rely on trial and error in order to try to obtain a working solution. It also does not rule out the possibility that certain problems may have no choice of discretisation which yields a working algorithm.

In order to try and minimise the possibility of these issues being caused by ill-conditioning of the linear systems propagating error throughout the ADFPA we employed a minimal residual (minres) iterative solver with a standard block preconditioner for the Stokes problem (see e.g. Elman et al. [63]). This technique has been shown to be particularly effective for the solution of the Stokes problem. However, this did not lead to any improvement in the behaviour of the PGD algorithm. Instead, we speculate that this could be related to a lack of LBB-like stability whereby these choices of discretisation fail to satisfy the discrete analogue of the rank-one tensor inf-sup condition (2.56). Overall it seems that the progressive Galerkin PGD algorithm for the Stokes problem is an unreliable and inefficient algorithm.

2.6 Conclusions

In this chapter we have reviewed and investigated the progressive Galerkin PGD algorithm. We considered the simple case of the Poisson equation and demonstrated the effectiveness of the progressive Galerkin PGD algorithm for this problem. We also gave some results related to the use of a spectral element discretisation in the PGD which has not been previously widely considered. We further reviewed some techniques for applying the PGD to problems with non-homogeneous Dirichlet boundary conditions by using transfinite interpolation. A transfinite interpolating function was then provided for a problem defined on a general d-orthotope. The convergence of the progressive PGD algorithm was then reviewed by considering two distinct proofs. In both cases we were able to verify convergence of the progressive Galerkin PGD algorithm applied to the Poisson equation and more generally to continuous, symmetric and strongly coercive problems. Finally, we considered the weakly coercive Stokes problem for which these proofs of convergence no longer hold. Furthermore, we showed that we can longer guarantee LBB-like stability of our PGD approximations due to the need to satisfy the rank-one tensor inf-sup condition (2.56). Numerical results were able to yield good rates of convergence but only for a select choice of discretisation parameters; a problem which we believe can be attributed to the lack of LBB-like stability.

In the next Chapter we will instead investigate the progressive least-squares PGD algorithm. The big advantage of a least-squares formulation is that it provides one with a continuous, symmetric and strongly coercive problem for any elliptic problem. In the case of the Stokes problem this means we could now prove convergence of a progressive least-squares PGD algorithm and furthermore there is no longer any stability conditions (such as LBB) since this was an artifact of the weakly coercive problem. In this sense the progressive least-squares PGD algorithm can be thought of as the most theoretically sound setting for the PGD. This in itself warrants further investigation into this method.

Chapter 3

Least-Squares Proper Generalised Decompositions

3.1 Introduction

In the previous chapter we considered a Galerkin PGD algorithm for the solution to the Stokes problem. Unfortunately this algorithm had two major pitfalls: Firstly we could not guarantee LBB-like stablity of the algorithm. Indeed, we found that for certain choices of the discretisation parameters, the algorithm failed to converge in the linearisation iteration. Secondly, we were unable to prove convergence of the algorithm due to it being only weakly coercive, or equivalently the issue being that the Stokes problem possesses a Rayleigh-Ritz setting (i.e. (2.25)) which is a constrained optimisation problem (2.40). This is not a practical setting for the implementation of numerical approximations and hence the constraint was imposed by introducing pressure into the system in the form of a Lagrangian multiplier. This comes at the cost of sacrificing the Rayleigh-Ritz setting for the problem as we now obtain a saddle-point problem. This meant that it was no longer possible to define a greedy algorithm for this problem that we are able to prove convergence of and ultimately changing the problem in this ways leads to the need to satisfy stability conditions such as the LBB condition.

In this chapter we consider progressive PGD algorithms based on least-squares formulations rather than Galerkin formulations. The main idea of least-squares methods is to minimise the residual of a given differential equation in a certain norm. The choice of norm must be carefully selected. This then introduces an 'artificial' energy in the form of the so called quadratic least-squares functional which supplies us with an (unconstrained) Rayleigh-Ritz type setting for the problem even when a Galerkin formulation of the same problem may not have one (e.g the Stokes problem). This provides us with a platform to build a proof of convergence of the progressive PGD for a much larger class of problems. It also has the added benefit that we no longer need to satisfy stability conditions such as the LBB condition. PGD algorithms based on least-squares formulations have been considered by Nouy [106] under the name minimal residual PGDs. We will use the terminology 'least-squares PGDs' in order to highlight the fact that we construct PGD algorithms based on rigorous continuous least-squares principles rather than simply minimising the residual in some norm. Nouy observed that minimal residual PGDs can suffer from slow convergence rates. This is an issue we will investigate further in this chapter.

Least-squares formulations were first studied in the early 1970s by Bramble and Schatz [28, 29]. In these papers they sought a Rayleigh-Ritz type setting for problems where auxiliary conditions, such as the boundary conditions or the incompressibility condition in the Stokes problem, need not be included in the trial space. This was considered for the particular example of Dirichlet's problem on second order operators in [28] and then extended to the more general setting of $2m$th order elliptic problems in [29]. Later on, in 1985, a key paper was published in the development of the theory of least-squares formulations by Aziz, Kellogg and Stephens [11] which made a connection between least-squares and the much earlier work of Agmon, Douglis and Nirenberg [1, 2]. The theory in these two papers is now referred to as ADN theory and it provides a more general definition of ellipticity, now called ADN ellipticity, which provided the framework for the analysis of least-squares problems in the paper by Aziz et al. [11].

It was not until the idea of recasting problems into equivalent first order systems (see e.g. [24]) that research into least-squares methods gained a considerable amount of interest. This was because in earlier works the practicality of the method was limited by the fact that the discretisation required C^1 or higher finite/spectral element spaces. This meant that standard piecewise continuous finite/spectral element methods could not be used and it also led to algebraic systems with high condition numbers [11]. This idea has lead to a large amount of applications of least-squares methods as well as theoretical work. Two excellent resources on the recent progression into least-squares methods can be found in the books of Jiang [79] and Bochev and Gunzburger [25].

This chapter is structured as follows: We begin by considering an abstract formulation in order to introduce the main concepts involved in least-squares methods. In Section 3.3 a proof of convergence is supplied for least-squares PGD algorithms for the abstract formulation. In the final three sections we consider particular examples of the Poisson equation, Convection-Diffusion equation and the Stokes problem. In the case of the Poisson equation we compare rates of convergence with the Galerkin PGD algorithm introduced in the previous chapter.

3.2 Abstract Least-Squares Formulation

3.2.1 The Abstract Problem

Consider the following abstract boundary value problem

$$\mathcal{L}u = f \quad \text{in} \quad \Omega, \tag{3.1}$$

$$\mathcal{R}u = g \quad \text{on} \quad \Gamma = \partial\Omega, \tag{3.2}$$

where $\mathcal{L}$ is a linear elliptic partial differential operator, $\mathcal{R}$ is a trace operator and f and g are given functions. We further assume that $\mathcal{L}$ is a first order differential operator since we can recast any higher order problem into equivalent systems of first order differential equations and in the least-squares method we need to do this in order to construct a practical method. We elaborate on this later.

Now if we also assume the above boundary value problem is well-posed and that there exists a homeomorphism $\{\mathcal{L}, \mathcal{R}\} : X \to Y \times Z$ where $X = X(\Omega), Y = Y(\Omega)$ and $Z = Z(\Gamma)$ are some underlying Hilbert spaces with norms $\|\cdot\|_X, \|\cdot\|_Y$ and $\|\cdot\|_Z$, respectively, then there exist constants $C_1, C_2 > 0$ such that:

$$C_1\|u\|_X \leq \|\mathcal{L}u\|_Y + \|\mathcal{R}u\|_Z \leq C_2\|u\|_X, \quad \forall u \in X. \tag{3.3}$$

If we let $\tilde{u}$ denote the unique solution of (3.1)-(3.2) then using the inequality (3.3) we can write

$$C_1\|u - \tilde{u}\|_X \leq \|\mathcal{L}u - f\|_Y + \|\mathcal{R}u - g\|_Z \leq C_2\|u - \tilde{u}\|_X, \quad \forall u \in X. \tag{3.4}$$

This norm equivalence between the error in the the X-norm and the residual in the differential equation in the $Y \times Z$-norm is termed the coercivity estimate (or *a priori* estimate) and it is the key ingredient in the analysis of least-squares methods. This is due to the fact that (3.4) implies that if we had a sequence of functions $u_n \in X$ such that $\|\mathcal{L}u_n - f\|_Y \to 0$ and $\|\mathcal{R}u_n - g\|_Z \to 0$ as $n \to \infty$ then $\|u_n - \tilde{u}\|_X \to 0$ as $n \to \infty$ and vice versa. This means that the sequence u_n converges to the true solution in the X-norm. Therefore, minimisation of the convex functional:

$$\mathcal{J}(u) = \|\mathcal{L}u - f\|_Y^2 + \|\mathcal{R}u - g\|_Z^2, \quad \forall u \in X, \tag{3.5}$$

yields the unique solution $\tilde{u}$ to the boundary value problem (3.1)-(3.2). In fact this functional $\mathcal{J}$ is the previously mentioned quadratic least squares functional and it has been proven (see e.g. [25]) that the unique minimiser of $\mathcal{J}$ is the unique solution $\tilde{u}$. We are then able to derive the Euler-Lagrange equation associated with the minimisation of (3.5): Find $u \in X$ such that:

$$\begin{aligned}
0 =& \lim_{\epsilon \to 0} \frac{d}{d\epsilon} \mathcal{J}(u + \epsilon v), \qquad \forall v \in X \\
=& \lim_{\epsilon \to 0} \frac{d}{d\epsilon} (\|\mathcal{L}u - f + \epsilon \mathcal{L}v\|_Y^2 + \|\mathcal{R}u - g + \epsilon \mathcal{R}v\|_Z^2) \\
=& \lim_{\epsilon \to 0} \Big(\langle \mathcal{L}v, \mathcal{L}u - f \rangle_Y + \langle \mathcal{R}v, \mathcal{R}u - g \rangle_Z + 2\epsilon\big(\langle \mathcal{L}v, \mathcal{L}v \rangle_Y + \langle \mathcal{R}v, \mathcal{R}v \rangle_Z\big)\Big) \\
=& \langle \mathcal{L}v, \mathcal{L}u - f \rangle_Y + \langle \mathcal{R}v, \mathcal{R}u - g \rangle_Z,
\end{aligned}$$

where $\langle \cdot, \cdot \rangle_Y$ and $\langle \cdot, \cdot \rangle_Z$ denote the Y and Z inner products, respectively. This then leads to the following variational formulation: Find $u \in X$ such that:

$$A(u, v) = L(v), \quad \forall v \in X \tag{3.6}$$

where

$$A(u, v) = \langle \mathcal{L}u, \mathcal{L}v \rangle_Y + \langle \mathcal{R}u, \mathcal{R}v \rangle_Z, \quad L(v) = \langle f, \mathcal{L}v \rangle_Y + \langle g, \mathcal{R}v \rangle_Z.$$

We can see from this that least-squares methods always yield symmetric linear systems. This is a major advantage of least-squares methods since it means one is able to use robust iterative solvers such as the conjugate gradient method for problems which may not yield symmetric systems in the Galerkin formulation of the same problem.

At the beginning of this section we made the assumption that Hilbert spaces X, Y and Z exist and that they provide a homeomorphism $\{\mathcal{L}, \mathcal{R}\} : X \to Y \times Z$. The difficulty lies in choosing suitable Hilbert spaces so that the problem is well-defined and such a homeomorphism exists. These assumptions are valid for a large number of PDEs. In particular, these assumptions are true for well-posed ADN elliptic PDEs. The selection of suitable Hilbert spaces then follows directly from the ADN theory. We elaborate on this in the following section.

3.2.2 ADN Theory

The theory of Agmon, Douglis and Nirenberg (ADN theory) was developed in a series of two papers published in 1959 [1] and 1964 [2], almost a decade before the first papers on least-squares methods. The first of these papers was concerned with equations of just a single dependent variable and the second paper extended the theory to cover systems involving multiple dependent variables. Its importance in relation to least-squares methods was first made clear in the paper by Aziz et al. [11] in 1985 and we shall cover the relevant elements of ADN theory that were used in this paper. We begin by considering again the abstract boundary value problem:

$$\mathcal{L}u = f \quad \text{in} \quad \Omega, \tag{3.7}$$

$$\mathcal{R}u = g \quad \text{on} \quad \Gamma, \tag{3.8}$$

where $\mathcal{L}$ and $\mathcal{R}$ are as in (3.1)-(3.2) with constant coefficients and where $\mathcal{L} = \mathcal{L}_{i,j}(D)$, $i, j = 1, \ldots, n$ and $\mathcal{R} = \mathcal{R}_{l,j}(D)$, $l = 1, \ldots, m$, $j = 1, \ldots, n$, where n is the number of dependent variables, m is the number of boundary conditions and D is the differential operator:

$$D = (\partial/\partial x_1, \ldots, \partial/\partial x_d)^T,$$

where d is the number of independent variables (the dimension). If we consider $\mathcal{L}_{i,j}$ evaluated at a general d-dimensional vector $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_d)$ rather that at the differential operator vector D then the usual definition of ellipticity is that $\det(\mathcal{L}^p_{i,j}(\boldsymbol{\xi})) \neq 0$ for all real-valued $\boldsymbol{\xi} \neq 0$. Here the principal part $\mathcal{L}^p$ of $\mathcal{L}$ is defined to be:

$$\mathcal{L}^p \equiv \mathcal{L}^p_{i,j}(D) = \begin{cases} \mathcal{L}_{i,j}(D), & \text{if} \quad \deg(\mathcal{L}_{i,j}(\boldsymbol{\xi})) = \max\limits_{k,l} \deg(\mathcal{L}_{k,l}(\boldsymbol{\xi})), \\ 0, & \text{otherwise.} \end{cases}$$

For example, if

$$\mathcal{L} = \mathcal{L}_{i,j}(D) = \begin{pmatrix} \frac{\partial^2}{\partial x_1^2} & 1 \\ \frac{\partial}{\partial x_2} & \frac{\partial^2}{\partial x_1 \partial x_2} \end{pmatrix},$$

then

$$\mathcal{L}^p_{i,j}(\boldsymbol{\xi}) = \begin{pmatrix} \xi_1^2 & 0 \\ 0 & \xi_1\xi_2 \end{pmatrix}.$$

It can then be seen that the above operator is not elliptic since $\det(\mathcal{L}^p_{i,j}(\boldsymbol{\xi})) = 0$ for $\boldsymbol{\xi} = (0, 1)^T$.

To extend the idea of ellipticity to the more general idea of ADN ellipticity we need to introduce two sets of integer indices: the set $\{s_i\}$, $s_i \leq 0$, assigned to the n equations and the set $\{t_j\}$, $t_j \geq 0$, assigned to the n dependent variables. These indices are chosen in such a way that for each $i, j = 1, \ldots, n$, we have that $\deg(\mathcal{L}_{i,j}(\boldsymbol{\xi})) \leq s_i + t_j$. The principal part $\mathcal{L}^p$ is then defined to be:

$$\mathcal{L}^p \equiv \mathcal{L}^p_{i,j}(D) = \begin{cases} \mathcal{L}_{i,j}(D), & \text{if} \quad \deg(\mathcal{L}_{i,j}(\boldsymbol{\xi})) = s_i + t_j, \\ 0, & \text{otherwise.} \end{cases}$$

The principal part $\mathcal{R}^p$ can be defined analogously by introducing the set of indices $\{r_l\}$, $r_l \leq 0$, assigned to the m boundary conditions such that $\deg(\mathcal{R}_{l,j}(\boldsymbol{\xi})) \leq r_l + t_j$. Note that the choice of indices is not, in general, unique and hence problems can have more than one principal part. We are now in a position to present the following definition of ADN ellipticity:

Definition 1. *The linear differential operator $\mathcal{L}$ is called ADN elliptic if there exists integer sets $\{s_i\}$, $i = 1, \ldots, n$, and $\{t_j\}$, $j = 1, \ldots, n$, such that:*

(i) $\deg(\mathcal{L}_{i,j}(\boldsymbol{\xi})) \leq s_i + t_j$,

(ii) $\mathcal{L}_{i,j}(\boldsymbol{\xi}) \equiv 0$ *if* $s_i + t_j < 0$,

(iii) $\det(\mathcal{L}^p_{i,j}(\boldsymbol{\xi})) \neq 0$ *for all real-valued* $\boldsymbol{\xi} \neq 0$.

We further say $\mathcal{L}$ is ADN elliptic of order $2m$ if $\deg(\det(\mathcal{L}^p_{i,j}(\boldsymbol{\xi}))) = 2m$ and uniformly ADN elliptic of order $2m$ if there exists a constant $c_e > 0$ such that

$$c_e^{-1}|\boldsymbol{\xi}|^{2m} \leq |\det(\mathcal{L}^p_{i,j}(\boldsymbol{\xi}))| \leq c_e|\boldsymbol{\xi}|^{2m},$$

where m is, as before, the number of prescribed boundary conditions.

Note that ellipticity in the usual sense is associated with the indices $s_i = 0$, $t_j = 1$ for all $i, j = 1, \ldots, n$. From now on we refer to systems of this type as homogeneous elliptic.

For a problem to be well-posed the operators $\mathcal{L}$ and $\mathcal{R}$ cannot be chosen independently. They must be chosen in such a way that their principal parts $\mathcal{L}^p$ and $\mathcal{R}^p$ 'complement' one another. As such Agmon, Douglis and Nirenberg [2] introduced a so called complementing condition which we shall outline after presenting the following supplementary condition which must be satisfied by $\mathcal{L}$:

Definition 2. ***Supplementary condition***
Let $\mathcal{L}$ be ADN elliptic of order $2m$ then the operator $\mathcal{L}$ is said to satisfy the supplementary condition if for all pairs of linearly independent real-valued vectors $\boldsymbol{\xi}$ and $\boldsymbol{\xi}'$ the polynomial in τ given by $\det(\mathcal{L}^p_{i,j}(\boldsymbol{\xi} + \tau\boldsymbol{\xi}'))$ has exactly m roots with positive imaginary part.

Remark 5. *This supplementary condition is automatically satisfied when the dimension of the problem is greater than two and hence it only needs to be checked for 2D problems [2].*

The following useful lemma and proof can be found in the PhD thesis of Proot [111]:

Lemma 4. *If the determinant of the principal part satisfies:*

$$\det(\mathcal{L}^p_{i,j}(\boldsymbol{\xi})) = c|\boldsymbol{\xi}|^{2m},$$

for some constant c, then the supplementary condition is satisifed.

Proof. For linearly independent real vectors $\boldsymbol{\xi}$ and $\boldsymbol{\xi}'$ we have that

$$\det(\mathcal{L}^p_{i,j}(\boldsymbol{\xi}+\tau\boldsymbol{\xi}')) = c(|\boldsymbol{\xi}|^2 + 2\tau\boldsymbol{\xi}\cdot\boldsymbol{\xi}' + \tau^2|\boldsymbol{\xi}'|^2)^m$$

which has the following roots of multiplicity m:

$$\tau_{1,2} = \frac{-\boldsymbol{\xi}\cdot\boldsymbol{\xi}' \pm \sqrt{(\boldsymbol{\xi}\cdot\boldsymbol{\xi}')^2 - |\boldsymbol{\xi}|^2|\boldsymbol{\xi}'|^2}}{|\boldsymbol{\xi}'|^2},$$

and since we have that

$$(\boldsymbol{\xi}\cdot\boldsymbol{\xi}')^2 < |\boldsymbol{\xi}|^2|\boldsymbol{\xi}'|^2,$$

for linearly independent vectors $\boldsymbol{\xi}$, $\boldsymbol{\xi}'$, then the roots τ_1 and τ_2 are complex conjugate. Therefore exactly one of τ_1 and τ_2 has positive imaginary part and since it has multiplicity m there are exactly m roots with positive imaginary part. Hence the supplementary condition is satisfied.

□

We now introduce the following notation: Let $\tau^+_k(\boldsymbol{\xi})$, $k = 1,\dots,m$, denote the m roots of $\det(\mathcal{L}^p_{i,j}(\boldsymbol{\xi}+\tau\boldsymbol{\xi}'))$ whose existence is ensured by the supplementary condition and let $M^+(\boldsymbol{\xi},\tau)$ with positive imaginary part be the polynomial in τ for a given $\boldsymbol{\xi}$ given by:

$$M^+(\boldsymbol{\xi},\tau) = \prod_{k=1}^{m}\Big(\tau - \tau^+_k(\boldsymbol{\xi})\Big).$$

The complementing condition is as follows:

Definition 3. ***Complementing condition***

For any point $P \in \Gamma$, let $\mathbf{n}$ be the outward unit normal vector to Γ at P. For any real valued $\boldsymbol{\xi} \neq \mathbf{0}$ tangent to Γ at P consider the matrix with the following entries:

$$\sum_{j=1}^{n} \mathcal{R}^p_{l,j}(\boldsymbol{\xi}+\tau\mathbf{n})\mathcal{L}'_{j,k}(\boldsymbol{\xi}+\tau\mathbf{n}), \tag{3.9}$$

which are polynomials in τ and where $\mathcal{L}'$ denotes the adjoint matrix of $\mathcal{L}^p$. The operators $\mathcal{L}$ and $\mathcal{R}$ are said to satisfy the complementing condition if the rows of the matrix defined by (3.9) *are linearly independent modulo $M^+(\boldsymbol{\xi},\tau)$. In other words:*

$$\sum_{l=1}^{m} c_l \sum_{j=1}^{n} \mathcal{R}^p_{l,j}\mathcal{L}'_{j,k} \equiv 0 \pmod{M^+}, \quad \forall k = 1,\dots,n,$$

if and only if $c_l = 0$ for all $l = 1,\dots,m$.

We are now in a position to present the key theorem for least-squares methods in the ADN theory since it provides us with the required *a priori* estimates and the associated functional spaces. In what follows we shall use the notation that $\|\cdot\|_i$

and $\| \cdot \|_{i,\Gamma}$ are norms on the spaces $H^i(\Omega)$ and $H^i(\Gamma)$, respectively, and equivalently for their inner products: $\langle \cdot, \cdot \rangle_i$ and $\langle \cdot, \cdot \rangle_{i,\Gamma}$. The following theorem has been proven in [2]:

Theorem 5. *Let $\mathcal{L}$ be a uniformly ADN elliptic operator of order $2m$ which satisfies the supplementary condition and together with the trace operator $\mathcal{R}$ satisfies the complementing condition. Then assume that for some $q \geq 0$: $u \in \prod_{j=1}^{n} H^{q+t_j}(\Omega)$, $f \in \prod_{i=1}^{n} H^{q-s_i}(\Omega)$ and $g \in \prod_{l=1}^{m} H^{q-r_l-1/2}(\Gamma)$. Then there exists a constant $C > 0$ such that:*

$$\sum_{j=1}^{n} \|u_j\|_{q+t_j} \leq C \left(\sum_{i=1}^{n} \|f_i\|_{q-s_i} + \sum_{l=1}^{m} \|g_l\|_{q-r_l-1/2,\Gamma} + \sum_{j=1}^{n} \|u_j\|_0 \right), \tag{3.10}$$

where $u = (u_1, \ldots, u_n)^T$, $f = (f_1, \ldots, f_n)^T$ and $g = (g_1, \ldots, g_m)^T$. Moreover if the problem (3.7)-(3.8) has a unique solution then the term on the RHS of (3.10) involving the L^2-norm can be omitted.

We can always ensure our problem has an unique solution (and hence the L^2-norm term can be omitted) by including additional constraints (e.g. the zero mean pressure constraint in the Stokes problem). Indeed, if we define our k additional constraints to be given by $\ell(u) = c$ where $\ell : X \to \mathbb{R}^k$ then we can include this in the quadratic least-squares functional (3.5) in the following way:

$$\mathcal{J}(u) = \|\mathcal{L}u - f\|_Y^2 + \|\mathcal{R}u - g\|_Z^2 + |\ell(u) - c|^2, \quad \forall u \in X. \tag{3.11}$$

We are then able to derive the Euler-Lagrange equations associated with the minimisation problem (3.11): Find $u \in X$ such that:

$$A(u, v) = L(v), \quad \forall v \in X$$

where

$$\begin{aligned} A(u, v) &= \langle \mathcal{L}u, \mathcal{L}v \rangle_Y + \langle \mathcal{R}u, \mathcal{R}v \rangle_Z + \ell(u) \cdot \ell(v), \\ L(v) &= \langle f, \mathcal{L}v \rangle_Y + \langle g, \mathcal{R}v \rangle_Z + c \cdot \ell(v). \end{aligned}$$

Note that the theory provided in the previous chapter can be extended to cover inclusion of constraints by assuming that there exists a homeomorphism $\{\mathcal{L}, \mathcal{R}, \ell\} : X \to Y \times Z \times \mathbb{R}^k$. The same results then follow including the fact that the least squares functional (3.11) provides the unique solution of the PDE (see Bochev and Gunzburger [25]).

Armed with the knowledge that we can always obtain a unique solution it follows that Theorem 5 has provided us with the lower bound in the *a priori* estimate (3.3). To see this we let $X = \prod_{j=1}^{n} H^{q+t_j}(\Omega)$, $Y = \prod_{i=1}^{n} H^{q-s_i}(\Omega)$

and $Z = \prod_{l=1}^{m} H^{q-r_l-1/2}(\Gamma)$ with corresponding norms $\| \cdot \|_X = \sum_{j=1}^{n} \| \cdot \|_{q+t_j}$, $\| \cdot \|_Y = \sum_{i=1}^{n} \| \cdot \|_{q-s_i}$ and $\| \cdot \|_Z = \sum_{l=1}^{m} \| \cdot \|_{q-r_l-1/2,\Gamma}$. The inequality (3.10) then reduces to:

$$\|u\|_X \leq C(\|f\|_Y + \|g\|_Z) = C(\|\mathcal{L}u\|_Y + \|\mathcal{R}u\|_Z).$$

The upper bound of the *a priori* estimate (3.3) follows directly from the continuity of the operator $\{\mathcal{L}, \mathcal{R}\}$ and combining this with the above lower bound we obtain the required *a priori* estimate and, in particular, the appropriate choices of the Hilbert spaces X, Y and Z.

This concludes this section on ADN theory but before we can apply such formulations to PGD algorithms we first need to mention some practical issues that must be considered.

3.2.3 Practical Issues

We mentioned earlier that for least-squares formulations to be a practical method for approximating the solution of differential equations it is necessary to first recast the problem as a first-order system. The reason for this is clear if we assume that, in the above material, $\mathcal{L}$ is a second-order differential operator and $Y = L^2(\Omega)$. The variational formulation (3.6) then requires one to evaluate the inner product $\langle \mathcal{L}u, \mathcal{L}v \rangle_0$. Notice that the differential operator appears in both arguments of the inner product and hence, unlike in Galerkin formulations, we cannot weaken the required differentiability on the space X via Green's first integral identity. This means that we would need to provide a conforming discretisation space $X^h \subset X$ that is C^1-continuous. This is a property that is not satisfied by standard linear finite/spectral element approximations over element edges. However, while there are finite element methods that can be constructed which are C^1-continuous on element edges, they tend to be difficult to work with and are impractical [119]. It is often the case, for example in Galerkin formulations of fourth-order problems such as the equations governing plate bending, that non-conforming finite element methods are used instead such that $X^h \not\subset X$. However, in least-squares methods it becomes unclear how the norm equivalence in the coercivity estimate (3.4) is affected if non-conforming discretisation spaces are used [25]. It is for this reason that we must recast the problem into a system of first-order differential equations so that $\mathcal{L}$ is a first-order differential operator.

Secondly, we also require that the differentiability of the spaces X and Y do not exceed 1 and 0, respectively. Indeed, if, for example, $X = H^2(\Omega)$ and/or $Y = H^1(\Omega)$ then a conforming discretisation space $X^h \subset X$ would again need to be C^1-continuous due to the derivatives that appear in H^2 and H^1-norms.

Systems that do not suffer from these two practical issues are called homogeneous elliptic. For first-order systems they are associated with the choice of indices $s_i = 0$, $t_j = 1$ for all $i, j = 1, \ldots, n$. Homogeneous elliptic systems are therefore what we earlier referred to as elliptic in the usual sense. However, ellipticity in the usual sense does not take into account compatibility between the differential operator and the boundary conditions. For non-homogeneous elliptic systems, the least-squares method can still be made practical by extending the coercivity estimate in Theorem 5 to hold for negative q. Unfortunately, this has the side-effect of introducing negative index norms into the least-squares functional. These negative index norms are problematic to work with. Indeed, the minus one norm given by (see e.g. [25]):

$$\|f\|_{-1} = \sup_{u \in H_0^1(\Omega)} \frac{\langle f, u \rangle_0}{\|u\|_1},$$

does not lend itself to being computed easily using a finite/spectral element method.

Besides these negative index norms we also have problematic trace norms (i.e. the Z-norm). Indeed, trace norms on fractional Sobolev spaces are defined by (see e.g. [25]):

$$\|u\|_{s-1/2} = \inf_{v \in H_g^s(\Omega)} \|v\|_s,$$

where $H_g^s(\Omega) = \{v \in H^s(\Omega) \ : \ v = g \ \text{ on } \ \Gamma\}$. This again does not lend itself to be easily computed by finite/spectral element methods. To make these problematic norms more practical we replace them by specially weighted L^2-norms. The fundamental idea which allows us to do this is that in finite dimensional spaces all norms are equivalent. Therefore, upon the discretisation of the problem the continuous coercivity estimate from Theorem 5 is somehow preserved. More formally, consider the norm generating operators $\mathcal{S}_Y$ and $\mathcal{S}_Z$ such that we can rewrite the energy norm, $|||\cdot|||$, in terms of L^2-norms:

$$|||u||| := \|\mathcal{L}u\|_Y + \|\mathcal{R}u\|_Z = \|\mathcal{S}_Y \circ \mathcal{L}u\|_0 + \|\mathcal{S}_Z \circ \mathcal{R}u\|_{0,\Gamma}.$$

We can compute discrete approximations for the norm generating operators such that we have the following discrete energy norm:

$$\left|\left|\left|u^h\right|\right|\right|_h := \|\mathcal{S}_Y^h \circ \mathcal{L}^h u^h\|_0 + \|\mathcal{S}_Z^h \circ \mathcal{R}^h u^h\|_{0,\Gamma}.$$

The choice of approximations for the discrete norm generating operators can lead to two distinct cases. Firstly we can retain the norm equivalence in the discrete energy norm, so that there exists a constant c such that:

$$c\|u^h\|_X \leq \left|\left|\left|u^h\right|\right|\right|_h \leq c\|u^h\|_X.$$

This is the most desirable situation since the coercivity estimate is preserved and optimal rates of convergence in h can be achieved [25]. Secondly we can obtain only quasi-norm equivalence in the discrete energy norm such that the constants in the norm equivalence now depend on the mesh parameter h. i.e. we have that:

$$c(h)\|u^h\|_X \leq \left|\!\left|\!\left| u^h \right|\!\right|\!\right|_h \leq c(h)\|u^h\|_X.$$

The dependence on h means that it becomes unclear, as $h \to 0$, how well the discrete energy norm $|\!|\!|\cdot|\!|\!|_h$ represents the true energy norm $|\!|\!|\cdot|\!|\!|$. In particular this means that, in general, optimal rates of convergence in h are not guaranteed and also it can lead to high condition numbers [25]. It is also unclear in the context of the PGD how this discrete norm equivalence is affected by the rank, J, of the PGD approximation.

A potential advantage, in the context of least-squares PGD, is that Dirichlet boundary conditions can be imposed weakly. This means we can avoid the potentially difficult task of constructing a transfinite interpolating function that satisfies the boundary conditions. However, in practice we try to avoid weakly imposed boundary conditions since the approximation of the norm generating operators $\mathcal{S}_Z$ can lead to linear systems that are severely ill-conditioned [111]. This means it is currently not viable to impose boundary conditions weakly despite the potential advantages in the context of the PGD.

There is a further issue with non-standard boundary conditions such as the imposition of different boundary conditions on different parts of Γ. These kind of boundary conditions are not covered by the ADN theory since they cannot be expressed as a linear boundary operator $\mathcal{R}$. For specific cases, coercivity estimates, such as the ones provided by Theorem 5 in the ADN theory, can be derived from their vector-operator setting for problems with non-standard boundary conditions (see [25], for example). To provide us with a simpler template to investigate least-squares PGD algorithms we will avoid these non-standard boundary conditions. As a result, in this work, we will only consider Dirichlet boundary conditions defined on the whole of Γ. To this end we define the affine subspace, X_g, whose elements satisfy the boundary conditions

$$X_g = \{u \in X : \mathcal{R}u = g \quad \text{on} \quad \Gamma\},$$

hence $\|\mathcal{R}u - g\|_Z = 0$ for $u \in X_g$. This leads to the simplified quadratic least squares functional:

$$\mathcal{J}(u) = \|\mathcal{L}u - f\|_Y^2, \quad \forall u \in X_g,$$

and associated Euler-Lagrange equation: Find $u \in X_g$ such that:

$$A(u, v) = L(v), \quad \forall v \in X_0 \tag{3.12}$$

where

$$A(u,v) = \langle \mathcal{L}u, \mathcal{L}v \rangle_Y, \quad L(v) = \langle f, \mathcal{L}v \rangle_Y.$$

A final practical issue we must mention is an issue related to the implementation of least-squares methods in the PGD. The issue is that if we have a problem defined in high-dimensional space then a first-order reformulation of such a problem would have a large number of dependent variables. For example, given a 100-dimensional Poisson equation, a Div-Grad type formulation of this problem would have 101 dependent variables. This means that the number of unknowns in a least-squares PGD algorithm will no longer grow linearly as the dimension increases. Indeed, consider a Galerkin PGD formulation of a problem in d-dimensional space which we have previously mentioned has $N \times J \times d$ unknowns. A least-squares formulation of the same problem would instead have $N \times J \times d \times d^{\alpha}$ unknowns, where d^{α} represents the rate at which the number of dependent variables increases with the dimension d. This means that for least-squares PGD algorithms we will obtain, at best, a quadratic rate of increase of the number of unknowns as the dimension increases. While this is certainly worse than for Galerkin PGD algorithms it is still a vast improvement over the exponential rate of increase one would obtain with a standard tensor-product based approach.

We now turn our attention to convergence of least-squares PGD algorithms. One of the key reasons that we are interested in using least-squares formulations in conjunction with the PGD is that they provide us with a minimisation principle of an artificial energy functional. This enables us to define associated greedy algorithms for which convergence can be proved.

3.3 Convergence of Least-Squares PGD Algorithms

We aim to prove that least-squares PGD algorithms converge for all problems which fit into the abstract theory covered in the last section (i.e. all linear ADN elliptic problems). We shall again consider the proof based on minimisation of energies by Cancès et al. [37] as well as the proof based on a generalised Eckart-Young theorem by Falcó and Nouy. [66]

3.3.1 Energy Minimisation

The fundamental issue that prevented us from providing a proof of convergence of a Galerkin PGD algorithm for the Stokes problem was the absence of an unconstrained energy minimisation principle (Rayleigh-Ritz setting) that could be used to define an associated greedy algorithm. The least-squares PGD algorithm has overcome this by providing us with an artificial energy functional:

$$\mathcal{J}(u) = \|\mathcal{L}u - f\|_Y^2 \tag{3.13}$$

Recall from Section 2.4.2 that to ensure that we can include least-squares PGD problems into the general theoretical setting outlined in [37], in which we are able to prove convergence of the associated greedy algorithms, the following two assumptions on $\mathcal{J}$ must hold:

(A1) $\mathcal{J}$ is strongly convex for $\|\cdot\|_X$ so that there exists a constant $\alpha > 0$ such that for $t \in [0,1]$:

$$\mathcal{J}(tu + (1-t)v) \leq t\mathcal{J}(u) + (1-t)\mathcal{J}(v) - \frac{\alpha}{2}t(1-t)\|u-v\|_X^2, \quad \forall u, v \in X.$$

We then say that $\mathcal{J}$ is α-convex [77].

(A2) $\mathcal{J}$ is differentiable and its Fréchet derivative is Lipschitz continuous so that there exists a constant $L \geq 0$ such that

$$\|\mathcal{J}'(u) - \mathcal{J}'(v)\|_X \leq L\|u-v\|_X, \quad \forall u, v \in X,$$

where $\mathcal{J}'$ denotes the Fréchet derivative of $\mathcal{J}$.

Note that these are the same assumptions that were listed at the end of Section 2.4.2 and are included here for ease of reference.

Lemma 5. *The least-squares functional* (3.13) *satisfies both the above conditions.*

Proof. The key ingredient to proving that these two conditions hold for the least-squares functional is the coercivity relation arising from the ADN Theory

$$C_1\|u\|_X \leq \|\mathcal{L}u\|_Y \leq C_2\|u\|_X, \ \ \forall u \in X. \tag{3.14}$$

Indeed, since we know that

$$\|u-v\|_X^2 \geq \frac{1}{C_2^2}\|\mathcal{L}(u-v)\|_Y^2 = \frac{1}{C_2^2}\|\mathcal{L}u - \mathcal{L}v\|_Y^2,$$

then proving strong convexity amounts to proving that for $t \in [0,1]$:

$$\mathcal{J}(tu + (1-t)v) \leq t\mathcal{J}(u) + (1-t)\mathcal{J}(v) - \frac{\alpha}{2C_2^2}t(1-t)\|\mathcal{L}u - \mathcal{L}v\|_Y^2, \tag{3.15}$$

for all $u, v \in X$. Indeed, if we consider the left-hand side of (3.15) we have:

$$\begin{aligned}
&\mathcal{J}(tu + (1-t)v) = \ \|t\mathcal{L}u + (1-t)\mathcal{L}v - f\|_Y^2 \\
&= t^2\|\mathcal{L}u\|_Y^2 + (1-t)^2\|\mathcal{L}v\|_Y^2 + \|f\|_Y^2 - 2t\langle \mathcal{L}u, f\rangle_Y - 2(1-t)\langle \mathcal{L}v, f\rangle_Y \\
&\quad + 2t(1-t)\langle \mathcal{L}u, \mathcal{L}v\rangle_Y \\
&= t(\|\mathcal{L}u\|_Y^2 - 2\langle \mathcal{L}u, f\rangle_Y + \|f\|_Y^2) + (1-t)(\|\mathcal{L}v\|_Y^2 - 2\langle \mathcal{L}v, f\rangle_Y + \|f\|_Y^2) \\
&\quad - t(1-t)(\|\mathcal{L}u\|_Y^2 - 2\langle \mathcal{L}u, \mathcal{L}v\rangle_Y + \|\mathcal{L}v\|_Y^2) \\
&= t\|\mathcal{L}u - f\|_Y^2 + (1-t)\|\mathcal{L}v - f\|_Y^2 - t(1-t)\|\mathcal{L}u - \mathcal{L}v\|_Y^2
\end{aligned}$$

which is the right-hand side of (3.15) with $\alpha = 2C_2^2$. Hence $\mathcal{J}$ is $\alpha = 2C_2^2$-convex.

For the second part of the proof we do not need to evaluate explicitly the Fréchet derivative $\mathcal{J}'$. Instead we use the fact that the functional derivative, which is exactly the Euler-Lagrange equation associated with the minimisation of $\mathcal{J}$, is equal to the X-inner product of its Fréchet derivative with a test function. More precisely we know that:

$$\langle \mathcal{J}'(u), w\rangle_X = \langle \mathcal{L}w, \mathcal{L}u - f\rangle_Y, \qquad \forall w \in X.$$

For all $u, v, w \in X$ we have:

$$\begin{aligned}
|\langle \mathcal{J}'(u) - \mathcal{J}'(v), w\rangle_X| &= |\langle \mathcal{J}'(u), w\rangle_X - \langle \mathcal{J}'(v), w\rangle_X| \\
&= |\langle \mathcal{L}w, \mathcal{L}u - f\rangle_Y - \langle \mathcal{L}w, \mathcal{L}v - f\rangle_Y| \\
&= |\langle \mathcal{L}w, \mathcal{L}u - \mathcal{L}v\rangle_Y|,
\end{aligned}$$

and by Cauchy-Schwarz we have that

$$\begin{aligned}
|\langle \mathcal{L}w, \mathcal{L}u - \mathcal{L}v\rangle_Y| &\le \|\mathcal{L}w\|_Y \|\mathcal{L}u - \mathcal{L}v\|_Y \\
&\le C_2^2 \|w\|_X \|u - v\|_X,
\end{aligned}$$

using the coercivity relation (3.14). Hence we have:

$$|\langle \mathcal{J}'(u) - \mathcal{J}'(v), w\rangle_X| \le C_2^2\|w\|_X\|u - v\|_X, \quad \forall u, v, w \in X.$$

In particular, taking $w = \mathcal{J}'(u) - \mathcal{J}'(v)$ yields:

$$\|\mathcal{J}'(u) - \mathcal{J}'(v)\|_X \le C_2^2\|u - v\|_X.$$

Therefore $\mathcal{J}'$ is Lipschitz continuous.

□

Remark 6. *Note that the above proof can be extended trivially to cover the least-*

squares functional with weakly imposed boundary conditions

$$\mathcal{J}(u) = \|\mathcal{L}u - f\|_Y^2 + \|\mathcal{R}u - g\|_Z^2.$$

Again, recall from Section 2.4.2 that there are two additional conditions on the involved functional spaces that must also be satisfied in order for the proof of convergence given in [37] to hold. If we once again let $\mathcal{S}_1$ denote the set of all rank-one tensors then the following conditions must be satisfied:

1. Span$(\mathcal{S}_1)$ is a dense subset of X for $\|\cdot\|_X$.

2. $\mathcal{S}_1$ is weakly closed in $(X, \|\cdot\|_X)$.

The ADN Theory supplies us with a functional space X that is simply a Sobolev space depending on the set of indices defining the principal part of the differential operator $\mathcal{L}$. As a result these two conditions will hold for a least-squares formulated problem. Indeed, a proof of this for the simple case of H^1 spaces can be found in the paper by Cancès et al. [37] in the context of a high-dimensional Poisson equation.

The four conditions are therefore satisfied by a least-squares formulated problem. This means that convergence of the greedy algorithm associated with any least-squares PGD algorithm is covered by the general proof provided by Cancès et al. [37]

3.3.2 Generalised Eckart-Young Approach

A proof of convergence for least-squares PGD algorithms was also given by Falcó and Nouy [66] based on their generalised Eckart-Young theorem approach. Given that the Euler-Lagrange equation associated with the minimisation of the quadratic least-squares functional yields the following variational problem: Find $u \in X$ such that:

$$A(u, v) = L(v), \quad \forall v \in X$$

where

$$A(u, v) = \langle \mathcal{L}u, \mathcal{L}v \rangle_Y, \qquad L(v) = \langle f, \mathcal{L}v \rangle_Y.$$

Now since we can write this equivalently as:

$$A(u, v) = \langle \mathcal{L}^*\mathcal{L}u, v \rangle_X, \qquad L(v) = \langle \mathcal{L}^* f, v \rangle_X,$$

then we can introduce the inner product $\langle \cdot, \cdot \rangle_{\mathcal{L}}$ induced by the operator $\mathcal{L}^*\mathcal{L}$:

$$\langle u, v \rangle_{\mathcal{L}} = \langle \mathcal{L}^*\mathcal{L}u, v \rangle_X = \langle \mathcal{L}u, \mathcal{L}v \rangle_Y,$$

and associated norm:

$$\|u\|_{\mathcal{L}} = \sqrt{\langle u, u \rangle_{\mathcal{L}}}$$

Now since $\|u\|_{\mathcal{L}}^2 = \|\mathcal{L}u\|_Y^2$ then norm equivalence between $\|\cdot\|_{\mathcal{L}}$ and $\|\cdot\|_X$ follows directly from the coercivity estimate (3.3). Hence under the assumption that $\mathcal{S}_1$ is weakly closed in $(X, \|\cdot\|_X)$ we have that it is also weakly closed in $(X, \|\cdot\|_{\mathcal{L}})$ since equivalent norms induce the same weak topology. For a given $z \in X$ we then use the $\mathcal{L}$-norm to define an associated rank-one projector $\Pi_{\mathcal{L}}(z)$ with which we can define the optimal progressive rank-J separated representation of the solution $u = \mathcal{L}^{-1}f$ by:

$$u_J = \sum_{j=1}^{J} u^{(j)}, \quad u^{(j)} \in \Pi_{\mathcal{L}}(u - u_{j-1}).$$

The generalised Eckart-Young Theorem in [66] then ensures that this sequence converges as $J \to \infty$. The additional assumption that $\mathrm{Span}(\mathcal{S}_1)$ is a dense subset of X for $\|\cdot\|_X$ ensures that it converges to the solution $u = \mathcal{L}^{-1}f$.

Remark 7. *As before this proof can be trivially extended to cover the case where we have weakly imposed boundary conditions so that we have:*

$$A(u, v) = \langle \mathcal{L}u, \mathcal{L}v \rangle_Y + \langle \mathcal{R}u, \mathcal{R}v \rangle_Z.$$

3.3.3 Rate of Convergence

While there is no theoretical results of convergence rates for PGD algorithms specific to least-squares formulations, it has been noted by Nouy [106] that PGD algorithms based on least-squares formulations converge slower than their Galerkin counterparts. It was also noted that the rate of convergence could be improved by weighting the norms in the quadratic least-squares functional. To investigate this further we shall firstly consider least-squares formulations of the Poisson equation so we can compare rates of convergence with earlier results on the Galerkin formulation. We will not use the Stokes problem for this comparison due to the issues related to the lack of LBB-like stability.

3.4 Least-Squares Formulation of the Poisson Equation

We begin by noting that there is no practical use in solving a least-squares formulation of the Poisson equation. This is because it already possesses a natural Rayleigh-Ritz setting and hence there are no benefits to be gained in constructing an artificial one by residual minimisation. The reason we have chosen to apply this formulation is to be able to compare convergence rates of PGD algorithms based on least-squares and Galerkin formulations of equivalent problems. We hope this will give us a better understanding of how the rates of convergence differ in both formulations and, in particular, how it can be improved.

We consider the following Poisson equation:

$$\begin{aligned} -\nabla^2\phi &= f \quad \text{in } \ \Omega, \\ \phi &= g \quad \text{on } \ \Gamma. \end{aligned} \tag{3.16}$$

To begin applying least-squares formulations we must recast this as a first-order system. There are several ways this can be done but we shall only consider reformulations that preserve the Dirichlet boundary conditions. This is so we can avoid the practical issues associated with weakly imposed boundary conditions in least-squares methods that we mentioned earlier.

3.4.1 Div-Grad System

The simplest way to recast (3.16) in the form of a first order system is to introduce the vector $\mathbf{u} = -\nabla\phi$ and since $-\nabla^2\phi = -\nabla\cdot\nabla\phi = \nabla\cdot\mathbf{u}$ which leads to the following Div-Grad system equivalent to (3.16):

$$\nabla\cdot\mathbf{u} = f \quad \text{in } \ \Omega, \tag{3.17}$$

$$\mathbf{u} + \nabla\phi = 0 \quad \text{in } \ \Omega, \tag{3.18}$$

$$\phi = g \quad \text{on } \ \Gamma. \tag{3.19}$$

where, in 2D, $\mathbf{u} = (u, v, 0)^T$. If we define $\boldsymbol{v} = (\phi, u, v)^T$, then the following representations of the 2D Div-Grad operator and boundary operator are obtained:

$$\boldsymbol{\mathcal{L}}\boldsymbol{v} = \begin{pmatrix} 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial x} & 1 & 0 \\ \frac{\partial}{\partial y} & 0 & 1 \end{pmatrix} \begin{pmatrix} \phi \\ u \\ v \end{pmatrix}, \quad \boldsymbol{\mathcal{R}}\boldsymbol{v} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \phi \\ u \\ v \end{pmatrix}.$$

Curiously, this first-order system is not elliptic in the usual sense even though the Poisson equation (3.16) is. Indeed, the standard principal part, which for a first-order system is associated with the ADN indices $s_i = \{0, 0, 0\}$, $t_j = \{1, 1, 1\}$, is given by:

$$\boldsymbol{\mathcal{L}}^p = \begin{pmatrix} 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial x} & 0 & 0 \\ \frac{\partial}{\partial y} & 0 & 0 \end{pmatrix}.$$

and this is not elliptic since $\det(\boldsymbol{\mathcal{L}}^p(\boldsymbol{\xi})) = 0$. Unfortunately, this means that this system is non-homogeneous elliptic for any choice of boundary condition. This system is, however, ADN elliptic. Indeed, if we make the choice of indices $s_i =$

$\{0, -1, -1\}$, $t_j = \{2, 1, 1\}$ then we obtain the following principal part:

$$\mathcal{L}_1^p = \begin{pmatrix} 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial x} & 1 & 0 \\ \frac{\partial}{\partial y} & 0 & 1 \end{pmatrix}.$$

From Definition 1 it is clear that this choice of indices satisfies all the conditions for ADN ellipticity. Indeed condition (i) only needs to be checked when $s_i + t_j < 1$ since it is a first-order system. This only occurs when $i, j = 2, 3$ at which point the operator $\mathcal{L}_{i,j}(\boldsymbol{\xi})$ is constant (degree 0) hence (i) is satisfied. Condition (ii) is trivially satisfied since $s_i + t_j \geq 0$ for all $i, j = 1, 2, 3$ and condition (iii) also holds since:

$$\det(\mathcal{L}_1^p(\boldsymbol{\xi})) = \begin{vmatrix} 0 & \xi_1 & \xi_2 \\ \xi_1 & 1 & 0 \\ \xi_2 & 0 & 1 \end{vmatrix} = -\xi_1^2 - \xi_2^2 = -|\boldsymbol{\xi}|^2. \tag{3.20}$$

This tells us that $\mathcal{L}_1^p$ is uniformly ADN elliptic of order two which means we must impose $m = 1$ boundary condition. Indeed, this is what we would expect since this is same as in the standard definition of the Poisson equation (3.16). It is also clear from (3.20) and Lemma 4 that this also satisfies the supplementary condition.

To verify the complementing condition we first define the principal part of the boundary operator with respect the boundary index $r_l = \{-2\}$. This simply yields a principal part such that $\mathcal{R}^p = \mathcal{R}$. If we now consider the vector, $\boldsymbol{\eta} = \boldsymbol{\xi} + \tau\mathbf{n} = (\eta_1, \eta_2)^T$, then one can readily calculate the adjoint of $\mathcal{L}_1^p(\boldsymbol{\eta})$ by:

$$\mathcal{L}_1'(\boldsymbol{\eta}) = \begin{pmatrix} 1 & -\eta_1 & -\eta_2 \\ -\eta_1 & -\eta_2^2 & \eta_1\eta_2 \\ -\eta_2 & \eta_1\eta_2 & -\eta_1^2 \end{pmatrix}.$$

Hence we have that:

$$\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}_1'(\boldsymbol{\eta}) = \begin{pmatrix} 1 & -\eta_1 & -\eta_2 \end{pmatrix}. \tag{3.21}$$

Since $\mathbf{n}$ is a unit normal vector to Γ and $\boldsymbol{\xi}$ is tangent to Γ we have that $\boldsymbol{\xi} \cdot \mathbf{n} = 0$ and $|\mathbf{n}| = 1$. We can further assume, without loss of generality, that $|\boldsymbol{\xi}| = 1$. Hence from (3.20) we have that:

$$\det(\mathcal{L}_1^p(\boldsymbol{\eta})) = -|\boldsymbol{\eta}|^2 = -(|\boldsymbol{\xi}|^2 + 2\tau\boldsymbol{\xi} \cdot \mathbf{n} + \tau^2|\mathbf{n}|^2) = -(1 + \tau^2),$$

which has roots $\pm i$, hence $\tau_1^+(\boldsymbol{\xi}) = i$, and $M^+(\boldsymbol{\xi}, \tau) = (\tau - i)$. Since the matrix

(3.21) only has one row the complementing condition reduces to showing that:

$$\begin{aligned} c &= (\tau - i)p_1(\tau), \\ -c\eta_1 &= (\tau - i)p_2(\tau), \\ -c\eta_2 &= (\tau - i)p_3(\tau), \end{aligned}$$

is only satisfied for $c = 0$ where $p_1(\tau), p_2(\tau), p_3(\tau)$ are some complex polynomials in τ. Indeed, this is trivially the case with $p_1(\tau) = p_2(\tau) = p_3(\tau) = 0$. Hence the complementing condition between $\mathcal{R}$ and $\mathcal{L}_1^p$ is satisfied. Therefore, if we put the corresponding choice of indices into the coercivity estimate (3.10) from Theorem 5, where we impose the boundary conditions strongly, we obtain the following coercivity estimate:

$$\|\phi\|_{q+2} + \|\mathbf{u}\|_{q+1} \leq C_q(\|\nabla \cdot \mathbf{u}\|_q + \|\nabla\phi + \mathbf{u}\|_{q+1}),$$

for some constant $C_q > 0$. This estimate holds for all $q \geq 0$ but, since this system is non-homogeneous elliptic, the required differentiability on the involved function spaces is impractical for all $q \geq 0$. However, Bochev and Gunzburger [25] proved that the estimate can be extended to all $q \in \mathbb{R}$. This result enables us to choose $q = -1$ yielding the following estimate:

$$\|\phi\|_1 + \|\mathbf{u}\|_0 \leq C_{-1}(\|\nabla \cdot \mathbf{u}\|_{-1} + \|\nabla\phi + \mathbf{u}\|_0). \tag{3.22}$$

This has overcome the practical issue of differentiability but has introduced a new problem in the form of the negative index norm.

3.4.2 Dealing with the Negative Index Norm

The difficulties in using negative index norms (as well as trace norms) were stated in Section 3.2.3. As we mentioned in this section we deal with such norms by using an approximation of their discrete norm generating operator. The norm generating operator for the minus one norm is given by $\mathcal{S}_Y = (-\Delta)^{-1/2}$ where $(-\Delta)^{-1}$ denotes the inverse operator of the Poisson equation with homogeneous Dirichlet boundary conditions under the additional assumption that Ω is a bounded domain [25]. In other words we have that:

$$\|\boldsymbol{\psi}\|_{-1}^2 = \|(-\Delta)^{-1/2}\boldsymbol{\psi}\|_0^2, \quad \forall\boldsymbol{\psi} \in H^{-1}(\Omega). \tag{3.23}$$

We consider the following two simple approximations of the discrete norm generating operator $\mathcal{S}_Y^h$ (see e.g. [23]):

- The identity operator $\mathcal{S}_Y^h = \mathcal{I}$.

- A mesh parameter-scaled identity operator $\mathcal{S}_Y^h = h\mathcal{I}$.

The first of these approximations is equivalent to simply replacing the minus one norm by an L^2-norm yielding the following least-squares functional:

$$\mathcal{J}_1(\phi, \mathbf{u}) = \|\nabla \cdot \mathbf{u} - f\|_0^2 + \|\nabla\phi + \mathbf{u}\|_0^2,$$

whereas the second approximation yields the following weighted functional:

$$\mathcal{J}_2(\phi, \mathbf{u}) = h^2\|\nabla \cdot \mathbf{u} - f\|_0^2 + \|\nabla\phi + \mathbf{u}\|_0^2.$$

Both these approximations have the advantage that they are very simple to implement but they also both lead to discrete norms which are only quasi-equivalent. This means that it becomes unclear, as we refine our approximation space, how well the estimate (3.22) is preserved. For the first of these approximations this also means that we are unable to provide a proof of an optimal rate of convergence in h. For the second approximation optimal rates of convergence can still be proven using carefully constructed duality arguments [25]. However, an undesirable consequence is that it increases the condition number of the involved linear systems [25]. An additional disadvantage of both these approximations is that it is unclear how the rank of the PGD approximation affects the discrete norm-equivalence.

There is a third way of approximating the discrete norm generating operator that was first considered by Bramble et al. [27]. This involves considering the inner-product generating operator $\mathcal{S}_Y^2 = (-\Delta)^{-1}$ defined by:

$$\langle \boldsymbol{\psi}, \boldsymbol{\phi}\rangle_{-1} = \langle (-\Delta)^{-1/2}\boldsymbol{\psi}, (-\Delta)^{-1/2}\boldsymbol{\phi}\rangle_0 = \langle (-\Delta)^{-1}\boldsymbol{\psi}, \boldsymbol{\phi}\rangle_0, \quad \forall \boldsymbol{\psi}, \boldsymbol{\phi} \in H^{-1}(\Omega).$$

One then uses the discrete approximation $(\mathcal{S}_Y^2)^h = h^2\mathcal{I} + \mathcal{K}^h$ where $\mathcal{K}^h$ is a spectrally equivalent approximation of the Galerkin solution operator for $-\Delta$ [25]. Note that, in the literature, it is often the case that this is stated as approximating the discrete norm generating operator by $\mathcal{S}_Y^h = h\mathcal{I} + (\mathcal{K}^h)^{1/2}$ (e.g. [23], [25]). This should be thought of as an abuse of notation since this approximation would actually introduce an additional unwanted term, $2h\langle (\mathcal{K}^h)^{1/2}(\nabla \cdot \mathbf{u} - f), \nabla \cdot \mathbf{u} - f\rangle_0$, into the least-squares functional. Unlike the other two methods, this approximation retains norm-equivalence in the discrete norms and hence optimal rates of convergence in h follow directly. This comes at the cost of being a more expensive approximation to implement in practice. The difficulty lies in calculating a suitable, self-adjoint, operator $\mathcal{K}^h$. As mentioned earlier this operator must be a spectrally equivalent approximation of the Galerkin solution operator for $-\Delta$. In other words, if we let $\mathcal{G}^h : H^{-1}(\Omega) \mapsto H_0^{1,h}(\Omega)$, where $\mathcal{G}^h\boldsymbol{\psi} = u^h$ if and only if

$$\langle \nabla u^h, \nabla v^h\rangle_0 = \langle \boldsymbol{\psi}, v^h\rangle_0 \quad \forall v^h \in H_0^{1,h}(\Omega).$$

Then we need to find an operator $\mathcal{K}^h$ that is spectrally equivalent to $\mathcal{G}^h$ i.e. there exists some constant $c > 0$ for which [25]:

$$c^{-1}\langle \mathcal{G}^h v^h, v^h \rangle_0 \leq \langle \mathcal{K}^h v^h, v^h \rangle_0 \leq c \langle \mathcal{G}^h v^h, v^h \rangle_0, \quad \forall v^h \in H_0^{1,h}(\Omega).$$

This is a property that is satisfied by any good preconditioner of the Poisson equation. This can be expensive to construct in standard implementations of least-squares methods but unfortunately the problem is even more prevalent for least-squares PGD algorithms. To see why this is the case consider the 2D domain $\Omega = \Omega_x \times \Omega_y$. We know from [37] that $H^1(\Omega) \neq H^1(\Omega_x) \otimes H^1(\Omega_y)$ and hence we also have the same property for its dual, $H^{-1}(\Omega) \neq H^{-1}(\Omega_x) \otimes H^{-1}(\Omega_y)$. This means that the Galerkin solution operator $\mathcal{G}^h$ cannot be expanded as a finite sum of tensorised operators, i.e. there exists no operators $\mathcal{G}^h_{x,j} : H^{-1}(\Omega_x) \mapsto H_0^{1,h}(\Omega_x)$ and $\mathcal{G}^h_{y,j} : H^{-1}(\Omega_y) \mapsto H_0^{1,h}(\Omega_y)$ such that $\mathcal{G}^h = \sum_{j=1}^{J} \mathcal{G}^h_{x,j} \otimes \mathcal{G}^h_{y,j}$. This important point was made by Cancès et al. [38] in the context of inverting Riesz operators for use in dual residual minimisation in the PGD. For our purpose it means that we cannot find suitable preconditioners $\mathcal{K}^h_{x,j}$ and $\mathcal{K}^h_{y,j}$, such that $\mathcal{K}^h = \sum_{j=1}^{J} \mathcal{K}^h_{x,j} \otimes \mathcal{K}^h_{y,j}$, for use in the alternating steps of our fixed point linearisation without a great deal of expense. For this reason we believe this third way of treating the negative index norm to be impractical in the context of the PGD and hence we shall not consider it further.

Returning our attention to the Div-Grad system, we are able to derive the Euler-Lagrange equations associated with the minimisation of the two quadratic least-squares functionals, $\mathcal{J}_k(\phi, \mathbf{u})$, $k = 1, 2$: Find $\boldsymbol{v} = (\phi, \mathbf{u})^T \in H_g^1(\Omega) \times H(\text{div})$ such that:

$$A_k(\boldsymbol{v}, \boldsymbol{v}^*) = L_k(\boldsymbol{v}^*), \quad \forall \boldsymbol{v}^* = (\phi^*, \mathbf{u}^*)^T \in H_0^1(\Omega) \times H(\text{div})$$

for $k = 1, 2$, where:

$$\begin{aligned} A_1(\boldsymbol{v}, \boldsymbol{v}^*) &= \langle \nabla \cdot \mathbf{u}, \nabla \cdot \mathbf{u}^* \rangle_0 + \langle \nabla \phi + \mathbf{u}, \nabla \phi^* + \mathbf{u}^* \rangle_0, \\ A_2(\boldsymbol{v}, \boldsymbol{v}^*) &= h^2 \langle \nabla \cdot \mathbf{u}, \nabla \cdot \mathbf{u}^* \rangle_0 + \langle \nabla \phi + \mathbf{u}, \nabla \phi^* + \mathbf{u}^* \rangle_0, \end{aligned}$$

and

$$L_1(\boldsymbol{v}) = \langle f, \nabla \cdot \mathbf{u}^* \rangle_0, \quad L_2(\boldsymbol{v}) = h^2 \langle f, \nabla \cdot \mathbf{u}^* \rangle_0,$$

Note that, while the dependent variable $\mathbf{u}$ has its energy measured in $(L^2(\Omega))^2$ it actually belongs to $H(\text{div})$ since the quantity $\nabla \cdot \mathbf{u}$ is not defined in $(L^2(\Omega))^2$.

3.4.3 Second-Order Formulation

In order for least-squares methods derived from the ADN Theory to be practical the problem must first be recast as a first-order system. However, a coercivity estimate, similar to that provided by Theorem 5, can be obtained by the following general result of Grisvard [74]:

Theorem 6. *Assume that Ω is a bounded convex polygon or polyhedron in $\mathbb{R}^2$ and $\mathbb{R}^3$, respectively. Then there exists a positive constant c such that:*

$$c\|\phi\|_1 \leq \|\Delta\phi\|_{-1} \quad \forall\phi \in H^1_0(\Omega). \tag{3.24}$$

Essentially this theorem states that the coercivity estimate arising from Theorem 5 for the standard second-order formulation of the Poisson equation (3.16) can be extended to $q = -1$. This coercivity estimate enables us to define a quadratic least-squares functional for (3.16). It is given by:

$$\mathcal{J}(\phi) = \|\Delta\phi + f\|^2_{-1}.$$

Furthermore, we do not have to treat this negative index norm in the same way as we did in the Div-Grad system. Indeed, the Euler-Lagrange equation for this functional is given by:

$$\langle \Delta\phi, \Delta\phi^* \rangle_{-1} = \langle -f, \Delta\phi^* \rangle_{-1} \quad \forall\phi^* \in H^1_0(\Omega),$$

and using the inner-product associated with the minus one norm generating operator we have that:

$$\langle \Delta\phi, \Delta\phi^* \rangle_{-1} = \langle (-\Delta)^{-1}\Delta\phi, \Delta\phi^* \rangle_0 = \langle -\phi, \Delta\phi^* \rangle_0 = \langle \nabla\phi, \nabla\phi^* \rangle_0.$$

Similarly we have:

$$\langle -f, \Delta\phi^* \rangle_{-1} = \langle f, \phi^* \rangle_0.$$

Therefore this second-order least-squares formulation of the Poisson equation reduces to solving: Find $\phi \in H^1_0(\Omega)$ such that:

$$\langle \nabla\phi, \nabla\phi^* \rangle_0 = \langle f, \phi^* \rangle_0 \quad \forall\phi^* \in H^1_0(\Omega),$$

which is exactly the Galerkin formulation of the same problem. This interesting result was noted by Bochev and Gunzburger [25].

Contrary to evidence that PGD algorithms based on least-squares formulations converge slower than their Galerkin counterparts [106], this result may imply that PGD algorithms based on least-squares formulations can converge as quickly as Galerkin PGD algorithms provided the least-squares method is based on a continuous least-squares estimate such as (3.24). As a result we now seek a first-order formulation of the Poisson equation which is homogeneous elliptic and hence completely practical to implement while retaining the continuous least-squares estimate. We will then compare rates of convergence between the non-homogeneous elliptic formulation (the Div-Grad system), the homogeneous elliptic system and the Galerkin formulation.

3.4.4 Extended Div-Grad System

The next first-order formulation of the Poisson equation that we shall consider is the extended Div-Grad system. This is essentially the same as the Div-Grad system (3.17)-(3.19) with the inclusion of an additional redundant equation and boundary condition [41]:

$$\nabla \cdot \mathbf{u} = f \quad \text{in} \;\; \Omega, \tag{3.25}$$

$$\mathbf{u} + \nabla \phi = 0 \quad \text{in} \;\; \Omega, \tag{3.26}$$

$$\nabla \times \mathbf{u} = 0 \quad \text{in} \;\; \Omega, \tag{3.27}$$

$$\phi = 0 \quad \text{on} \;\; \Gamma, \tag{3.28}$$

$$\mathbf{n} \times \mathbf{u} = 0 \quad \text{on} \;\; \Gamma, \tag{3.29}$$

where $\mathbf{n}$ denotes the outward unit normal to Γ. The additional boundary condition holds since from (3.26) we have that $\mathbf{n} \times \mathbf{u} = -\mathbf{n} \times \nabla\phi = 0$ since the boundary condition on ϕ implies that its tangential derivatives vanish on the boundary. Note that for simplicity of notation we have made the Dirichlet boundary condition on ϕ homogeneous. For the non-homogeneous case, $\phi = g$ on Γ, the additional boundary condition (3.29) should be replaced by $\mathbf{n}\times\mathbf{u} = -\mathbf{n}\times\nabla g$.

The additional equation (3.27) is derived by taking the curl of (3.26) and using the identity $\nabla \times \nabla\phi = 0$. Note that the curl operator is only defined in $\mathbb{R}^2$ and $\mathbb{R}^3$. For higher-dimensional Poisson equations one should instead consider the exterior derivative d_{k+1} in the differential de Rham complex [10], where $d_k = \nabla$, so that $\mathrm{Ker}(d_{k+1}) = \mathrm{Im}(\nabla)$ and hence $d_{k+1}\mathbf{u} = -d_{k+1}\nabla\phi = 0$.

The system (3.25)-(3.29) now has more equations than unknowns. As a result we cannot apply the ADN theory in its current state. Hence we must introduce a slack variable, ψ. In 2D we include the slack variable into the system in the following way:

$$\nabla \cdot \mathbf{u} = f \quad \text{in} \;\; \Omega, \tag{3.30}$$

$$\mathbf{u} + \nabla \phi + \nabla^{\perp}\psi = 0 \quad \text{in} \;\; \Omega, \tag{3.31}$$

$$\nabla \times \mathbf{u} = 0 \quad \text{in} \;\; \Omega, \tag{3.32}$$

$$\phi = 0 \quad \text{on} \;\; \Gamma, \tag{3.33}$$

$$\mathbf{n} \times \mathbf{u} = 0 \quad \text{on} \;\; \Gamma. \tag{3.34}$$

Note that in 2D we have two curl operators: $\nabla^{\perp} = (-\frac{\partial}{\partial y}, \frac{\partial}{\partial x})^T$, which takes scalars into vectors and $\nabla \times \mathbf{u} = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}$, which takes vectors into scalars.

The reason for the inclusion of a slack variable in this way is to generate a homogeneous elliptic system and so that the slack variable has exact solution $\psi = 0$ and hence can be removed from the system. Indeed, taking the curl of (3.31) and using the vector identities: $\nabla \times \nabla\phi = 0$ and $\nabla \times \nabla^{\perp}\psi = \nabla^2\psi$ together with equation (3.32) we obtain that the slack variable satisfies the Laplace equation:

$$\nabla^2\psi = 0 \quad \text{in } \Omega.$$

If we then take the cross product of (3.31) with $\mathbf{n}$ and use the additional boundary condition (3.29) then we have $\mathbf{n} \times \nabla^{\perp}\psi = 0$ on Γ. This represents a Neumann type boundary condition on the slack variable and since we know that ψ satisfies the Laplace equation then we obtain the exact solution $\psi = c$ for some constant c. In practice we will remove the non-uniqueness of this solution, enforcing $\psi = 0$, by simply removing it from the system. Note that the reason we did not simply impose a boundary condition on the slack variable directly is that this would lead to a boundary operator that does not satisfy the complementing condition with the homogeneous elliptic principal part.

If we let $\boldsymbol{v} = (\phi, \psi, u, v)^T$, this then leads to the following representation of the 2D extended Div-Grad operator and boundary operator:

$$\boldsymbol{\mathcal{L}}\boldsymbol{v} = \begin{pmatrix} 0 & 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ 0 & 0 & -\frac{\partial}{\partial y} & \frac{\partial}{\partial x} \\ \frac{\partial}{\partial x} & -\frac{\partial}{\partial y} & 1 & 0 \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 & 1 \end{pmatrix} \begin{pmatrix} \phi \\ \psi \\ u \\ v \end{pmatrix}, \quad \boldsymbol{\mathcal{R}}\boldsymbol{v} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -n_2 & n_1 \end{pmatrix} \begin{pmatrix} \phi \\ \psi \\ u \\ v \end{pmatrix}.$$

As we had hoped this leads to a homogeneous elliptic system such that we have ADN ellipticity for the choice of indices $s_i = \{0, 0, 0, 0\}$ and $t_j = \{1, 1, 1, 1\}$ yielding the following principal part:

$$\boldsymbol{\mathcal{L}}^p = \begin{pmatrix} 0 & 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ 0 & 0 & -\frac{\partial}{\partial y} & \frac{\partial}{\partial x} \\ \frac{\partial}{\partial x} & -\frac{\partial}{\partial y} & 0 & 0 \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 & 0 \end{pmatrix}.$$

Indeed, the three requirements for ADN ellipticity in Definition 1 are satisfied. Requirements (i) and (ii) are trivially satisfied since $s_i + t_j = 1$ for all $i, j = 1, \ldots, 4$. We also have that (iii) is satisfied since:

$$\det(\boldsymbol{\mathcal{L}}^p(\boldsymbol{\xi})) = \begin{vmatrix} 0 & 0 & \xi_1 & \xi_2 \\ 0 & 0 & -\xi_2 & \xi_1 \\ \xi_1 & -\xi_2 & 0 & 0 \\ \xi_2 & \xi_1 & 0 & 0 \end{vmatrix} = (\xi_1^2 + \xi_2^2)^2 = |\boldsymbol{\xi}|^4. \tag{3.35}$$

This tells us that $\boldsymbol{\mathcal{L}}^p$ is uniformly ADN elliptic of order four which means we must impose $m = 2$ boundary conditions. Indeed, this explains why we needed to include the additional boundary condition (3.29). It is also clear from (3.35) and Lemma 4 that this also satisfies the supplementary condition.

To verify the complementing condition we begin by defining the principal part of the boundary operator with respect to the boundary indices $r_l = \{-1, -1\}$. This yields a principal part such that $\mathcal{R}^p = \mathcal{R}$. Given the vector, $\boldsymbol{\eta} = \boldsymbol{\xi} + \tau\mathbf{n} = (\eta_1, \eta_2)^T$, one can readily calculate the adjoint of $\mathcal{L}^p(\boldsymbol{\eta})$ by:

$$\mathcal{L}'(\boldsymbol{\eta}) = |\boldsymbol{\eta}|^2 \begin{pmatrix} 0 & 0 & \eta_1 & \eta_2 \\ 0 & 0 & -\eta_2 & \eta_1 \\ \eta_1 & -\eta_2 & 0 & 0 \\ \eta_2 & \eta_1 & 0 & 0 \end{pmatrix}.$$

Hence we have that:

$$\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}'(\boldsymbol{\eta}) = |\boldsymbol{\eta}|^2 \begin{pmatrix} 0 & 0 & \eta_1 & \eta_2 \\ -n_2\eta_1 + n_1\eta_2 & n_2\eta_2 + n_1\eta_1 & 0 & 0 \end{pmatrix}. \tag{3.36}$$

Since $\mathbf{n}$ is a unit normal vector to Γ and $\boldsymbol{\xi}$ is tangent to Γ we have that $\boldsymbol{\xi} \cdot \mathbf{n} = 0$ and $|\mathbf{n}| = 1$. We can further assume, without loss of generality, that $|\boldsymbol{\xi}| = 1$. Hence from (3.35) we have that:

$$\det(\mathcal{L}^p(\boldsymbol{\eta})) = |\boldsymbol{\eta}|^4 = (|\boldsymbol{\xi}|^2 + 2\tau\boldsymbol{\xi} \cdot \mathbf{n} + \tau^2|\mathbf{n}|^2)^2 = (1 + \tau^2)^2,$$

which has roots of multiplicity two: $\pm i$. Hence $\tau_1^+(\boldsymbol{\xi}) = \tau_2^+(\boldsymbol{\xi}) = i$ and $M^+(\boldsymbol{\xi}, \tau) = (\tau - i)^2$. Then the complementing condition holds if the rows of (3.36) are linearly independent modulo M^+. In other words we require that:

$$\begin{aligned} c_2(1 + \tau^2)(-n_2\eta_1 + n_1\eta_2) &= (\tau - i)^2 p_1(\tau), \\ c_2(1 + \tau^2)(n_2\eta_2 + n_1\eta_1) &= (\tau - i)^2 p_2(\tau), \\ c_1(1 + \tau^2)\eta_1 &= (\tau - i)^2 p_3(\tau), \\ c_1(1 + \tau^2)\eta_2 &= (\tau - i)^2 p_4(\tau), \end{aligned}$$

only holds when $c_1 = c_2 = 0$, where $p_1(\tau), \ldots, p_4(\tau)$ are some complex polynomials in τ. To simplify this we can use the fact that $(1 + \tau^2) = (\tau + i)(\tau - i)$ to cancel a $(\tau - i)$ term from both sides. We can further assume, without loss of generality, that the coordinate axes are aligned with the directions of $\boldsymbol{\xi}$ and $\mathbf{n}$ such that $\boldsymbol{\xi} = (1, 0)^T$,

$\mathbf{n} = (0,1)^T$ and $\boldsymbol{\eta} = (1,\tau)^T$. This yields:

$$\begin{aligned} -c_2(\tau + i) &= (\tau - i)p_1(\tau), \\ c_2\tau(\tau + i) &= (\tau - i)p_2(\tau), \\ c_1(\tau + i) &= (\tau - i)p_3(\tau), \\ c_1\tau(\tau + i) &= (\tau - i)p_4(\tau). \end{aligned}$$

This is trivially only satisfied with $c_1 = c_2 = 0$ and $p_1(\tau) = \cdots = p_4(\tau) = 0$. Hence the complementing condition between $\mathcal{R}$ and $\mathcal{L}^p$ is satisfied. Therefore, if we put the corresponding choice of indices into the coercivity estimate (3.10) from Theorem 5 (with $q = 0$), where we impose the boundary conditions strongly, we obtain the following coercivity estimate:

$$\|\phi\|_1 + \|\psi\|_1 + \|\mathbf{u}\|_1 \leq C_0(\|\nabla \cdot \mathbf{u}\|_0 + \|\mathbf{u} + \nabla\phi + \nabla^\perp\psi\|_0 + \|\nabla \times \mathbf{u}\|_0 + |\ell(\psi)|),$$

for some constant $C_0 > 0$. Note that the $|\ell(\psi)|$ term needs to be included here due to ψ being unique only up to an additive constant. We can remove the slack variable from the system by enforcing $\psi \equiv 0$. Then we obtain the following estimate:

$$\|\phi\|_1 + \|\mathbf{u}\|_1 \leq C_0(\|\nabla \cdot \mathbf{u}\|_0 + \|\mathbf{u} + \nabla\phi\|_0 + \|\nabla \times \mathbf{u}\|_0).$$

This leads to the following quadratic least-squares functional:

$$\mathcal{J}_3(\phi, \mathbf{u}) = \|\nabla \cdot \mathbf{u} - f\|_0^2 + \|\mathbf{u} + \nabla\phi\|_0^2 + \|\nabla \times \mathbf{u}\|_0^2. \tag{3.37}$$

We are then able to derive the Euler-Lagrange equations associated with the minimisation of (3.37): Find $\boldsymbol{v} = (\phi, \mathbf{u})^T \in H_0^1(\Omega) \times \mathbf{H}_\times^1(\Omega)$ such that:

$$A_3(\boldsymbol{v}, \boldsymbol{v}^*) = L_3(\boldsymbol{v}^*), \quad \forall \boldsymbol{v}^* = (\phi^*, \mathbf{u}^*)^T \in H_0^1(\Omega) \times \mathbf{H}_\times^1(\Omega),$$

where:

$$\mathbf{H}_\times^1(\Omega) = \{\mathbf{u} \in H(\mathrm{div}) \cap H(\mathrm{curl}) \mid \mathbf{n} \times \mathbf{u} = 0 \quad \text{on} \quad \Gamma\},$$

and where:

$$\begin{aligned} A_3(\boldsymbol{v}, \boldsymbol{v}^*) &= \langle \nabla \cdot \mathbf{u}, \nabla \cdot \mathbf{u}^* \rangle_0 + \langle \mathbf{u} + \nabla\phi, \mathbf{u}^* + \nabla\phi^* \rangle_0 + \langle \nabla \times \mathbf{u}, \nabla \times \mathbf{u}^* \rangle_0, \\ L_3(\boldsymbol{v}^*) &= \langle f, \nabla \cdot \mathbf{u}^* \rangle_0. \end{aligned}$$

In summary, we have considered four different least-squares methods for the Poisson equation. Firstly, we obtained two least-squares methods from the non-homogeneous elliptic Div-Grad system which are associated with minimisation of the functionals $\mathcal{J}_1(\phi, \mathbf{u})$ and $\mathcal{J}_2(\phi, \mathbf{u})$, the difference between the two methods being the choice of weighting of the norms. We then considered a least-squares method for the second-order formulation which turned out to be equivalent to the Galerkin formulation of the same problem. Finally, we obtained a least-squares method based on the homogeneous elliptic extended Div-Grad system associated with the minimisation of the functional $\mathcal{J}_3(\phi, \mathbf{u})$ given above. We shall now explain how these methods are implemented into the PGD framework and then compare the different methods using some numerical examples.

3.4.5 Implementation into the PGD

Consider a first-order formulation of the 2D Poisson equation on the rectangular domain $\Omega = [a, b] \times [c, d]$. In the PGD algorithm we seek a reduced basis separated representation of the dependent variables:

$$u(x, y) \approx \sum_{j=1}^{J} X_j^u(x) Y_j^u(y) =: u_J(x, y), \quad v(x, y) \approx \sum_{j=1}^{J} X_j^v(x) Y_j^v(y) =: v_J(x, y),$$

$$\phi(x, y) \approx \sum_{j=1}^{J} X_j^\phi(x) Y_j^\phi(y) =: \phi_J(x, y).$$

Least-squares methods are most commonly applied in conjunction with a finite element discretization however we shall continue to use spectral element methods which have been applied to least-squares formulations in the works of Proot and Gerritsma [111–113]. To this end we divide $[a, b]$ into K_x elements, $[a_{k_x-1}, a_{k_x}]$, $k_x = 1, \ldots, K_x$, and divide $[c, d]$ into K_y elements, $[c_{k_y-1}, c_{k_y}]$, $k_y = 1, \ldots, K_y$. The PGD basis functions $X_j^{(\bullet)}(x)$, $Y_j^{(\bullet)}(y)$ for $(\bullet) = \{\phi, u, v\}$ are then piecewise polynomials given by:

$$X_j^{(\bullet)}(x) = \begin{cases} \displaystyle\sum_{i=0}^{N} \alpha_{j,i,k_x}^{(\bullet)} h_{i,k_x}(x), & \text{if} \quad x \in [a_{k_x-1}, a_{k_x}], \\ 0, & \text{otherwise}, \end{cases}$$

$$Y_j^{(\bullet)}(y) = \begin{cases} \displaystyle\sum_{i=0}^{N} \beta_{j,i,k_y}^{(\bullet)} h_{i,k_y}(y), & \text{if} \quad y \in [c_{k_y-1}, c_{k_y}], \\ 0, & \text{otherwise}, \end{cases}$$

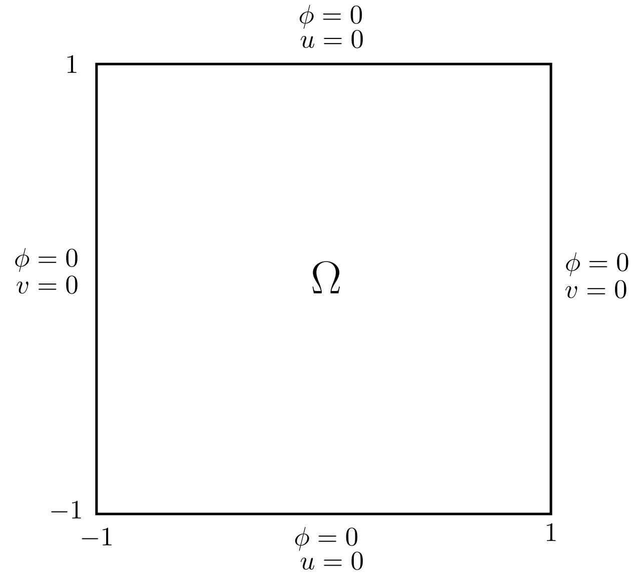
where $h_{i,k}$, $i = 0, \ldots, N$ are the Legendre interpolating polynomials on the k^{th} element.

The algorithm then proceeds in much the same way as for the Galerkin progressive PGD except that at the step where the enrichment couples $r^{(\bullet)}(x)$ and $s^{(\bullet)}(y)$ are calculated we no longer impose Galerkin orthogonality and instead employ the Euler-Lagrange equation associated with our chosen quadratic least-squares functional. This still leads to a nonlinear system in r and s which we again solve via an alternating directions fixed point algorithm.

3.4.6 Numerical Results

Example 5 (Infinite Rank Solution)**.**

Consider the Poisson equation (3.16) on the following domain:



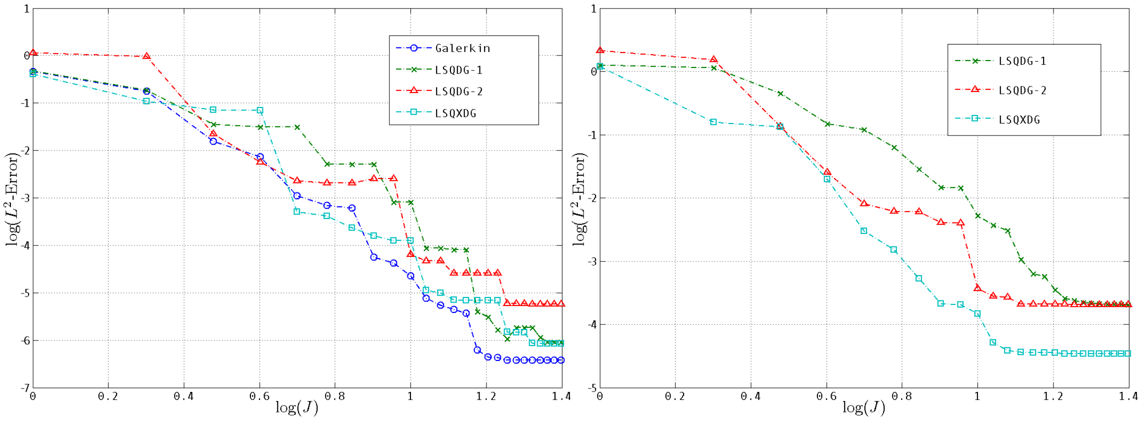
with source term

$$\begin{aligned} f(x,y) =& 4\pi^2(x^2(1-y^2)^2 + y^2(1-x^2)^2)\sin(\pi(1-x^2)(1-y^2)) \\ &+ 2\pi((1-x^2)+(1-y^2))\cos(\pi(1-x^2)(1-y^2)), \end{aligned}$$

where the boundary conditions on $\mathbf{u} = (u,v)^T$ are only relevant for the extended Div-Grad formulation. This problem has the exact solution $\phi = \sin(\pi(1-x^2)(1-y^2))$ for the primary dependent variable ϕ. This solution does not have a finite rank separated representation and hence we expect monotonic convergence as we increase the rank of our approximation.

Figure 3.1 shows the convergence in the rank, J, of the PGD approximation of ϕ for the Galerkin, the two least-squares Div-Grad (LSQDG-1 & LSQDG-2), and the least-squares extended Div-Grad (LSQXDG) PGD algorithms. The discretisation used was a spectral element method with degree $N = 8$ polynomials on $K_x = K_y = 3$ elements in each coordinate direction. This plot shows that even the least-squares methods based on the non-homogeneous elliptic Div-Grad system are converging at a rate that is competitive with that of the Galerkin PGD. However, the extended Div-Grad system does appear to be the best of the least-squares PGD algorithms with a rate of convergence very similar to the Galerkin PGD.

Figure 3.1(a) does not present a clear winner in terms of convergence in the primary dependent variable ϕ. However, we also note that ϕ does not appear in the



(a) Error in ϕ

(b) Error in $\mathbf{u}$

Figure 3.1: Comparison of Least-Squares and Galerkin PGDs for Example 5

problematic minus one norm in the continuous least-squares estimate (3.22) for the Div-Grad system. For this reason, in Figure 3.1(b), we have plotted convergence of the approximation for $\mathbf{u}$. Note that the Galerkin PGD has been left out of this comparison since $\mathbf{u}$ does not appear in the second order formulation (3.16).

Figure 3.1(b) gives a clearer picture that the least-squares PGD algorithms based on the non-homogeneous elliptic Div-Grad system converge slower than the least-squares PGD algorithm based on the homogeneous elliptic extended Div-Grad system. It further indicates that the Div-Grad system which uses a mesh-parameter weighted L^2-norm (LSQDG-2) converges faster than the non weighted L^2-norm approach (LSQDG-1).

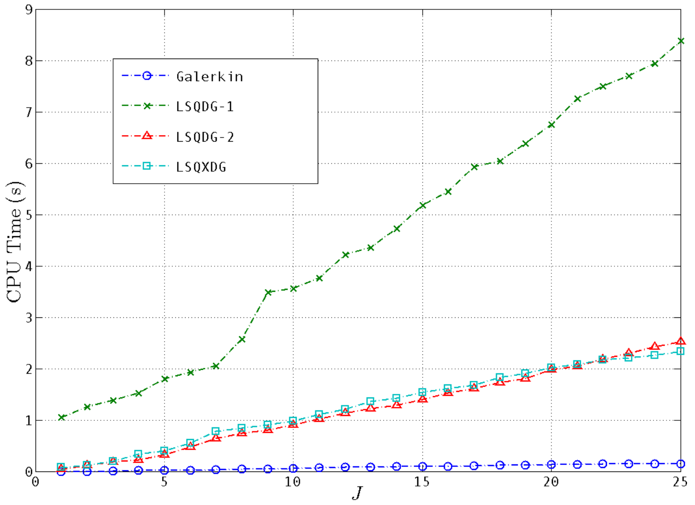


Figure 3.2: CPU Times for Example 5

Finally, in Figure 3.2, we have compared CPU times for each of the four PGD algorithms as we increase the rank of our approximation. This clearly shows that LSQDG-1 is much more expensive than the other methods. The reason for this difference is that LSQDG-1 takes longer to converge in the alternating directions fixed point algorithm. Note that LSQDG-2 and LSQXDG perform almost exactly the same in CPU time.

Example 6 (Rank-1 Solution)**.**

We now consider an example defined on the same domain as Example 5 and again with homogeneous boundary conditions but with source term:

$$f(x, y) = 2\pi^2 \sin(\pi x) \sin(\pi y).$$

This problem has the exact solution $\phi = \sin(\pi x) \sin(\pi y)$ for the primary dependent variable ϕ. This solution clearly has a rank-1 separated representation and hence we would hope that our PGD algorithms are able to converge in a single iteration. Indeed, we found that all four algorithms were able to converge in a single iteration. For this reason we do not present the convergence results as a plot but instead by the table given in Figure 3.3. The errors listed here are the errors attained after the first iteration (and hence every subsequent iteration).

		Algorithm			
		Galerkin	**LSQDG-1**	**LSQDG-2**	**LSQXDG**
L^2-Error	ϕ	8.005e-10	4.293e-10	1.401e-08	3.141e-09
	$\mathbf{u}$	N/A	7.186e-07	4.458e-07	1.686e-08

Figure 3.3: Comparison of Least-Squares and Galerkin PGDs for Example 6

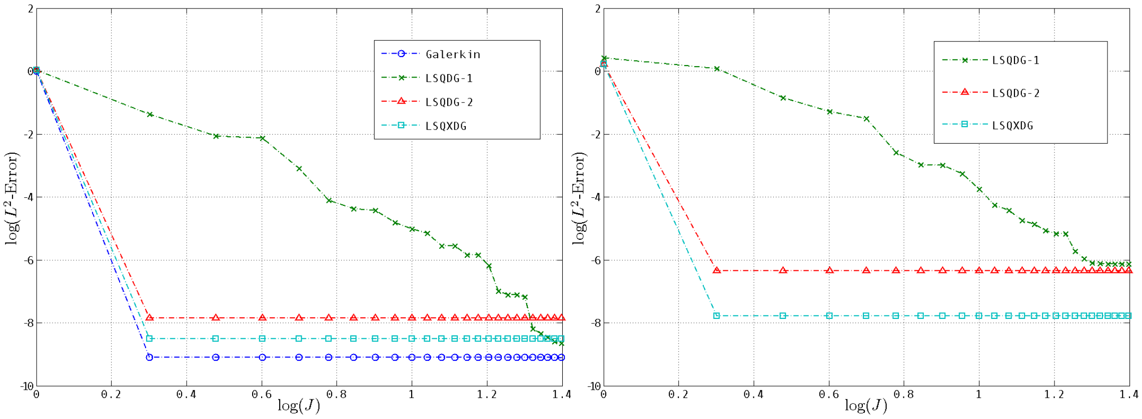
Curiously LSQDG-1 yields the lowest error for ϕ in this particular case, beating even the Galerkin solution, however the same can not be said for the error in $\mathbf{u}$ where the LSQXDG algorithm is clearly superior. All of the algorithms run very quickly for obvious reasons and hence we do not compare CPU times for this example.

Example 7 (Rank-2 Solution)**.**

To further test our algorithms are performing as expected we consider an example defined on the same domain as Examples 1 and 2 with homogeneous boundary conditions and with source term:

$$f(x, y) = 2\pi^2 \sin(\pi x) \sin(\pi y) + 2(2 - x^2 - y^2).$$

This problem has the exact solution $\phi = \sin(\pi x) \sin(\pi y) + (1 - x^2)(1 - y^2)$ for the primary dependent variable ϕ. This solution has a rank-2 separated representation and hence we would hope our PGD algorithms are able to converge after two iterations. In Figure 3.4 we see that all the algorithms besides LSQDG-1 converge in two iterations as expected. This indicates that this algorithm does not sufficiently represent the continuous problem. In other words due to the Div-Grad system being non-homogeneous elliptic we find that, in the LSQDG-1 algorithm, the continuous



(a) Error in ϕ

(b) Error in $\mathbf{u}$

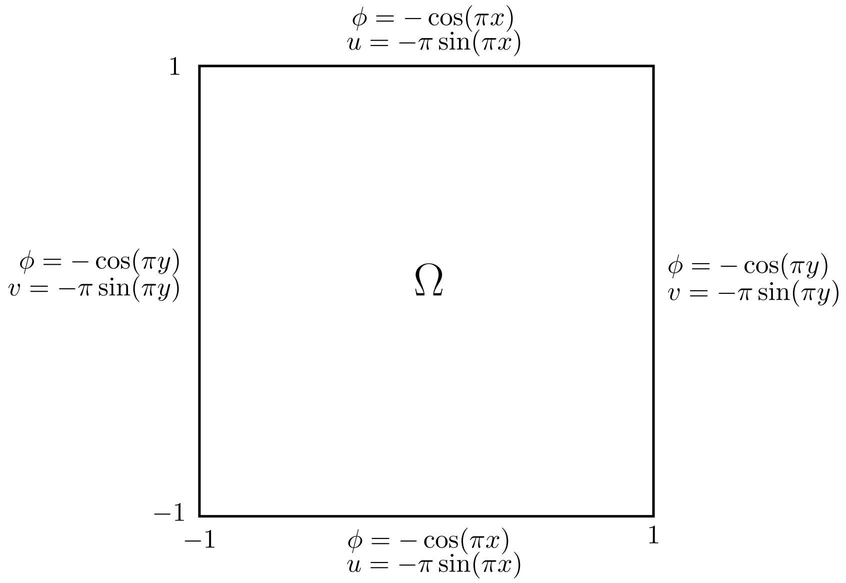
Figure 3.4: Comparison of Least-Squares and Galerkin PGDs for Example 7

least-squares estimate (3.22) has not been sufficiently preserved.

On the other hand, we find that LSQDG-2 is still able to converge in two iterations. This highlights the significance of using weighted L^2-norms for non-homogeneous elliptic systems. However, as in the previous examples, we again find that the LSQXDG algorithm displays superior convergence behaviour. All the algorithms bar LSQDG-1 run very quickly, hence we do not compare CPU times for this example.

Example 8 (Rank-2 Solution with Non-Homogeneous Boundary Conditions)**.**

In this final example we consider the Poisson equation on the following domain:

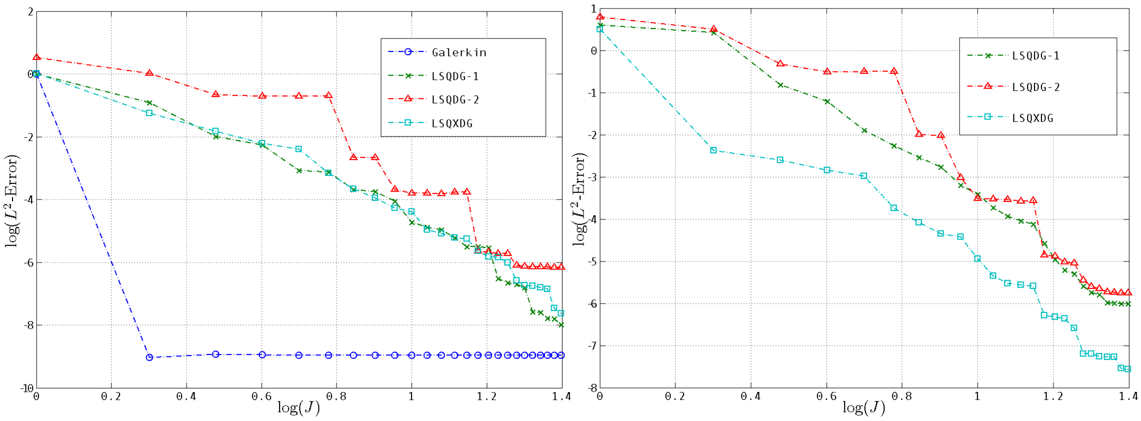


with source term:

$$f(x, y) = 2\pi^2(\sin(\pi x)\sin(\pi y) + \cos(\pi x)\cos(\pi y)),$$

where the boundary conditions on $\mathbf{u} = (u, v)^T$ are only relevant for the extended Div-Grad formulation. This problem has the exact solution $\phi = \sin(\pi x)\sin(\pi y) +$

$\cos(\pi x)\cos(\pi y)$ for the primary dependent variable ϕ. This solution has a rank-2 separated representation; however, we can no longer expect our algorithms to converge in two iterations. This is because the way we impose non-homogeneous boundary conditions in the PGD means that we start with, at most, a rank-4 separated representation in the form of the transfinite interpolating function.



(a) Error in ϕ

(b) Error in $\mathbf{u}$

Figure 3.5: Comparison of Least-Squares and Galerkin PGDs for Example 8

Figure 3.5 shows the convergence of the algorithms. We find that the Galerkin algorithm is able to converge in two iterations despite the non-homogeneous boundary conditions whereas the other three algorithms fail to do so. The fact that the Galerkin algorithm was still able to converge in two iterations can be explained if we inspect the transfinite interpolating function we obtain for these boundary conditions, it is given by:

$$T(x, y) = -(1 + \cos(\pi x) + \cos(\pi y)).$$

In order to converge in two iterations we would need to negate $T(x, y)$ while adding the correct modes, $\sin(\pi x)\sin(\pi y)$ and $\cos(\pi x)\cos(\pi y)$, so that we look for a solution of the form:

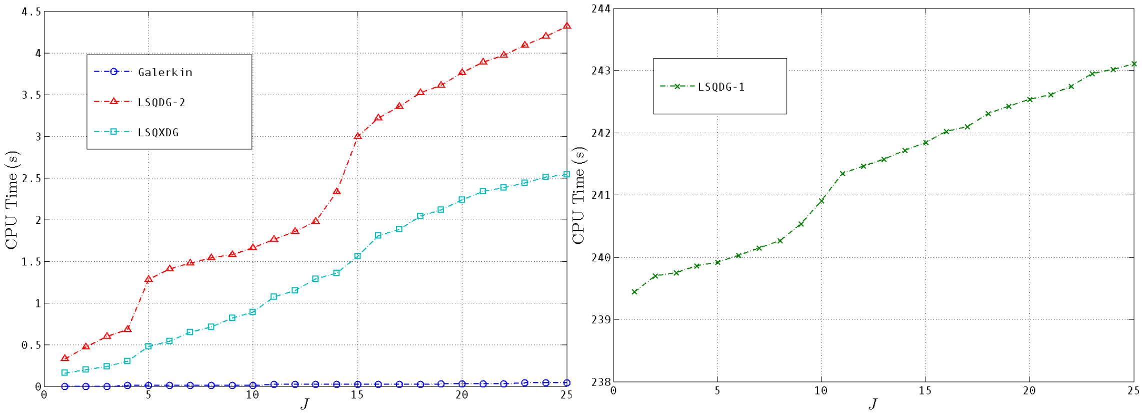
$$\phi(x, y) - T(x, y) = \sin(\pi x)\sin(\pi y) + \cos(\pi x)\cos(\pi y) + (1 + \cos(\pi x) + \cos(\pi y)),$$

one then notes that this can be factorised to yield:

$$\phi(x, y) - T(x, y) = \sin(\pi x)\sin(\pi y) + (\cos(\pi x) + 1)(\cos(\pi y) + 1).$$

This then has a rank-2 separated form where both modes have homogeneous boundary conditions. This explains how the Galerkin PGD algorithm was still able to converge in two iterations. However, the same is not true for the LSQXDG algorithm since we also need to construct transfinite interpolating functions for the additional boundary condition on $\mathbf{u}$. This explains why the LSQXDG algorithm, in this case, is unable to capture the rank-2 nature of the solution.

However, this is not true for the LSQDG-1 and LSQDG-2 algorithms in which the additional boundary condition does not appear and so one would still expect it to capture the rank-2 nature of the solution. This is again an indicator that these algorithms do not sufficiently preserve the continuous least-squares estimate (3.22). As for rates of convergence of the least-squares algorithms, we again note that the homogeneous elliptic LSQXDG algorithm provides superior rates of convergence.



(a) Without LSQDG-1

(b) LSQDG-1

Figure 3.6: CPU Time for Example 8

In Figure 3.6 we have plotted the CPU times for all four algorithms. The first thing to point out is that we have plotted the CPU time of the LSQDG-1 algorithm on a separate axis. The reason for this is that this algorithm takes considerably longer for the first iteration. Indeed, we find that the first iteration of the LSQDG-1 algorithm takes almost 4 minutes whereas the other three algorithms take less than half a second. The subsequent 24 iterations then only take around 3 seconds altogether. The fact that this algorithm seriously struggles to get started in the PGD iteration indicates a serious flaw with this formulation which is most likely caused by poorly conditioned systems. As for the other three algorithms we find that, of the two least-squares algorithms, the one based on the homogeneous elliptic extended Div-Grad system is the quickest algorithm. This differs to Example 5 in which we observed very similar CPU times for both these algorithms. Of course the fastest algorithm is again the Galerkin algorithm since it involves smaller linear systems and converges in just two iterations.

3.4.7 Conclusions

Throughout these examples the superior least-squares algorithm is consistently the one based on the homogeneous elliptic extended Div-Grad system. It always performs as expected in terms of capturing the natural rank of solutions and displays superior levels of convergence, particularly for the additional dependent variable $\mathbf{u}$. It may not seem useful to have good convergence in the variable $\mathbf{u}$

but it is often the case in applications that the additional dependent variables have important physical meaning. It is also the case that the primary variables can appear in the negative index norms in the continuous least-squares estimates. An example of this which we shall see later is the pressure in the Stokes problem. Therefore it is important to have the best rate of convergence for all dependent variables. We also noted in examples 1 and 4 that LSQXDG was also the fastest of the least-squares algorithms in terms of CPU time.

On the other hand, the algorithms based on the non-homogeneous elliptic Div-Grad system generally performed quite poorly. We noticed a few examples where they failed to capture the natural rank of solutions as well as poor rates of convergence for $\mathbf{u}$ in the infinite rank case. As far as CPU time is concerned, in example 1, we noticed that LSQDG-1 was significantly slower, and in example 4, we found that LSQDG-1 was extremely slow for the first iteration while LSQDG-2 was also now slower than LSQXDG. We believe the inferiority of these algorithms can be explained by the non-homogeneous ellipticity of the Div-Grad system. Indeed, we believe that the continuous least-squares estimate (3.22) is not sufficiently preserved in the context of PGD by these two algorithms. We also noticed that using the weighted L^2-norm in LSQDG-2, in general, was a significant improvement on the unweighted case. However, it was not significant enough to improve performance beyond that of LSQXDG.

In conclusion, to construct efficient and reliable least-squares PGD algorithms, homogeneous ellipticity of the underlying system appears to be a key factor. This is not the case in standard implementations of least-squares methods where non-homogeneous elliptic systems are often preferred for their simplicity. However, we have found significant evidence that they perform poorly within the PGD framework.

However, the best algorithm we have observed in these examples is consistently the Galerkin PGD algorithm. It always yields the best rates of convergence, which were about the same as for the LSQXDG algorithm contrary to the observations of Nouy [106]. More significantly it was always by far the fastest algorithm since it involved the solution of much smaller linear systems. In the following two sections we shall consider problems which do not possess natural energy minimisation principles namely the convection-diffusion equation and the Stokes equations. In these cases the proof of convergence no longer holds for Galerkin PGDs and hence we can no longer guarantee convergence. Furthermore, for the Stokes equations, we obtain LBB stability issues when using a Galerkin PGD algorithm. For these reasons we instead consider efficient least-squares PGD algorithms based on the results we found for the Poisson equation.

3.5 Least-Squares Formulation of the Convection-Diffusion Equation

Consider the following linear convection-diffusion equation:

$$\begin{aligned} -\nabla^2\phi + \mathbf{b}\cdot\nabla\phi &= f \quad \text{in } \Omega, \\ \phi &= g \quad \text{on } \Gamma. \end{aligned} \tag{3.38}$$

where for simplicity we assume $\mathbf{b} = (b_1, b_2)^T$ is some constant vector. We consider this equation due to its similarity to the Poisson equation which means that a lot of the least-squares theory can be taken directly from the previous section. It is also an interesting problem since it has no natural energy minimisation principle and furthermore it is a non-symmetric problem. PGD algorithms for problems of this type have recently been considered by Cancès et al. [38].

3.5.1 Div-Grad System

As with the Poisson equation the simplest first-order reformulation of the convection-diffusion equation is derived by introducing the vector $\mathbf{u} = -\nabla\phi$ yielding the following Div-Grad system:

$$\nabla\cdot\mathbf{u} + \mathbf{b}\cdot\nabla\phi = f \quad \text{in } \Omega, \tag{3.39}$$

$$\mathbf{u} + \nabla\phi = 0 \quad \text{in } \Omega, \tag{3.40}$$

$$\phi = g \quad \text{on } \Gamma. \tag{3.41}$$

If we let $\boldsymbol{v} = (\phi, u, v)^T$, this leads to the following representation of the 2D Div-Grad convection-diffusion operator and boundary operator:

$$\mathcal{L}\boldsymbol{v} = \begin{pmatrix} b_1\frac{\partial}{\partial x} + b_2\frac{\partial}{\partial y} & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial x} & 1 & 0 \\ \frac{\partial}{\partial y} & 0 & 1 \end{pmatrix} \begin{pmatrix} \phi \\ u \\ v \end{pmatrix}, \quad \mathcal{R}\boldsymbol{v} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \phi \\ u \\ v \end{pmatrix}.$$

This system is ADN elliptic with the choice of indices $s_i = \{0, -1, -1\}$, $t_j = \{2, 1, 1\}$, yielding the following principal part:

$$\mathcal{L}_1^p = \begin{pmatrix} 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial x} & 1 & 0 \\ \frac{\partial}{\partial y} & 0 & 1 \end{pmatrix}.$$

This is exactly the same principal part as for the Div-Grad reformulation of the Poisson equation. Therefore all the related theory in the previous section holds and we know that this system satisfies the supplementary and complementing conditions yielding the following coercivity estimate:

$$\|\phi\|_1 + \|\mathbf{u}\|_0 \le C_{-1}(\|\nabla\cdot\mathbf{u} + \mathbf{b}\cdot\nabla\phi\|_{-1} + \|\nabla\phi + \mathbf{u}\|_0),$$

for some constant $C_{-1} > 0$, with the associated quadratic least-squares functionals:

$$\begin{aligned}\mathcal{J}_1(\phi,\mathbf{u}) &= \|\nabla\cdot\mathbf{u} + \mathbf{b}\cdot\nabla\phi - f\|_0^2 + \|\nabla\phi + \mathbf{u}\|_0^2,\\ \mathcal{J}_2(\phi,\mathbf{u}) &= h^2\|\nabla\cdot\mathbf{u} + \mathbf{b}\cdot\nabla\phi - f\|_0^2 + \|\nabla\phi + \mathbf{u}\|_0^2.\end{aligned}$$

We are then able to derive the Euler-Lagrange equation associated with the minimisation of the above least-squares functionals: Find $\boldsymbol{v} = (\phi,\mathbf{u})^T \in H_g^1(\Omega)\times H(\mathrm{div})$ such that:

$$A_k(\boldsymbol{v},\boldsymbol{v}^*) = L_k(\boldsymbol{v}^*),\quad \forall\boldsymbol{v}^* = (\phi^*,\mathbf{u}^*)^T \in H_0^1(\Omega)\times H(\mathrm{div})$$

for $k = 1, 2$, where:

$$\begin{aligned}A_1(\boldsymbol{v},\boldsymbol{v}^*) &= \langle\nabla\cdot\mathbf{u} + \mathbf{b}\cdot\nabla\phi, \nabla\cdot\mathbf{u}^* + \mathbf{b}\cdot\nabla\phi^*\rangle_0 + \langle\nabla\phi + \mathbf{u}, \nabla\phi^* + \mathbf{u}^*\rangle_0,\\ A_2(\boldsymbol{v},\boldsymbol{v}^*) &= h^2\langle\nabla\cdot\mathbf{u} + \mathbf{b}\cdot\nabla\phi, \nabla\cdot\mathbf{u}^* + \mathbf{b}\cdot\nabla\phi^*\rangle_0 + \langle\nabla\phi + \mathbf{u}, \nabla\phi^* + \mathbf{u}^*\rangle_0,\end{aligned}$$

and

$$L_1(\boldsymbol{v}) = \langle f, \nabla\cdot\mathbf{u}^* + \mathbf{b}\cdot\nabla\phi^*\rangle_0,\quad L_2(\boldsymbol{v}) = h^2\langle f, \nabla\cdot\mathbf{u}^* + \mathbf{b}\cdot\nabla\phi^*\rangle_0,$$

As was the case for the Poisson equation, this system is clearly not homogeneous elliptic. We now look at a homogeneous elliptic reformulation of the convection-diffusion equation which is again analogous to the Poisson equation.

3.5.2 Extended Div-Grad System

We extend the Div-Grad convection-diffusion formulation (3.39)-(3.41), in the same way as for the Poisson equation, by including the following additional redundant equation and boundary condition:

$$\nabla\cdot\mathbf{u} + \mathbf{b}\cdot\nabla\phi = f \quad \text{in } \Omega, \tag{3.42}$$

$$\mathbf{u} + \nabla\phi = 0 \quad \text{in } \Omega, \tag{3.43}$$

$$\nabla\times\mathbf{u} = 0 \quad \text{in } \Omega, \tag{3.44}$$

$$\phi = 0 \quad \text{on } \Gamma, \tag{3.45}$$

$$\mathbf{n}\times\mathbf{u} = 0 \quad \text{on } \Gamma. \tag{3.46}$$

To apply the ADN theory we then include a slack variable, ψ, in the following way:

$$\nabla \cdot \mathbf{u} + \mathbf{b} \cdot \nabla \phi = f \quad \text{in } \Omega, \tag{3.47}$$

$$\mathbf{u} + \nabla \phi + \nabla^{\perp} \psi = 0 \quad \text{in } \Omega, \tag{3.48}$$

$$\nabla \times \mathbf{u} = 0 \quad \text{in } \Omega, \tag{3.49}$$

$$\phi = 0 \quad \text{on } \Gamma, \tag{3.50}$$

$$\mathbf{n} \times \mathbf{u} = 0 \quad \text{on } \Gamma. \tag{3.51}$$

If we let $\boldsymbol{v} = (\phi, \psi, u, v)^T$, this leads to the following representation of the 2D extended Div-Grad convection-diffusion operator and boundary operator:

$$\mathcal{L}\boldsymbol{v} = \begin{pmatrix} b_1\frac{\partial}{\partial x} + b_2\frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ 0 & 0 & -\frac{\partial}{\partial y} & \frac{\partial}{\partial x} \\ \frac{\partial}{\partial x} & -\frac{\partial}{\partial y} & 1 & 0 \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 & 1 \end{pmatrix} \begin{pmatrix} \phi \\ \psi \\ u \\ v \end{pmatrix}, \quad \mathcal{R}\boldsymbol{v} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -n_2 & n_1 \end{pmatrix} \begin{pmatrix} \phi \\ \psi \\ u \\ v \end{pmatrix}.$$

This system is ADN elliptic for the choice of indices $s_i = \{0,0,0,0\}$, $t_j = \{1,1,1,1\}$ (and hence is homogeneous elliptic) yielding the following principal part:

$$\mathcal{L}^p = \begin{pmatrix} b_1\frac{\partial}{\partial x} + b_2\frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ 0 & 0 & -\frac{\partial}{\partial y} & \frac{\partial}{\partial x} \\ \frac{\partial}{\partial x} & -\frac{\partial}{\partial y} & 0 & 0 \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 & 0 \end{pmatrix}.$$

This only differs from the principal part of the extended Div-Grad formulation of the Poisson equation in the top left entry. Conveniently this does not affect the value of $\det(\mathcal{L}^p(\boldsymbol{\xi}))$ so that we know $\det(\mathcal{L}^p(\boldsymbol{\xi})) = |\boldsymbol{\xi}|^4$ and hence this system satisfies the supplementary condition.

To verify the complementing condition we define the the principal part of the boundary operator with respect to the boundary indices $r_l = \{-1,-1\}$ such that $\mathcal{R}^p = \mathcal{R}$. Consider the vector $\boldsymbol{\eta} = \boldsymbol{\xi} + \tau\mathbf{n} = (\eta_1, \eta_2)^T$. Since $\det(\mathcal{L}^p)$ is the same as for the extended Div-Grad system for the Poisson equation we can assume without loss of generality that:

$$\det(\mathcal{L}^p(\boldsymbol{\eta})) = (1+\tau^2)^2,$$

and hence $M^+(\boldsymbol{\xi}, \tau) = (\tau - i)^2$. We can again further assume that the coordinate directions are aligned with the directions of $\boldsymbol{\xi}$ and $\mathbf{n}$ such that $\boldsymbol{\xi} = (1,0)^T$, $\mathbf{n} = (0,1)^T$ and $\boldsymbol{\eta} = (1,\tau)^T$. The complementing condition holds if the rows of $\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}'(\boldsymbol{\eta})$ are linearly independent modulo M^+ where $\mathcal{L}'$ is the adjoint of $\mathcal{L}^p$ which can readily be calculated to be:

$$\mathcal{L}'(\boldsymbol{\eta}) = \begin{pmatrix} 0 & 0 & (1+\tau^2) & \tau(1+\tau^2) \\ 0 & 0 & -\tau(1+\tau^2) & (1+\tau^2) \\ (1+\tau^2) & -\tau(1+\tau^2) & -(b_1+\tau b_2) & -\tau(b_1+\tau b_2) \\ \tau(1+\tau^2) & (1+\tau^2) & -\tau(b_1+\tau b_2) & -\tau^2(b_1+\tau b_2) \end{pmatrix},$$

and hence we have that:

$$\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}'(\boldsymbol{\eta}) = \begin{pmatrix} 0 & 0 & (1+\tau^2) & \tau(1+\tau^2) \\ -(1+\tau^2) & \tau(1+\tau^2) & (b_1+\tau b_2) & \tau(b_1+\tau b_2) \end{pmatrix}. \tag{3.52}$$

For the complementing condition to hold we require that:

$$-c_2(1+\tau^2) = (\tau - i)^2 p_1(\tau), \tag{3.53}$$

$$c_2\tau(1+\tau^2) = (\tau - i)^2 p_2(\tau), \tag{3.54}$$

$$c_1(1+\tau^2) + c_2(b_1+\tau b_2) = (\tau - i)^2 p_3(\tau), \tag{3.55}$$

$$c_1\tau(1+\tau^2) + c_2\tau(b_1+\tau b_2) = (\tau - i)^2 p_4(\tau), \tag{3.56}$$

only holds for $c_1 = c_2 = 0$. Indeed, writing $(1+\tau^2) = (\tau+i)(\tau-i)$ and cancelling a factor $(\tau - i)$ from both sides of (3.53) and (3.54) it becomes trivially clear that these two equations are only satisfied for $c_2 = 0$ and $p_1(\tau) = p_2(\tau) = 0$. Putting $c_2 = 0$ into (3.55) and (3.56) and cancelling $(\tau - i)$ from both sides once again reveals that it is also trivially true that $c_1 = 0$ and $p_3(\tau) = p_4(\tau) = 0$. Hence the complementing condition is satisfied.

If we then put the relevant choice of ADN indices into Theorem 5 (with $q = 0$) where we have removed the slack variable, ψ, and imposed the boundary conditions strongly we obtain the following coercivity estimate:

$$\|\phi\|_1 + \|\mathbf{u}\|_1 \leq C_0(\|\nabla\cdot\mathbf{u} + \mathbf{b}\cdot\nabla\phi\|_0 + \|\mathbf{u} + \nabla\phi\|_0 + \|\nabla\times\mathbf{u}\|_0),$$

for some constant $C_0 > 0$. This system is therefore homogeneous elliptic and has associated quadratic least-squares functional:

$$\mathcal{J}_3(\phi, \mathbf{u}) = \|\nabla\cdot\mathbf{u} + \mathbf{b}\cdot\nabla\phi - f\|_0^2 + \|\mathbf{u} + \nabla\phi\|_0^2 + \|\nabla\times\mathbf{u}\|_0^2.$$

We are then able to derive the Euler-Lagrange equation associated with the minimisation of $\mathcal{J}_3(\phi, \mathbf{u})$: Find $\boldsymbol{v} = (\phi, \mathbf{u})^T \in H_0^1(\Omega) \times \mathbf{H}_\times^1(\Omega)$ such that:

$$A_3(\boldsymbol{v}, \boldsymbol{v}^*) = L_3(\boldsymbol{v}^*), \quad \forall \boldsymbol{v}^* = (\phi^*, \mathbf{u}^*)^T \in H_0^1(\Omega) \times \mathbf{H}_\times^1(\Omega),$$

where:

$$\begin{aligned}
A_3(\boldsymbol{v}, \boldsymbol{v}^*) =& \langle \nabla \cdot \mathbf{u} + \mathbf{b} \cdot \nabla \phi, \nabla \cdot \mathbf{u}^* + \mathbf{b} \cdot \nabla \phi^* \rangle_0 + \langle \mathbf{u} + \nabla \phi, \mathbf{u}^* + \nabla \phi^* \rangle_0 \\
&+ \langle \nabla \times \mathbf{u}, \nabla \times \mathbf{u}^* \rangle_0, \\
L_3(\boldsymbol{v}^*) =& \langle f, \nabla \cdot \mathbf{u}^* + \mathbf{b} \cdot \nabla \phi^* \rangle_0.
\end{aligned}$$

3.5.3 Numerical Considerations

Before presenting some numerical results for this problem we first mention some numerical considerations related to the convection-diffusion equation. Namely, it is well known that for convection dominated problems (when $|\mathbf{b}|$ is sufficiently large) numerical methods for solving the convection-diffusion problem can suffer from numerical instabilities (see e.g. Brooks and Hughes [33]). One way that this problem can be overcome is by using a stabilised streamline upwind/Petrov-Galerkin (SUPG) method to solve the convection dominated problem [33]. The SUPG method has recently been applied in the context of the PGD by González et al. [72].

As for least-squares methods; numerical experiments by Hsieh and Yang [78] have revealed that standard least-squares methods perform very poorly for convection dominated problems where large spurious oscillations are observed. This issue was not alleviated by using a very fine mesh or by using higher order finite element basis functions. An early attempt to resolve this issue was proposed by Fiard et al. [68] who use an exponentially weighted least-squares functional. The disadvantages of this method were pointed out in [78] where the so-called residual free bubble strategy was proposed. This method, however, is relatively difficult to implement. A more recent proposal by Chen et al. [45] suggests imposing the Dirichlet boundary conditions weakly in a manner that ensures errors along boundary layers do not propagate into the whole domain.

As mentioned earlier Cancès et al. [38] have recently investigated PGD algorithms for non-symmetric problems such as convection-diffusion. In this paper they investigated use of a minimal residual PGD algorithm to symmetrise the problem and to provide a proof of convergence of the associated greedy algorithm. They first investigated minimising the residual in an L^2-norm as we do in least-squares PGD algorithms. However they argued that given a problem in operator form: $Au = l$, an L^2-minimal residual greedy algorithm instead solves the problem:

$$A^*Au = A^*l,$$

and hence the conditioning of this problem scales quadratically with the conditioning of the original problem. This is clearly an issue for convection dominated convection-diffusion equations where we expect the conditioning of the original problem to be particularly poor. The authors attempted to tackle this issue by using dual norm minimal residual greedy algorithms which proved inefficient due to the inherent issues in constructing an inverse of the Riesz operator since it cannot, in general, be expressed as a finite sum of tensorised operators. This is analogous to the issues mentioned in Section 3.4.2 with the third way of approximating the norm generating operator for the negative index norm involved in non-homogeneous elliptic systems which we rejected for exactly this reason.

For our purposes we do not intend to attempt to stabilise the problem but instead we look to investigate how standard least-squares PGD algorithms perform for convection dominated problems. In particular we would like to compare how the convergence in the rank of homogeneous and non-homogeneous elliptic formulations are affected as we increase the magnitude of the convective term $|\mathbf{b}|$. However, there is certainly great promise in using any of the previously mentioned methods for stabilising least-squares methods for convection dominated problems and we leave this as potential future work.

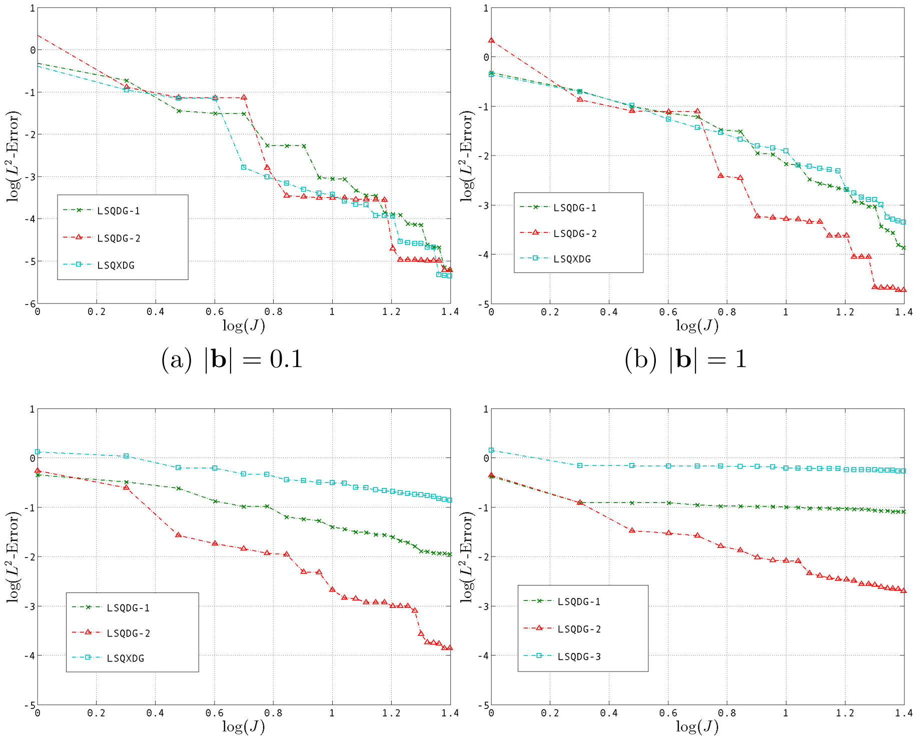
3.5.4 Numerical Results

To test how our least-squares PGD algorithms behave as we increase the magnitude of the convective term we shall consider a single example with an infinite rank solution. To this end we consider the same domain as in Example 5, that is $\Omega = [-1, 1]^2$ with homogeneous boundary conditions and with source term:

$$\begin{aligned} f(x, y) =& 4\pi^2(x^2(1-y^2)^2 + y^2(1-x^2)^2)\sin(\pi(1-x^2)(1-y^2)) \\ &+ 2\pi((1-b_1x)(1-y^2) + (1-b_2y)(1-x^2)\cos(\pi(1-x^2)(1-y^2)) \end{aligned}$$

This has the same exact solution as in Example 5: $\phi = \sin(\pi(1-x^2)(1-y^2))$.

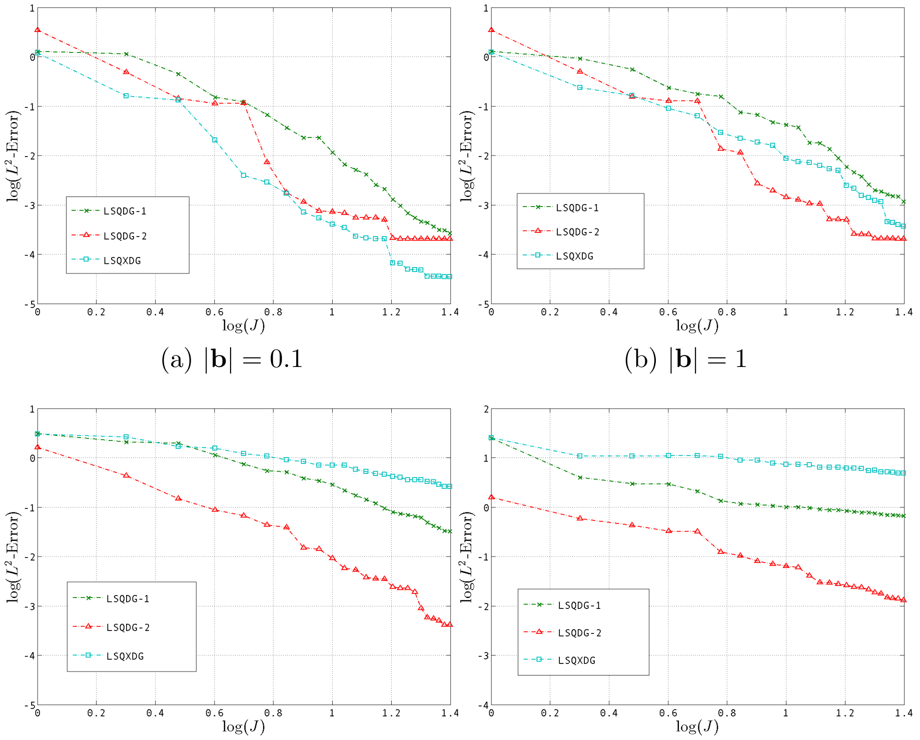
Figure 3.7 shows the convergence of ϕ for four increasing values of the magnitude of the convection term and Figure 3.8 shows convergence for $\mathbf{u}$. From these plots we find that when $|\mathbf{b}| = 0.1$ the rates of convergence closely resemble the results of the Poisson equation in Example 5. However, as we increase $|\mathbf{b}|$ and the problem becomes increasingly convection dominated we notice that the rate of convergence seriously degrades for all three algorithms. In particular, and rather surprisingly, the worst affected of the algorithms appears to be the homogeneous elliptic extended Div-Grad algorithm (LSQXDG). On the other hand the LSQDG-2 algorithm appears to far outperform the other algorithms for convection dominated problems.



(a) $|\mathbf{b}| = 0.1$ (b) $|\mathbf{b}| = 1$

(c) $|\mathbf{b}| = 10$ (d) $|\mathbf{b}| = 100$

Figure 3.7: Error in ϕ for Convection-Diffusion Equation

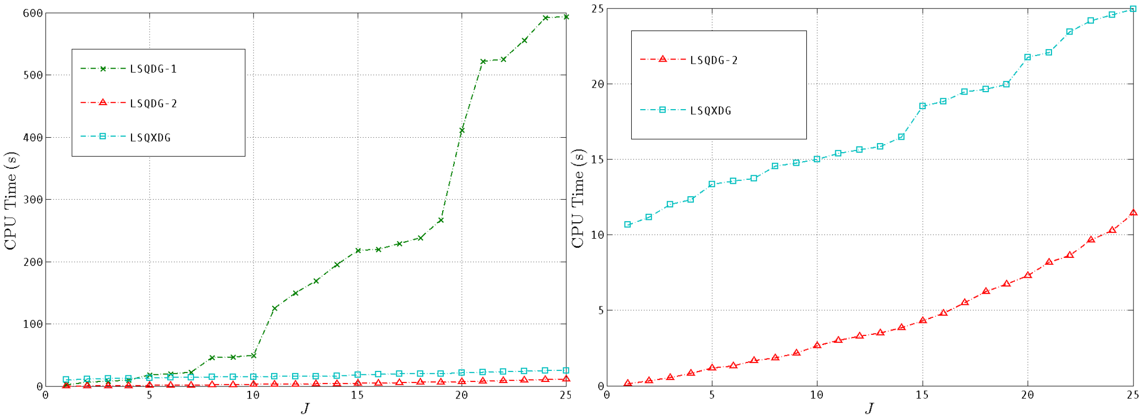


(a) $|\mathbf{b}| = 0.1$ (b) $|\mathbf{b}| = 1$

(c) $|\mathbf{b}| = 10$ (d) $|\mathbf{b}| = 100$

Figure 3.8: Error in $\mathbf{u}$ for Convection-Diffusion Equation

To further investigate this, in Figure 3.9, we have plotted the CPU times for the three algorithms when $|\mathbf{b}| = 100$. From plot (a) we find that the LSQDG-1 algorithm is once again considerably slower than the other two algorithms indicating serious conditioning problems which can be attributed to a combination of the convection dominated problem as well as the inherent issues with this formulation as indicated by the CPU time for the Poisson equation.



(a) With LSQDG-1

(b) Without LSQDG-1

Figure 3.9: CPU Time for Convection-Diffusion ($|\mathbf{b}| = 100$)

In plot (b) we have filtered out LSQDG-1 in order to more accurately compare the last two algorithms. From this we can see that the first iteration of the LSQXDG algorithm takes considerably longer than the following iterations. This is again an indicator of conditioning problems in this algorithm. The LSQDG-2 algorithm on the other hand runs very quickly even for the convection dominated problem.

3.5.5 Conclusions

In the results of the Poisson equation we noted that the LSQXDG algorithm was a clear winner in terms of both accuracy and efficiency. This is still the case for the convection-diffusion equation provided the convective term is sufficiently small. However, when the equation becomes convection dominated the situation dramatically alters. The LSQXDG algorithm then becomes the least accurate of the three algorithms being trumped considerably by the LSQDG-2 algorithm.

We found this to be a surprising result at first but we believe the reason for this is related to the conditioning of the problems. As mentioned previously Cancès et al. [38] explain why minimising in the L^2-norm leads to quadratic scaling of the conditioning of the original problem. We also note that the homogeneous elliptic LSQXDG algorithm is based on L^2-norm minimisation in the continuous sense and so we would certainly expect this problem to be experienced here. On the other hand, the non-homogeneous elliptic algorithms LSQDG-1 and 2 are based on a dual norm minimisation in the continuous sense (at least in the convective term) which is exactly what the authors of [38] tried to use to alleviate the problem of conditioning in L^2-norm minimisation. Although our discrete approximations of the norm generating operators are crude; we believe this may alleviate the conditioning sufficiently to outperform the LSQXDG algorithm.

However, we must point out again that these algorithms were run for an unstabilised convection diffusion equation. If one were to employ one of the previously mentioned methods of stabilising least-squares methods for convection dominated problems the outcome of these results could be very different. Indeed, if the problem could be stabilised sufficiently to improve the conditioning of the original problem we would expect to find the LSQXDG algorithm once again outperforming the non-homogeneous elliptic algorithms.

We will now move on to investigating another problem which does not possess a natural energy minimisation principle: the Stokes problem.

3.6 Least-Squares Formulation of the Stokes Problem

We now turn our attention back to the Stokes problem. We had previously attempted to apply a PGD algorithm to the Galerkin formulation of this problem. Unfortunately we could no longer guarantee the required LBB stability when seeking solutions in the non-linear manifold, $\mathcal{S}_1$, of rank-one tensors. As a result we found that the algorithm was unreliable and often got stuck in the ADFPA. By using least-squares method instead of the Galerkin formulation we no longer solve a saddle-point problem and hence no longer need to satisfy the LBB condition.

The Stokes problem in its classical form, with Dirichlet boundary conditions on the velocities, is given by:

$$-\nabla^2\mathbf{u} + \nabla p = \mathbf{f} \quad \text{in} \ \ \Omega, \tag{3.57}$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{in} \ \ \Omega, \tag{3.58}$$

$$\mathbf{u} = \mathbf{g} \quad \text{on} \ \ \Gamma. \tag{3.59}$$

There are several possible equivalent first order systems for the Stokes problem and a wide selection have been well documented in the thesis of Proot [111]. In this report we shall consider two of these formulations. The first is the velocity-vorticity-pressure (VVP) system which is the most commonly used reformulation of the Stokes problem in the literature. This is because it requires comparatively fewer dependent variables and can give a direct and accurate approximation to the vorticity. Unfortunately we shall see that that this formulation does not supply us with a homogeneous elliptic system when Dirichlet boundary conditions on the velocities are used. Therefore, we shall also consider the extended velocity gradient-velocity-pressure (Extended VGVP) system reformulation which has a larger number of dependent variables but does supply us with a homogeneous elliptic system when Dirichlet boundary conditions on the velocities are imposed.

3.6.1 VVP System

To derive the velocity-vorticity-pressure formulation of the Stokes problem we first define the vorticity in 2D by $\omega = \nabla \times \mathbf{u}$. Then using the identity $\nabla^{\perp}(\nabla \times \mathbf{u}) = -\nabla^2\mathbf{u} + \nabla(\nabla \cdot \mathbf{u})$ together with incompressibility $\nabla \cdot \mathbf{u} = 0$ we can write $-\nabla^2\mathbf{u} = \nabla^{\perp}(\nabla \times \mathbf{u}) = \nabla^{\perp}\omega$. Hence the VVP system is given by:

$$\nabla^{\perp}\omega + \nabla p = \mathbf{f} \quad \text{in } \Omega, \tag{3.60}$$

$$\omega - \nabla \times \mathbf{u} = 0 \quad \text{in } \Omega, \tag{3.61}$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega. \tag{3.62}$$

$$\mathbf{u} = \mathbf{g} \quad \text{on } \Gamma. \tag{3.63}$$

If we let $\boldsymbol{v} = (u, v, \omega, p)^T$ then this leads to the following representation of the 2D Stokes VVP operator and boundary operator:

$$\boldsymbol{\mathcal{L}}\boldsymbol{v} = \begin{pmatrix} 0 & 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \\ 0 & 0 & -\frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & -\frac{\partial}{\partial x} & 1 & 0 \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & 0 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ \omega \\ p \end{pmatrix}, \quad \boldsymbol{\mathcal{R}}\boldsymbol{v} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ \omega \\ p \end{pmatrix},$$

The differential operator $\boldsymbol{\mathcal{L}}$ is elliptic in the usual sense. Indeed, we have ADN ellipticity for the choice of indices $\{s_i\} = \{0, 0, 0, 0\}$ and $\{t_j\} = \{1, 1, 1, 1\}$ yielding the principal part:

$$\boldsymbol{\mathcal{L}}_1^p = \begin{pmatrix} 0 & 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \\ 0 & 0 & -\frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & -\frac{\partial}{\partial x} & 0 & 0 \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & 0 & 0 \end{pmatrix}.$$

From the definition of ADN ellipticity (Definition 1) we can see that this choice of indices satisfies the three conditions. Indeed, condition (i) reduces to $\deg(\boldsymbol{\mathcal{L}}_{i,j}(\boldsymbol{\xi})) \leq 1$ which is clearly true since $\boldsymbol{\mathcal{L}}$ is a first order differential operator, condition (ii) is satisfied trivially since $s_i + t_j \geq 0$ for all $i, j = 1, \ldots, 4$ and condition (iii) is also satisfied since:

$$\det(\boldsymbol{\mathcal{L}}_1^p(\boldsymbol{\xi})) = \begin{vmatrix} 0 & 0 & \xi_2 & \xi_1 \\ 0 & 0 & -\xi_1 & \xi_2 \\ \xi_2 & -\xi_1 & 0 & 0 \\ \xi_1 & \xi_2 & 0 & 0 \end{vmatrix} = (\xi_1^2 + \xi_2^2)^2 = |\boldsymbol{\xi}|^4,$$

Unfortunately it has been shown, in [25,111] for example, that the boundary operator $\boldsymbol{\mathcal{R}}$ associated with Dirichlet boundary conditions on the velocities does not satisfy the complementing condition with the principal part $\mathcal{L}_1^p$. This means that we are unable to obtain a homogeneous elliptic system with this choice of boundary condition and instead must consider a second principal part. The second principal part we consider comes from the choice of indices $\{s_i\} = \{0, 0, -1, -1\}$ and $\{t_j\} = \{2, 2, 1, 1\}$ yielding:

$$\mathcal{L}_2^p = \begin{pmatrix} 0 & 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \\ 0 & 0 & -\frac{\partial}{\partial x} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & -\frac{\partial}{\partial x} & 1 & 0 \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & 0 & 0 \end{pmatrix}.$$

Again this choice of indices can be seen to satisfy the three conditions for ADN ellipticity. Indeed, condition (i) only needs to be checked when $s_i + t_j < 1$ which occurs when $i, j = 3, 4$ at which point the operator $\mathcal{L}_{i,j}(\boldsymbol{\xi})$ only contains constant values (i.e. degree 0 terms) and so (i) is satisfied. Condition (ii) is again trivial since $s_i + t_j \geq 0$ for all $i, j = 1, \ldots, 4$ and for condition (iii) it can easily be checked that we again have that $\det(\mathcal{L}_2^p(\boldsymbol{\xi})) = |\boldsymbol{\xi}|^4$.

We can further see, from both these principal parts, that since $\det(\mathcal{L}_1^p(\boldsymbol{\xi})) = \det(\mathcal{L}_2^p(\boldsymbol{\xi})) = |\boldsymbol{\xi}|^4$ then the Stokes VVP operator $\mathcal{L}$ is uniformly ADN elliptic of order four. This means that we need to impose $m = 2$ boundary conditions which is what we expect since this is the number of boundary conditions we impose in the classical formulation of the Stokes problem. It is also clear, by Lemma 4, that this also satisfies the supplementary condition.

The complementing condition between $\mathcal{L}_2^p$ and $\mathcal{R}$ can also be proven to hold. We will not show the working for this since it can be found in various places in the literature (e.g. [25, 111]) This is due to the Stokes problem, and in particular the VVP system, being the most popular application of least-squares methods.

Therefore, if we put the corresponding choice of indices into the coercivity estimate (3.10) in Theorem 5, where we impose the boundary conditions strongly, we obtain the following estimate:

$$\|\mathbf{u}\|_{q+2} + \|\omega\|_{q+1} + \|p\|_{q+1} \leq C_q\big(\|\nabla^\perp\omega + \nabla p\|_q + \|\omega - \nabla \times \mathbf{u}\|_{q+1} + \|\nabla \cdot \mathbf{u}\|_{q+1}\big), \quad (3.64)$$

for some constant $C_q > 0$. In the same way as for the Div-Grad system, this can be extended to all $q \in \mathbb{R}$ (see Bochev and Gunzburger [24]). Hence we can choose $q = -1$ to overcome practicality issues related to the required differentiability of the involved function spaces. This yields the following coercivity estimate:

$$\|\mathbf{u}\|_1 + \|\omega\|_0 + \|p\|_0 \leq C_{-1}\big(\|\nabla^\perp\omega + \nabla p\|_{-1} + \|\omega - \nabla \times \mathbf{u}\|_0 + \|\nabla \cdot \mathbf{u}\|_0\big). \quad (3.65)$$

Note that these coercivity estimates (3.64)-(3.65) rely on the assumption that there exists a unique solution. Since the pressure can only be evaluated up to a constant, we need to include an additional constraint in the quadratic least-squares functionals to ensure uniqueness. For the Stokes problem we use the zero mean pressure constraint $\ell(p) = \int_\Omega p \, d\Omega = 0$. The negative index norm in (3.65) is treated in the same way as for the Div-Grad system yielding the following two quadratic least-squares functionals:

$$\mathcal{J}_1(\mathbf{u}, \omega, p) = \|\nabla^\perp \omega + \nabla p - \mathbf{f}\|_0^2 + \|\omega - \nabla \times \mathbf{u}\|_0^2 + \|\nabla \cdot \mathbf{u}\|_0^2 + \mu |\ell(p)|^2, \qquad (3.66)$$

$$\mathcal{J}_2(\mathbf{u}, \omega, p) = h^2 \|\nabla^\perp \omega + \nabla p - \mathbf{f}\|_0^2 + \|\omega - \nabla \times \mathbf{u}\|_0^2 + \|\nabla \cdot \mathbf{u}\|_0^2 + \mu |\ell(p)|^2, \qquad (3.67)$$

where $\mu > 0$ is an adjustable constant. We are then able to derive the Euler-Lagrange equations associated with the minimisation of the functionals (3.66)-(3.67): Find $\boldsymbol{v} = (\mathbf{u}, \omega, p) \in H_{\mathbf{g}}(\text{div}) \cap H_{\mathbf{g}}(\text{curl}) \times H^1(\Omega) \times H^1(\Omega)$ such that:

$$A_k(\boldsymbol{v}, \boldsymbol{v}^*) = L_k(\boldsymbol{v}^*), \quad \forall \boldsymbol{v}^* = (\mathbf{u}^*, \omega^*, p^*) \in H_{\mathbf{0}}(\text{div}) \cap H_{\mathbf{0}}(\text{curl}) \times H^1(\Omega) \times H^1(\Omega),$$

for $k = 1, 2$, where

$$\begin{aligned} A_1(\boldsymbol{v}, \boldsymbol{v}^*) =& \langle \nabla^\perp \omega + \nabla p, \nabla^\perp \omega^* + \nabla p^* \rangle_0 + \langle \omega - \nabla \times \mathbf{u}, \omega^* - \nabla \times \mathbf{u}^* \rangle_0 \\ &+ \langle \nabla \cdot \mathbf{u}, \nabla \cdot \mathbf{u}^* \rangle_0 + \mu \ell(p) \ell(p^*), \\ A_2(\boldsymbol{v}, \boldsymbol{v}^*) =& h^2 \langle \nabla^\perp \omega + \nabla p, \nabla^\perp \omega^* + \nabla p^* \rangle_0 + \langle \omega - \nabla \times \mathbf{u}, \omega^* - \nabla \times \mathbf{u}^* \rangle_0 \\ &+ \langle \nabla \cdot \mathbf{u}, \nabla \cdot \mathbf{u}^* \rangle_0 + \mu \ell(p) \ell(p^*), \end{aligned}$$

and

$$L_1(\boldsymbol{v}^*) = \langle \mathbf{f}, \nabla^\perp \omega^* + \nabla p^* \rangle_0, \quad L_2(\boldsymbol{v}^*) = h^2 \langle \mathbf{f}, \nabla^\perp \omega^* + \nabla p^* \rangle_0.$$

3.6.2 Extended VGVP System

We now consider the velocity gradient-velocity pressure formulation of the Stokes problem. We begin by defining the velocity gradient by:

$$\underline{\mathbf{V}} = (\nabla \mathbf{u})^T = \begin{pmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{pmatrix} = \begin{pmatrix} V_1 & V_2 \\ V_3 & V_4 \end{pmatrix}.$$

If we then define the divergence of a tensor to be the divergence of its rows then we obtain the identity $\nabla \cdot \underline{\mathbf{V}} = \nabla^2 \mathbf{u}$. Hence we can rewrite the Stokes problem (3.57)-(3.59) as the following first-order VGVP system:

$$-\nabla \cdot \underline{\mathbf{V}} + \nabla p = \mathbf{f} \quad \text{in } \Omega, \qquad (3.68)$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega, \qquad (3.69)$$

$$\underline{\mathbf{V}} - (\nabla \mathbf{u})^T = \underline{\mathbf{0}} \quad \text{in } \Omega, \qquad (3.70)$$

$$\mathbf{u} = \mathbf{g} \quad \text{on } \Gamma. \qquad (3.71)$$

Unfortunately, it has been shown by Cai et al. [36] that this does not lead to a homogeneous elliptic system. However, in the same manner as we did for the Div-Grad formulation of the Poisson equation we can include additional redundant equations to provide us with a problem which is homogeneous elliptic. Indeed, this leads to the following extended VGVP system:

$$-\nabla \cdot \underline{\mathbf{V}} + \nabla p = \mathbf{f} \quad \text{in } \Omega, \tag{3.72}$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega, \tag{3.73}$$

$$\underline{\mathbf{V}} - (\nabla \mathbf{u})^T = \underline{\mathbf{0}} \quad \text{in } \Omega, \tag{3.74}$$

$$\nabla(\operatorname{Tr} \underline{\mathbf{V}}) = \mathbf{0} \quad \text{in } \Omega, \tag{3.75}$$

$$\nabla \times \underline{\mathbf{V}} = \mathbf{0} \quad \text{in } \Omega, \tag{3.76}$$

$$\mathbf{u} = \mathbf{0} \quad \text{on } \Gamma, \tag{3.77}$$

$$\mathbf{n} \times \underline{\mathbf{V}} = \mathbf{0} \quad \text{on } \Gamma. \tag{3.78}$$

The additional boundary condition holds since from (3.74) we have that $\mathbf{n} \times \underline{\mathbf{V}} = \mathbf{n} \times (\nabla \mathbf{u})^T = 0$ since the boundary condition on $\mathbf{u}$ implies that its tangential derivatives vanish on the boundary. Note that for simplicity of notation we have made the Dirichlet boundary condition on $\mathbf{u}$ homogeneous. For the non-homogeneous case, $\mathbf{u} = \mathbf{g}$ on Γ, the additional boundary condition (3.78) should be replaced by $\mathbf{n} \times \underline{\mathbf{V}} = \mathbf{n} \times (\nabla \mathbf{g})^T$.

The first redundant equation, (3.75), is satisfied since $\operatorname{Tr} \underline{\mathbf{V}} = V_1 + V_4 = \nabla \cdot \mathbf{u} = 0$, where $\operatorname{Tr} \underline{\mathbf{V}}$ denotes the trace of $\underline{\mathbf{V}}$. The second redundant equation, (3.76), is satisfied since if we define the curl of a tensor to be the the curl of its rows then we have that:

$$\nabla \times \underline{\mathbf{V}} = \begin{pmatrix} \frac{\partial^2 u}{\partial x \partial y} - \frac{\partial^2 u}{\partial y \partial x} \\ \frac{\partial^2 v}{\partial x \partial y} - \frac{\partial^2 u}{\partial y \partial x} \end{pmatrix} = \mathbf{0}.$$

This system has been proven to be homogeneous elliptic in an ad hoc manner by Cai et al. [36]. However, we will show that this can also be proven using the ADN theory in 2D. To do this we must first introduce some slack variables as we did for the extended Div-Grad system. We need to include four slack variables, $\boldsymbol{\theta} = (\theta_1, \theta_2)^T$ and $\boldsymbol{\phi} = (\phi_1, \phi_2)^T$, to ensure that the system has the same number of unknowns as equations. To ensure homogeneous ellipticity we must include an extra equation acting on the slack variables only and hence a fifth slack variable, ψ. We include these into the system in the following way:

$$
\begin{aligned}
-\nabla \cdot \underline{\mathbf{V}} + \nabla p &= \mathbf{f} && \text{in } \Omega, && (3.79)\\
\nabla \cdot \mathbf{u} &= 0 && \text{in } \Omega, && (3.80)\\
\underline{\mathbf{V}} - (\nabla \mathbf{u})^T + \mathcal{D}\boldsymbol{\theta} + (\nabla^{\perp}\boldsymbol{\phi})^T &= \underline{\mathbf{0}} && \text{in } \Omega, && (3.81)\\
\nabla(\operatorname{Tr}\underline{\mathbf{V}}) + \nabla^{\perp}\psi &= \mathbf{0} && \text{in } \Omega, && (3.82)\\
\nabla \times \underline{\mathbf{V}} &= \mathbf{0} && \text{in } \Omega, && (3.83)\\
\nabla \times \boldsymbol{\theta} &= 0 && \text{in } \Omega, && (3.84)\\
\mathbf{u} &= \mathbf{0} && \text{on } \Gamma, && (3.85)\\
\mathbf{n} \times \underline{\mathbf{V}} &= \mathbf{0} && \text{on } \Gamma, && (3.86)\\
\boldsymbol{\theta}^{\perp} + \boldsymbol{\phi} &= \mathbf{0} && \text{on } \Gamma, && (3.87)
\end{aligned}
$$

where $\boldsymbol{\theta}^{\perp} = (-\theta_2, \theta_1)^T$, and where:

$$\mathcal{D} := \begin{pmatrix} \nabla\cdot & 0 \\ 0 & \nabla\cdot \end{pmatrix}.$$

We include the slack variables in this way to ensure the system is homogeneous elliptic and so that the slack variables have an exact solution of zero and hence can be removed from the system. Indeed, if we take the curl of equation (3.81) and use equation (3.83) we obtain the following:

$$\nabla^{\perp}(\nabla \cdot \boldsymbol{\theta}) + \nabla^2 \boldsymbol{\phi} = \mathbf{0}. \tag{3.88}$$

In 2D we have the following identity:

$$\nabla^{\perp}(\nabla \times \boldsymbol{\theta}) + \nabla(\nabla \cdot \boldsymbol{\theta}) = \nabla^2 \boldsymbol{\theta}, \tag{3.89}$$

as well as the orthogonal identity:

$$\nabla(\nabla \times \boldsymbol{\theta}) + \nabla^{\perp}(\nabla \cdot \boldsymbol{\theta}) = \nabla^2 \boldsymbol{\theta}^{\perp}. \tag{3.90}$$

We can use this orthogonal identity (3.90) and the fact that $\nabla \times \boldsymbol{\theta} = 0$ to rewrite (3.88) as the following Laplace equation:

$$\nabla^2(\boldsymbol{\theta}^{\perp} + \boldsymbol{\phi}) = \mathbf{0}.$$

Together with the boundary condition (3.87) we can see that $\boldsymbol{\phi} = -\boldsymbol{\theta}^{\perp}$. Putting this into equation (3.81) we obtain the following:

$$\underline{\mathbf{V}} - (\nabla \mathbf{u})^T + \mathcal{D}\boldsymbol{\theta} - (\nabla^{\perp}\boldsymbol{\theta}^{\perp})^T = \underline{\mathbf{0}}, \tag{3.91}$$

where

$$(\nabla^{\perp}\boldsymbol{\theta}^{\perp})^T = \begin{pmatrix} \frac{\partial\theta_2}{\partial y} & \frac{-\partial\theta_2}{\partial x} \\ \frac{-\partial\theta_1}{\partial y} & \frac{\partial\theta_1}{\partial x} \end{pmatrix}.$$

If we now take the gradient of the trace of equation (3.91) and use equation (3.82) we obtain the following:

$$-\nabla^{\perp}\psi + 2\nabla(\nabla\cdot\boldsymbol{\theta}) - \nabla(\nabla\cdot\boldsymbol{\theta}) = -\nabla^{\perp}\psi + \nabla(\nabla\cdot\boldsymbol{\theta}) = \mathbf{0}.$$

We can then use the identity (3.89) together with the fact that $\nabla\times\boldsymbol{\theta} = 0$ to rewrite this as the system:

$$\begin{aligned} \nabla^2\boldsymbol{\theta} - \nabla^{\perp}\psi = \mathbf{0} \quad &\text{in } \Omega, \\ \nabla\times\boldsymbol{\theta} = 0 \quad &\text{in } \Omega. \end{aligned}$$

We can rewrite this in terms of $\boldsymbol{\theta}^{\perp}$ to obtain the following homogeneous Stokes equations:

$$\nabla^2\boldsymbol{\theta}^{\perp} + \nabla\psi = \mathbf{0} \quad \text{in } \Omega, \tag{3.92}$$

$$\nabla\cdot\boldsymbol{\theta}^{\perp} = 0 \quad \text{in } \Omega. \tag{3.93}$$

To obtain a boundary condition for this Stokes problem we cross product equation (3.91) with $\mathbf{n}$, and use the additional boundary condition (3.86), which yields $\mathbf{n}\cdot(\nabla\boldsymbol{\theta}^{\perp}) = \mathbf{0}$. This is a Neumann type boundary condition on $\boldsymbol{\theta}^{\perp}$ which we can use together with the Stokes equations (3.92)-(3.93) to obtain the solutions $\boldsymbol{\theta} = \boldsymbol{\theta}^{\perp} = \boldsymbol{\phi} = \mathbf{0}$ and $\psi = 0$ (up to additive constants). We remove the non-uniqueness of the solution, imposing all the slack variables to be zero, by simply removing them from the system.

We have now shown that all the slack variables can be removed from the system in practice. It remains to show that this extended VGVP system with the slack variables included satisfies the supplementary and complementing condition.

If we let $\boldsymbol{v} = (u, v, p, V_1, V_2, V_3, V_4, \theta_1, \theta_2, \phi_1, \phi_2, \psi)^T$ then this leads to the following representation of the 2D XVGVP operator:

$$\mathcal{L}\boldsymbol{v} = \begin{pmatrix} 0 & 0 & \frac{\partial}{\partial x} & \frac{-\partial}{\partial x} & \frac{-\partial}{\partial y} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial}{\partial y} & 0 & 0 & \frac{-\partial}{\partial x} & \frac{-\partial}{\partial y} & 0 & 0 & 0 & 0 & 0 \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{-\partial}{\partial x} & 0 & 0 & 1 & 0 & 0 & 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{-\partial}{\partial y} & 0 & 0 \\ \frac{-\partial}{\partial y} & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{-\partial}{\partial x} & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \frac{-\partial}{\partial y} & 0 \\ 0 & \frac{-\partial}{\partial y} & 0 & 0 & 0 & 0 & 1 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial x} & 0 \\ 0 & 0 & 0 & \frac{\partial}{\partial x} & 0 & 0 & \frac{\partial}{\partial x} & 0 & 0 & 0 & 0 & \frac{-\partial}{\partial y} \\ 0 & 0 & 0 & \frac{\partial}{\partial y} & 0 & 0 & \frac{\partial}{\partial y} & 0 & 0 & 0 & 0 & \frac{\partial}{\partial x} \\ 0 & 0 & 0 & \frac{-\partial}{\partial y} & \frac{\partial}{\partial x} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{-\partial}{\partial y} & \frac{\partial}{\partial x} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-\partial}{\partial y} & \frac{\partial}{\partial x} & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ p \\ V_1 \\ V_2 \\ V_3 \\ V_4 \\ \theta_1 \\ \theta_2 \\ \phi_1 \\ \phi_2 \\ \psi \end{pmatrix},$$

and associated boundary operator:

$$\mathcal{R}\boldsymbol{v} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -n_2 & n_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -n_2 & n_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ p \\ V_1 \\ V_2 \\ V_3 \\ V_4 \\ \theta_1 \\ \theta_2 \\ \phi_1 \\ \phi_2 \\ \psi \end{pmatrix}.$$

To show that this system is homogeneous elliptic we consider the indices $s_i = \{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\}$ and $t_j = \{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\}$. This yields a principal part such that $\det(\mathcal{L}^p(\boldsymbol{\xi})) = |\boldsymbol{\xi}|^{12}$ which can be verified using a suitable symbolic mathematical software package. Hence this system satisfies condition (iii) in the definition of ADN ellipticity and the first two conditions are also trivially satisfied. Hence this system is uniformly ADN elliptic of order 12 and we need to impose $m = 6$ boundary conditions. It is also clear from Lemma 4 that this system satisfies the supplementary condition.

To verify the complementing condition we first define the principal part of the boundary operator with respect to the boundary indices $r_l = \{-1, -1, -1, -1, -1, -1\}$. This simply yields a principal part such that $\mathcal{R}^p = \mathcal{R}$. Consider the vector $\boldsymbol{\eta} = \boldsymbol{\xi} + \tau\mathbf{n} = (\eta_1, \eta_2)^T$. Since $\mathbf{n}$ is the unit normal vector to Γ and $\boldsymbol{\xi}$ is tangent to Γ we have that $\boldsymbol{\xi} \cdot \mathbf{n} = 0$ and $|\mathbf{n}| = 1$. We can further assume, without loss of generality, that $|\boldsymbol{\xi}| = 1$. Hence we have that:

$$\det(\mathcal{L}^p(\boldsymbol{\eta})) = |\boldsymbol{\eta}|^{12} = (|\boldsymbol{\xi}|^2 + 2\tau\boldsymbol{\xi} \cdot \mathbf{n} + \tau^2|\mathbf{n}|^2)^6 = (1 + \tau^2)^6,$$

which has roots of multiplicity six: i and $-i$. Hence $\tau_l^+(\boldsymbol{\xi}) = i$ for $l = 1, \ldots, 6$ and $M^+(\boldsymbol{\xi}, \tau) = (\tau - i)^6$. The complementing conditions holds if the rows of $\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}'(\boldsymbol{\eta})$ are linearly independent modulo M^+, where $\mathcal{L}'$ denotes the adjoint of $\mathcal{L}^p$. To simplify calculations we will begin by assuming that the coordinate axes are aligned with the directions of $\boldsymbol{\xi}$ and $\mathbf{n}$ such that $\boldsymbol{\xi} = (1,0)^T$, $\mathbf{n} = (0,1)^T$ and $\boldsymbol{\eta} = (1,\tau)^T$. One can then verify, using mathematical software, that $\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}'(\boldsymbol{\eta})$ is equal to the following:

$$|\boldsymbol{\eta}|^8 \begin{pmatrix} 0 & 0 & |\boldsymbol{\eta}|^2 & -\tau^2 & -\tau^3 & \tau & \tau^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \tau|\boldsymbol{\eta}|^2 & \tau & \tau^2 & -1 & -\tau & 0 & 0 & 0 & 0 & 0 \\ \tau^2 & -\tau & 0 & 0 & 0 & 0 & 0 & -1 & -\tau & \tau^3 & 1 & 0 \\ -\tau & 1 & 0 & 0 & 0 & 0 & 0 & -\tau & -\tau^2 & -\tau^2 & \tau(|\boldsymbol{\eta}|^2+1) & 0 \\ 0 & 0 & 0 & -\tau|\boldsymbol{\eta}|^2 & |\boldsymbol{\eta}|^2 & 0 & 0 & 0 & 0 & 0 & 0 & -|\boldsymbol{\eta}|^2 \\ 0 & 0 & 0 & 0 & 0 & -\tau|\boldsymbol{\eta}|^2 & |\boldsymbol{\eta}|^2 & 0 & 0 & 0 & 0 & -\tau|\boldsymbol{\eta}|^2 \end{pmatrix},$$

where $|\boldsymbol{\eta}|^2 = (1+\tau^2)$.

Consider the first column of $\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}'(\boldsymbol{\eta})$, for the complementing condition to be satisfied we must have that the following equation is satisfied:

$$(1+\tau^2)^4(c_3\tau^3 - c_4\tau) = (\tau - i)^6 p_1(\tau),$$

if and only if $c_3 = c_4 = 0$, where $p_1(\tau)$ is some polynomial in τ. Using the fact that $(1+\tau^2) = (\tau+i)(\tau-i)$ we can write this as:

$$\tau(\tau+i)^4(c_3\tau^2 - c_4) = (\tau-i)^2 p_1(\tau).$$

For this equation to hold we would need $(c_3\tau^2 - c_4) = k(\tau - i)^2$ for some non-zero constant k. This is not satisfied by any non-zero c_3 and c_4 hence we must have $c_3 = c_4 = 0$.

Consider now the equation associated with the third column of $\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}'(\boldsymbol{\eta})$ simplified in the same way as above:

$$(\tau+i)^5(c_1 + c_2\tau) = (\tau - i)p_3(\tau),$$

for some polynomial $p_3(\tau)$. This is satisfied when $c_1 + c_2\tau = k(\tau - i)$ for some constant k or in other words when $c_1 = -ic_2$. If we then consider the equation associated with the fourth column of $\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}'(\boldsymbol{\eta})$:

$$\tau(\tau+i)^4(-c_1\tau + c_2 - c_5(1+\tau^2)) = (\tau-i)^2 p_4(\tau),$$

for some polynomial $p_4(\tau)$, then we can use the result, $c_1 = -ic_2$, to rewrite this as:

$$-\tau(\tau+i)^4(c_5\tau^2 - ic_2\tau + (c_5 - c_2)) = (\tau - i)^2 p_4(\tau).$$

This is only satisfied if $(c_5\tau^2 - ic_2\tau + (c_5 - c_2)) = k(\tau - i)^2$ for some non-zero constant k. This is not satisfied for any non-zero c_2, c_5 hence we must have that

$c_2 = c_5 = 0$. We also know that $c_1 = 0$ since $c_1 = -ic_2$ and it is then trivially true that $c_6 = 0$ where c_6 is the constant associated with the sixth row of $\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}'(\boldsymbol{\eta})$. This means that that the rows of $\mathcal{R}^p(\boldsymbol{\eta})\mathcal{L}'(\boldsymbol{\eta})$ are linearly independent modulo M^+ and hence the complementing condition is satisfied.

If we now put the corresponding choice of ADN indices into the coercivity estimate (3.10) from Theorem 5 (with $q = 0$), where the boundary conditions have been imposed strongly and the slack variables have been removed, we obtain the following estimate:

$$\begin{aligned}\|\mathbf{u}\|_1 + \|p\|_1 + \|\underline{\mathbf{V}}\|_1 \leq C(\| - \nabla \cdot \underline{\mathbf{V}} + \nabla p\|_0 + \|\nabla \cdot \mathbf{u}\|_0 + \|\underline{\mathbf{V}} - (\nabla \mathbf{u})^T\|_0 \\ + \|\nabla(\mathrm{Tr}\,\underline{\mathbf{V}})\|_0 + \|\nabla \times \underline{\mathbf{V}}\|_0), \qquad (3.94)\end{aligned}$$

for some constant $C > 0$. This is exactly the estimate proven to hold by Cai et al. [36] and it leads to the following quadratic least-squares functional:

$$\begin{aligned}\mathcal{J}_3(\mathbf{u}, p, \underline{\mathbf{V}}) =& \| - \nabla \cdot \underline{\mathbf{V}} + \nabla p - \mathbf{f}\|_0^2 + \|\nabla \cdot \mathbf{u}\|_0^2 + \|\underline{\mathbf{V}} - (\nabla \mathbf{u})^T\|_0^2 \\ &+ \|\nabla(\mathrm{Tr}\,\underline{\mathbf{V}})\|_0^2 + \|\nabla \times \underline{\mathbf{V}}\|_0^2 + \mu|\ell(p)|^2.\end{aligned}$$

We are then able to derive the Euler-Lagrange equation associated with the minimisation of $\mathcal{J}_3(\mathbf{u}, p, \underline{\mathbf{V}})$: Find $\boldsymbol{v} = (\mathbf{u}, p, \underline{\mathbf{V}}) \in \mathbf{H}^1_{\mathbf{0}}(\Omega) \times H^1(\Omega) \times \underline{\mathbf{H}}^1_{\times}(\Omega)$ such that:

$$A_3(\boldsymbol{v}, \boldsymbol{v}^*) = L_3(\boldsymbol{v}^*), \quad \forall \boldsymbol{v}^* = (\mathbf{u}^*, p^*, \underline{\mathbf{V}}^*) \in \mathbf{H}^1_{\mathbf{0}}(\Omega) \times H^1(\Omega) \times \underline{\mathbf{H}}^1_{\times}(\Omega),$$

where:

$$\underline{\mathbf{H}}^1_{\times}(\Omega) = \{\underline{\mathbf{V}} \in \underline{\mathbf{H}}^1(\Omega) \mid \mathbf{n} \times \underline{\mathbf{V}} = \mathbf{0} \quad \text{on} \quad \Gamma\},$$

and where:

$$\begin{aligned}A_3(\boldsymbol{v}, \boldsymbol{v}^*) =& \langle -\nabla \cdot \underline{\mathbf{V}} + \nabla p, -\nabla \cdot \underline{\mathbf{V}}^* + \nabla p^*\rangle_0 + \langle\ \underline{\mathbf{V}} - (\nabla \mathbf{u})^T, \underline{\mathbf{V}}^* - (\nabla \mathbf{u}^*)^T\rangle_0 \\ &+ \langle \nabla \cdot \mathbf{u}, \nabla \cdot \mathbf{u}^*\rangle_0 + \langle \nabla(\mathrm{Tr}\,\underline{\mathbf{V}}), \nabla(\mathrm{Tr}\,\underline{\mathbf{V}}^*)\rangle_0 + \langle \nabla \times \underline{\mathbf{V}}, \nabla \times \underline{\mathbf{V}}^*\rangle_0 \\ &+ \mu\ell(p)\ell(p^*),\end{aligned}$$

and

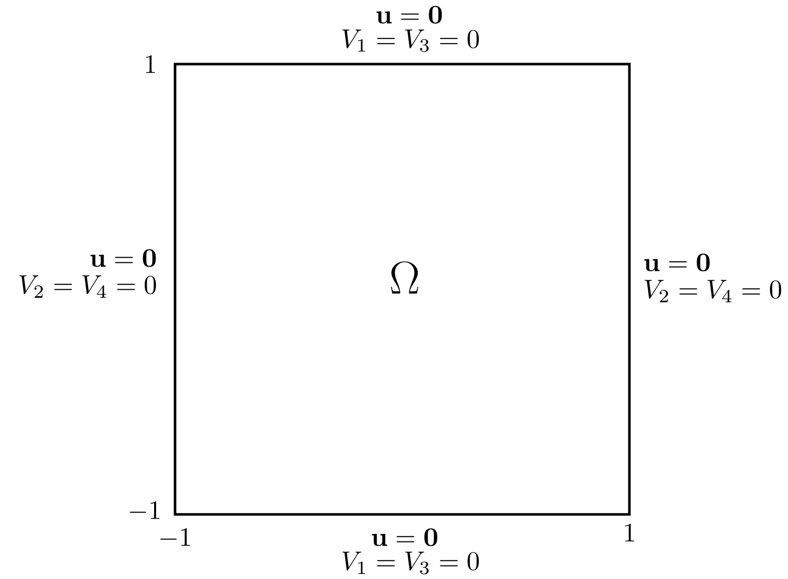
$$L_3(\boldsymbol{v}^*) = \langle \mathbf{f}, -\nabla \cdot \underline{\mathbf{V}}^* + \nabla p^*\rangle_0,$$

where we have included the zero mean pressure constraint $\ell(p) = \int_\Omega p \ d\Omega = 0$ to ensure uniqueness of the solution and where, as before, $\mu > 0$ is an adjustable constant.

3.6.3 Numerical Results

Example 9 (Infinite Rank Pressure Solution)**.**

Consider the Stokes problem (3.57)-(3.59) on the following domain:


$\mathbf{u} = \mathbf{0}$
$V_1 = V_3 = 0$
1
$\mathbf{u} = \mathbf{0}$
$V_2 = V_4 = 0$
Ω
$\mathbf{u} = \mathbf{0}$
$V_2 = V_4 = 0$
-1
-1
$\mathbf{u} = \mathbf{0}$
$V_1 = V_3 = 0$
1

with source term

$$\mathbf{f}(x, y) = \begin{pmatrix} \pi y \cos(\pi xy) + 4\pi^2 \sin(2\pi y)(2\cos(2\pi x) - 1) \\ \pi x \cos(\pi xy) - 4\pi^2 \sin(2\pi x)(2\cos(2\pi y) - 1) \end{pmatrix},$$

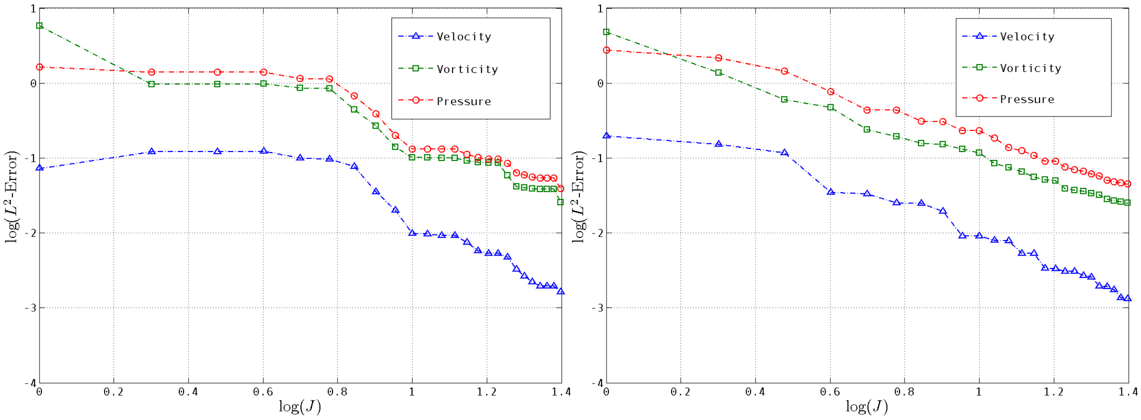
where the boundary conditions on the velocity gradient terms, $V_1, \dots V_4$, are only relevant to the extended VGVP formulation. This Stokes problem has the following exact solution:

$$\mathbf{u} = \begin{pmatrix} -\sin(2\pi y)(\cos(2\pi x) - 1) \\ \sin(2\pi x)(\cos(2\pi y) - 1) \end{pmatrix},$$

$$p = \sin(\pi xy).$$

The velocity, $\mathbf{u}$, possesses a natural rank-1 separated representation and so we might expect our algorithms to converge in a single iteration for the velocity. The pressure, on the other hand, does not have a finite rank separated representation and so we expect this to simply converge monotonically as we increase the rank of our approximation.

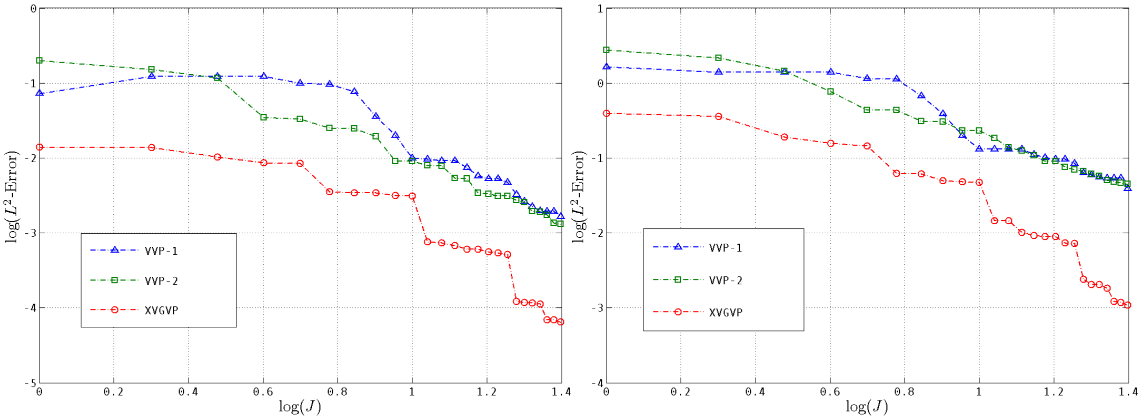
Figure 3.10 shows the convergence in the rank, J, for both the non-homogeneous elliptic VVP least-squares PGD algorithms. We used a spectral element discretisation with degree $N = 8$ polynomials on $K_x = K_y = 3$ elements in each coordinate direction. Here VVP-1 and VVP-2 denote the methods based on the least squares functionals $\mathcal{J}_1(\mathbf{u}, \omega, p)$, (3.66), and $\mathcal{J}_2(\mathbf{u}, \omega, p)$, (3.67), respectively. We observe disappointing rates of convergence in both cases, in particular for the vorticity and pressure since these are the dependent variables which appear in the minus one norm in the continuous least-squares estimate (3.65). We also note that there is no significant difference in the rates of convergence for these two methods.



(a) VVP-1 (b) VVP-2

Figure 3.10: Convergence in Rank of VVP Least-Squares PGDs

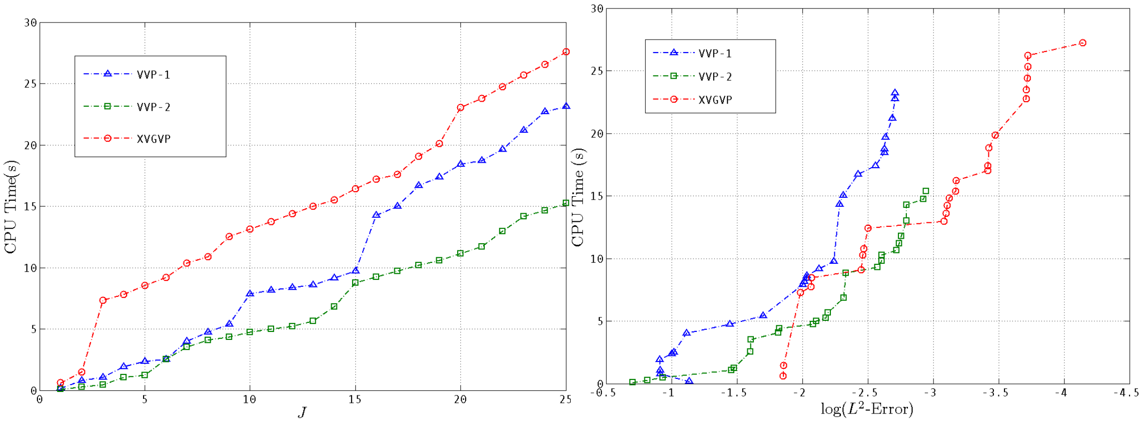
Figure 3.11 shows convergence in the rank for the velocity and pressure in all three least-squares PGD algorithms for the Stokes problem. We have only compared these dependent variables since they are the only ones shared by both the VVP and VGVP systems. It is clear from this that the algorithm based on the homogeneous elliptic extended VGVP formulation (XVGVP) displays the best rate of convergence.



(a) Velocity (b) Pressure

Figure 3.11: Comparison of Least-Squares PGDs for Example 9

Unfortunately, none of the algorithms captured the natural rank-1 separated form of the true solution to the velocity. This is particularly disappointing when we take into consideration that, for the same problem, the Galerkin progressive PGD algorithm (see Figure 2.14 in Section 2.5.4) we were able to capture the rank-1 nature of the velocity and furthermore we observed better rates of convergence in the pressure. However, the key point is that the Galerkin PGD algorithm for the Stokes problem was very unreliable and failed to run for the majority of choices of the discretisation parameters. The least-squares PGD algorithms, on the other hand, worked consistently for any choice of discretisation. This is due to the least-squares formulation not suffering from the stability issues related to the weakly coercive Galerkin formulation of the problem. This makes these algorithms more reliable and hence more efficient despite what may initially seem like disappointing rates of convergence.



(a) Increasing Rank

(b) Decreasing Error

Figure 3.12: CPU Time for Example 9

In Figure 3.12, we have plotted the CPU time for each of the three algorithms. Despite the fact the XVGVP algorithm involves linear systems almost twice the size of the VVP algorithms it is still has a runtime which is only slightly slower than the algorithms based on the much smaller VVP system. In Figure 3.12(b) we show how the runtime increases as the error decreases. This shows that the VVP-2 and XVGVP are actually more on par in terms of their convergence. However, this mainly seems to be caused by particularly expensive steps in terms of CPU time yielding very little reduction in error, especially the third iteration in the XVGVP.

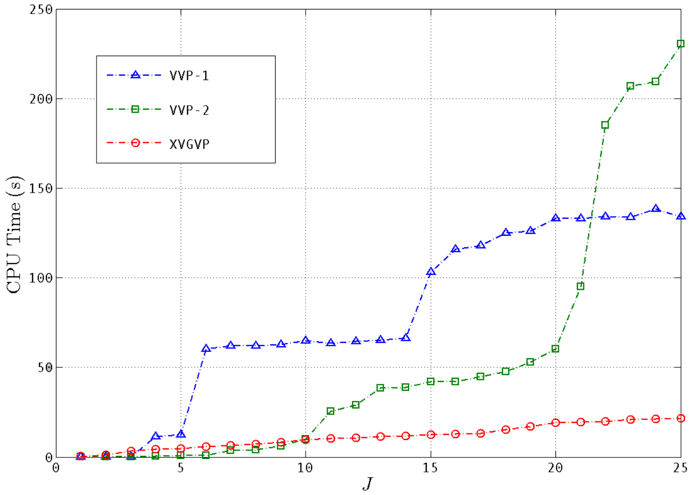


Figure 3.13: CPU Time for Example 9 Without Zero Mean Pressure

Finally, in Figure 3.13, we have plotted the CPU time for the same problem without the zero mean pressure constraint imposed implicitly. Due to the iterative nature of the PGD we are still able to obtain a solution for the pressure and we can then simply modify the solution to have zero mean afterwards by adding a suitable constant. The reason for showing this plot is that the imposition of the zero mean pressure yields a linear system for the pressure which involves a full matrix. It is then reasonable to assume that this may result in a computationally slower algorithm. However, as we can see from Figure 3.13, this is not the case. The VVP algorithms in particular are significantly slower whereas the speed of the XVGVP algorithm is relatively unaltered. The large runtime increase of the VVP algorithms is due to the algorithm getting stuck in the alternating directions linearisation. This can be seen by the large jumps in Figure 3.13. In fact the VVP algorithms needed to have a very coarse convergence criterion in the linearisation in order to make them run at all. This not only highlights the importance of imposing the zero mean pressure condition implicitly but also highlights issues related to the non-homogeneous elliptic VVP formulations.

Example 10 (Rank-1 Pressure Solution)**.**

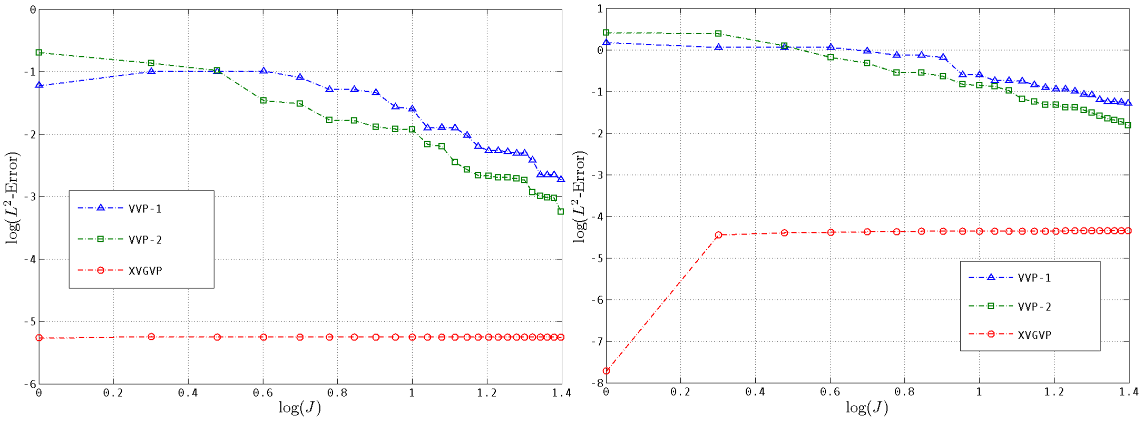
To test our reasoning for the algorithms not capturing the rank-1 nature of the velocity solution we now consider following example on the same domain as Example 9 with homogeneous boundary conditions and source term:

$$\mathbf{f}(x,y) = \begin{pmatrix} \pi\cos(\pi x)\sin(\pi y) + 4\pi^2\sin(2\pi y)(2\cos(2\pi x)-1) \\ \pi\cos(\pi y)\sin(\pi x) - 4\pi^2\sin(2\pi x)(2\cos(2\pi y)-1) \end{pmatrix}.$$

This has the following exact solution:

$$\mathbf{u} = \begin{pmatrix} -\sin(2\pi y)(\cos(2\pi x)-1) \\ \sin(2\pi x)(\cos(2\pi y)-1) \end{pmatrix},$$

$$p = \sin(\pi x)\sin(\pi y).$$



(a) Velocity

(b) Pressure

Figure 3.14: Comparison of Least-Squares PGDs for Example 10

In Figure 3.14 we have plotted the convergence in the rank of the velocity and pressure for all three least-squares Stokes algorithms. We used the same discretisation as in the previous example. From this plot we can see that the only algorithm which was able to capture the rank-1 nature of the velocity and pressure is the homogeneous elliptic XVGVP algorithm. Indeed, we once again find that the non-homogeneous elliptic VVP algorithms display very poor rates of convergence particularly for the pressure. We also note that after the second iteration of the PGD the XVGVP algorithm obtains a solution to the pressure which is significantly worse than the previous iteration. We are not sure what the reason for this might be but in a practical situation it would not be an issue since our global convergence criterion would be satisfied after the first iteration of the PGD and this increase in error of the pressure would not be experienced.

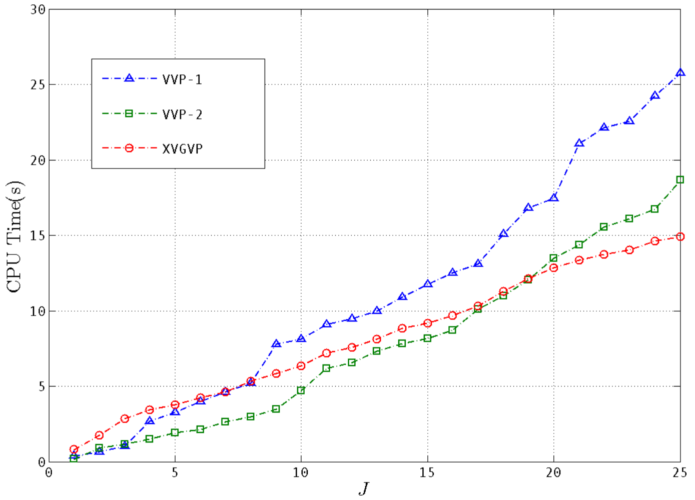


Figure 3.15: CPU Times for Example 10

Figure 3.15 displays the CPU times for the three least-squares algorithms for this example. From this we can see that the XVGVP algorithm now displays a runtime which is much more competitive with the VVP algorithms than in the previous example. Combined with the extremely better rate of convergence of the XVGVP algorithm it is clear that in this case it is by far the superior method.

Unlike the previous example, we will not show the CPU times when the zero mean pressure condition is not included implicitly. The reason for this is that without the implicit zero mean pressure we find that the algorithms fail to converge after a certain small number of PGD iterations. This in itself again indicates the importance of the zero mean pressures inclusion in this way.

3.6.4 Conclusions

We found that the difference between algorithms based on homogeneous elliptic and non-homogeneous elliptic systems are even more significant in the Stokes problem than they were for the Poisson equation. Indeed, we find that the homogeneous elliptic XVGVP system yielded superior rates of convergence in both velocity and pressure as well as being able to capture the rank-1 nature of the solution in

Example 10. We also found that, despite the much larger involved linear systems, the XVGVP algorithm displayed run times comparable with the VVP algorithms. This is a significant piece of evidence that homogeneous elliptic systems are crucial to constructing efficient least-squares PGD algorithms. This strengthens the conclusions we made previously.

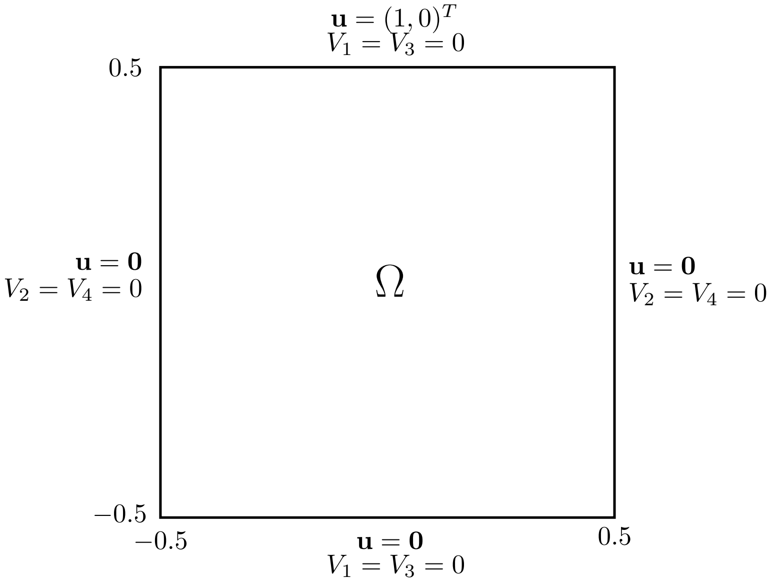
We also noted the significance of including the zero mean pressure constraint implicitly. Without it we found the VVP algorithms to be much slower and in the case of Example 10 the algorithms even failed to converge in the linearisation. This highlights the importance of having an underlying coercivity estimate since it was a requirement of Theorem 5 for such a coercivity estimate to exist we require the solution to be unique. In the case of the Stokes problem this meant we needed the pressure to have a unique solution which we could enforce by including the zero mean pressure constraint in the least-squares functional. Furthermore, the runtime of the VVP algorithms suffered considerably more than the XVGVP algorithm which is further evidence that these non-homogeneous elliptic algorithms are inferior to the homogeneous elliptic XVGVP algorithm.

In contrast with the Galerkin PGD algorithm for the Stokes equations in Section 2.5, we noticed that the least-squares algorithms worked consistently for any choice of discretisation provided that the zero mean pressure was imposed implicitly. This means that least-squares PGD algorithms are far more reliable and hence more efficient than Galerkin PGD algorithms for the Stokes problem. This strengthens our hypothesis that the unreliability of the Galerkin PGD algorithm was down to a lack of LBB-like stability. This is because there are no longer any stability conditions that need to be satisfied when considering a least-squares formulation.

So far we have only considered problems where an analytical solution exists. We now wish to test our XVGVP algorithm for a problem where the solution is not analytical. In particular we will consider the benchmark problem of the lid driven cavity.

3.6.5 Lid Driven Cavity Problem

The lid driven cavity problem is one of the most common benchmark problems used for verifying fluid dynamics models. We will be testing the best of our algorithms, the XVGVP algorithm, from the previous section. Consider the cavity defined by the square domain $\Omega = [-0.5, 0.5]^2$:



Note that the boundary conditions on the velocity are discontinuous in the top corners of the cavity. This is the so called singular lid driven cavity problem. This is much more problematic to solve, in particular for the PGD, when it comes to constructing a suitable transfinite interpolating function which satisfies the boundary conditions. Hence we regularise the problem. To do this we shall replace the velocity boundary condition on the top of the cavity by the following approximation used by Shankar [120] which is continuous along the whole boundary:

$$\mathbf{u} = (\mathcal{F}(x;\delta), 0)^T,$$

where

$$\mathcal{F}(x;\delta) = \begin{cases} \frac{1}{2}(1 + \cos[\frac{\pi}{\delta}(x + \frac{1}{2}(1 - 2\delta))]), & \text{if} \quad x \in [-0.5, -0.5 + \delta], \\ 1, & \text{if} \quad x \in [-0.5 + \delta, 0.5 - \delta], \\ \frac{1}{2}(1 + \cos[\frac{\pi}{\delta}(x - \frac{1}{2}(1 - 2\delta))]), & \text{if} \quad x \in [0.5 - \delta, 0.5], \end{cases}$$

and where $\delta > 0$ is some regularisation parameter. Since we have changed this boundary condition we must also change the boundary condition on the top of the cavity for V_1. This is to ensure that the problem still satisfies the extended VGVP formulation. This boundary condition is replaced by $V_1 = \mathcal{F}'(x;\delta)$, where:

$$\mathcal{F}'(x;\delta) = \begin{cases} -\frac{\pi}{2\delta}\sin[\frac{\pi}{\delta}(x + \frac{1}{2}(1 - 2\delta))], & \text{if} \quad x \in [-0.5, -0.5 + \delta] \\ 0, & \text{if} \quad x \in [-0.5 + \delta, 0.5 - \delta], \\ -\frac{\pi}{2\delta}\sin[\frac{\pi}{\delta}(x - \frac{1}{2}(1 - 2\delta))], & \text{if} \quad x \in [0.5 - \delta, 0.5]. \end{cases}$$

In standard implementations of the lid driven cavity problem using the usual second order form of the Stokes equations one would expect, as $\delta \to 0$, the regularised problem will approach the singular problem. Unfortunately in the extended VGVP formulation, as $\delta \to 0$, we notice that the boundary condition on the velocity gradient component, V_1, will blow up near the top corners of the cavity. For this reason the extended VGVP formulation is not completely well suited for use in this regularised lid driven cavity problem. Indeed, this would be case for any regularisation of the velocity boundary condition due to the inherent steep gradients near the top corners of the cavity. We will proceed by choosing δ carefully so that it is not too small to yield inaccuracies in the velocity gradient while not being too large to cause inaccuracies in the velocity.

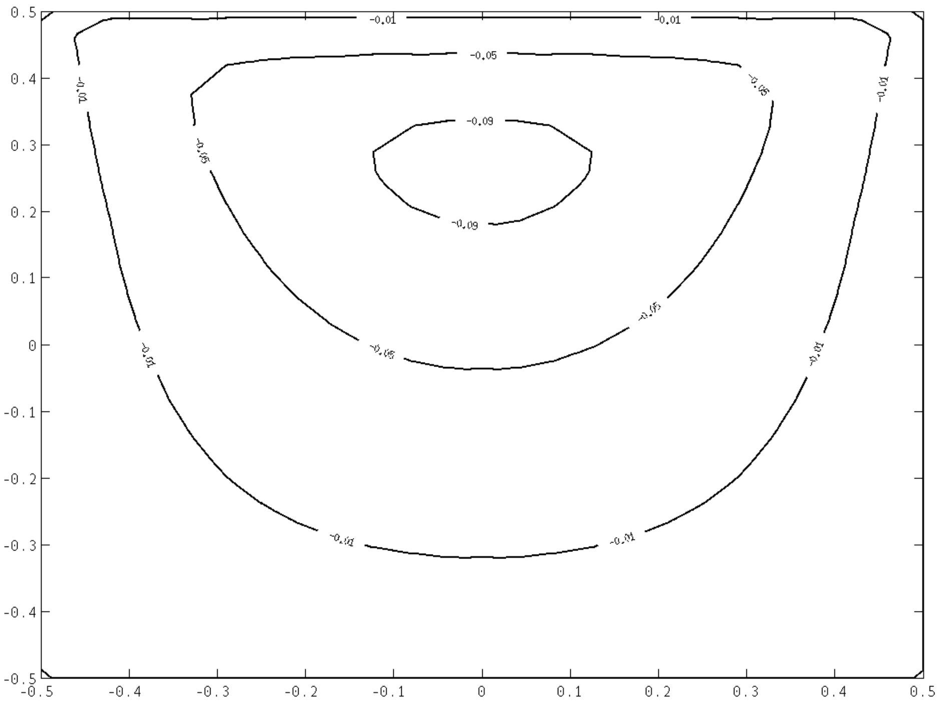


Figure 3.16: Stream Function for Stokes Flow Lid Driven Cavity Problem

Figure 3.16 shows the resulting stream function using our XVGVP least-squares PGD algorithm with a spectral element discretisation of $K_x = K_y = 4$ elements in each coordinate direction with degree $N = 8$ Legendre polynomial basis functions. The number of PGD modes we took was $J = 25$ and we used the value $\delta = 0.05$ for the regularisation parameter.

The results for the primary eddy structure agree excellently with the results for the same problem by Shankar [120]. However, the smaller corner eddies do not appear to be as accurate. While they do appear in our results they are a lot smaller than the corner eddies found by Shankar. We believe this may either be an indication that not enough PGD modes have been taken to accurately capture these subtle features or it is simply an artifact of the unsuitability of the extended VGVP formulation for the regularised problem. We also observed similar eddies appearing in the top corners of the cavity in our results which can be dismissed as erroneous since, due to the regularisation, we can expect to obtain high error in a neighbourhood of these points.

We also noted that the problem was very sensitive to the choice of the regularisation parameter δ which is unsurprising due to the previously mentioned inherent issues with regularised boundary conditions in the extended VGVP formulation. The problem was further sensitive to the choice of discretisation which can most likely be attributed to the same issue. Despite these issues we were able to obtain some very promising results for this problem in particular for the primary eddy structure which is the most significant aspect of the solution. This is a promising result and it would be interesting to see how well the algorithm performs for more complex practical problems.

3.7 Conclusions and Future Work

In this chapter we have reviewed the theory behind least-squares methods and in particular have demonstrated how rigorous least-squares estimates can be derived. We have shown that PGD algorithms based on these formulations can be proven to converge with the proofs crucially relying on the derived coercivity estimates. This suggests that, in order to gain the most benefit from employing least-squares PGD algorithms, these estimates should be preserved in the actual implementation of the method.

Indeed, throughout this chapter we have demonstrated that a crucial component to constructing efficient least-squares PGD algorithms is homogeneous ellipticity of the underlying system. What we really mean by this is that in order to construct an efficient least-squares PGD algorithm we require that the discrete, low-rank, least-squares estimate sufficiently represents the underlying continuous least squares estimate. When the system is homogeneous elliptic this is certainly the case since the continuous least-squares estimate only involves L^2-norms of the differential operators. This is then extremely simple to implement discretely while still retaining the continuous estimate. However, homogeneous ellipticity is a term only associated with the estimates derived from the ADN theory. The ADN theory is a very powerful tool since it reduces verification of continuous estimates to the verification of some algebraic conditions but at the cost of only being applicable to a particular class of problems. Indeed, recall that the ADN theory only applies to linear elliptic PDEs with standard boundary conditions. The question is then how can one obtain estimates for problems which do not fit into this class of problems?

For problems with non-standard boundary conditions (that is when we do not have the same type of boundary condition over the whole boundary) we have already mentioned that continuous estimates can be derived in an ad hoc manner from the vector-operator setting of the problem. That is to say, instead of considering least-squares estimates involving H^1 and L^2 norms, we consider estimates based on vector-operator norms such as the $H(\mathrm{div})$ or $H(\mathrm{curl})$ norms (see [25] for more information). It would then be of interest for us to see how well least-squares PGDs perform for problems of this type.

For nonlinear problems, what is done in practice is that least-squares methods derived for the linear part of the equation are used for the nonlinear equation. For example, least-squares methods for the Navier-Stokes equations simply make use of the least-squares formulations derived for the Stokes problem. While this may not sound particularly rigorous it is justified by the abstract nonlinear approximation theory of Brezzi et al. [32]. In this paper they show that for nonlinear equations that are compact perturbations of a linear operator, the functional setting of the nonlinear equation is governed by that of the linear part. Error estimates can then be established provided that the nonlinear equations satisfy certain assumptions (see [25] for details). This has most notably been successively applied to the Navier-Stokes equations (see e.g. [22], [25], [79]) and it would be very interesting to see how least-squares PGD algorithms for the Navier-Stokes equations perform especially since it will become further unclear how the continuous estimates are preserved. It has also been noted by Bochev and Gunzburger [25] that the extended VGVP formulation performs poorly for the Navier-Stokes equations when the solution is not sufficiently smooth which in the PGD setting may outweigh the benefits of having a homogeneous elliptic system.

Finally, there are also ways to apply least-squares methods for non elliptic problems. For parabolic PDEs it is most often the case that the problem is time dependent and involves a first-order time derivative. For example, the heat equation:

$$\frac{\partial u}{\partial t} - \nabla^2 u = 0,$$

where $u = u(\mathbf{x}, t)$. In this case we can use a semi-discretisation in time such as a backwards Euler method whereby we discretise in time so that at each timestep, $t_i = i\Delta t$, we solve the following elliptic problem for $u_i = u(\mathbf{x}, t_i)$:

$$\frac{1}{\Delta t} u_i - \nabla^2 u_i = \frac{1}{\Delta t} u_{i-1}.$$

The ADN theory can then be applied to this series of elliptic problems to derive estimates upon which least-squares methods can be built. However, it is also possible to apply a least-squares method to the parabolic problem using space-time elements (see e.g. Bell and Surana [17]). These methods have a variety of difficulties surrounding them but are also showing great promise and it would certainly be of interest to see how least-squares PGDs perform in this case. As for hyperbolic PDEs, there has been less success in applying least-squares methods. While some techniques have been applied including minimisation in Hilbert space norms

(Bochev and Gunzburger [25]) as well as Banach space norms (Guermond [75]) these methods currently can not compete with other specialised methods for solving hyperbolic PDEs.

Other future work for least-squares PGDs include the previously mentioned stabilisation of convection dominated problems and also the application of least-squares PGDs to other nonsymmetric problems. From a theoretical perspective we would also like to obtain error estimates specific to the convergence rate of least-squares PGDs and furthermore establish how the continuous estimates in non-homogeneous elliptic systems are affected by the rank of the separated representation in the PGD.

In the next chapter we turn our attention to a practical application of the PGD. So far we have mainly focused on applying the PGD to the Stokes problem. While this can be used to obtain efficient 2D or fully 3D models of creeping flow, the dimensionality of the problem is limited to the three physical spatial directions. Recall that the original purpose of the PGD was to be used for efficiently solving problems defined in high-dimensional space. For this reason we now return to the problem discussed in the first papers describing the PGD algorithm by Ammar et al. [8, 9]. In these papers the authors sought a PGD approximation of the solution to the Fokker-Planck equation which arises in the kinetic theory modeling of dilute polymers. This is a potentially high-dimensional problem and hence the PGD is particularly well suited. For this problem we return to the progressive Galerkin PGD although the possibility of a least-squares PGD is also discussed.

Chapter 4

An Application of the PGD to Kinetic Theory Models in Polymer Rheology

4.1 Introduction

Kinetic theory modelling provides a finer level of description of a fluid than that of a continuum mechanics approach while being sufficiently coarse-grained, in comparison to atomistic or quantum mechanics approaches, to provide computable solutions. The main idea of kinetic theory modelling in polymer rheology is to provide a description of the microstructure of the polymer chains. This began life with simple dumbbell models and has expanded into other models such as bead-rod chains and bead-spring chains as well as a variety of more complex models. An excellent resource on such models is the book of Bird et al. [21].

In this chapter we will be concerned with bead-spring models. In particular we are interested in so-called FENE models (finitely extensible non-linear elastic). These models have a spring force law that has a maximum extension unlike Hookean springs where it is possible for the springs to extend infinitely. Note that FENE models are not the only models to incorporate finite extensibility of the springs, indeed, one could also consider CPAIL models [55] (Cohen's Padé approximant to the Inverse Langevin), for example. Hookean dumbbell models (two beads attached by a spring) have been used extensively in modelling viscoelastic fluids. It is mathematically equivalent to the Oldroyd-B model derived by Oldroyd in 1950 [107]. This model is particularly attractive since one is able to obtain a closed form constitutive equation for the description of the flow. However, a drawback of this model is that the extensional viscosity blows up at a finite extensional rate due to the infinite extensibility of the Hookean springs (see e.g. Owens and Phillips [109]). FENE springs do not suffer from this drawback but, on the other hand, we are unable to find an equivalent closed form constitutive equation. One way to combat this is to use so called closure approximations. These are approximations of the FENE spring force law which do yield a closed form constitutive equation. Examples of such closure approximations are the FENE-P and FENE-CR models (see e.g [109]). However, it was agreed in the IWMMCOF'06 conference [121] that these models are unsympathetic to the physics behind the problems and in some cases they fail to even agree qualitatively with experimental data. For this reason we must consider alternative methods of applying FENE models.

These alternative methods involve coupling the microscale kinetic theory models with macroscale continuum mechanics to model complex flows of viscoelastic fluids. These types of methods are aptly named micro-macro methods and a variety of different approaches have been applied in order to solve them. A review of these methods was compiled by Keunings [81] as well as a more recent review by Lozinski et al. [100]. The key component of micro-macro methods is that the microscale kinetic theory model is solved to yield the polymeric contribution to the stress tensor which then feeds into the macroscale continuum mechanics model. This can either be done deterministically by directly solving the Fokker-Planck equation or by using a stochastic approach.

Until recently the stochastic approach has been the preferred option. Indeed, much progress has been made in this area since the introduction of the CONNFFESSIT method (Calculation of Non-Newtonian Flow: Finite Elements and Stochastic Simulation Technique) by Laso and Öttinger [93] in 1993. Unfortunately these stochastic methods suffer from issues related to statistical noise (see e.g. [81]). It is for this reason that the deterministic approach becomes desirable.

Solving the deterministic Fokker-Planck equation, however, is not so simple. Indeed, as the number of beads in the bead-spring chain is increased the dimension of the problem also increases. Therefore for models such as a generalised FENE Rouse chain (see e.g. [109]) we would expect the Fokker-Planck equation to be high dimensional. This has made the deterministic approach to micro-macro methods difficult to implement. Indeed, most computational results so far have only considered low-dimensional dumbbell models (see e.g. Chauvière and Lozinski [44]). However, as we have already seen, the PGD has developed into a powerful tool for solving problems defined in high-dimensional spaces. In fact, the first paper proposing the PGD by Ammar et al. [8] was in the context of solving the Fokker-Planck equation and the authors went on further to apply the PGD to the transient problem [9]. The PGD has also been applied to the Fokker-Planck equation by Leonenko and Phillips [96] as well as an application of this to the startup of Couette flow in a FENE fluid [97]. There has also been theoretical progress on this problem with a proof of convergence of a PGD algorithm for the Fokker-Planck equation by

Figueroa and Süli [69].

One of the first derivations of the Fokker-Planck equation in polymer rheology can be found in the book of Bird et al. [21] under the name “the diffusion equation”. However, the derivation of this form of the Fokker-Planck equation assumes that the flow is homogeneous and as a result is not suitable for coupling with a Navier-Stokes continuum model within a micro-macro method framework. Indeed, this equation does not take into account the movement of the centres of mass of the polymer molecule models within physical space and only describes the evolution of a probability density function in the (potentially high-dimensional) configuration space relevant to the kinetic theory model under consideration. However, it will still be of interest to us to consider this problem in isolation since it will provide a platform for us to test our PGD algorithm on a simpler problem. When it comes to applying micro-macro methods we will instead need to consider the fully non-homogeneous Fokker-Planck equation (full Fokker-Planck equation). Derivations of this equation can be found in the papers of Lozinski et al. [99] and Barrett and Süli [16]. The full Fokker-Planck equation is defined in both configuration and physical space and can be solved using an operator splitting scheme (see e.g. [44]) in which we solve problems defined in physical space and configuration space separately. Unfortunately, as we shall show, this method is not applicable when using a PGD in configuration space. As a result we instead propose a PGD approximation of the full Fokker-Planck equations where the physical variable is included in our separated representation.

4.2 The Fokker-Planck Equation in Configuration Space

We consider the bead-spring model shown in Figure 4.1. We define the end-to-end vectors of the springs by $\mathbf{q}_i = \mathbf{r}_{i+1} - \mathbf{r}_i$, $i = 1, \ldots, d$, which will become the independent variables in the Fokker-Planck equation. Considering for the time being a simple dumbbell model ($d = 1$) then the dimensionless form of the Fokker-Planck equation is given by the following (see e.g. [100]):

$$\frac{\partial \psi}{\partial t} = -\nabla_{\mathbf{q}} \cdot \left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}\psi - \frac{1}{2\,\mathrm{We}}\mathbf{F}(\mathbf{q})\psi - \frac{1}{2\,\mathrm{We}}(\hat{\mathbf{q}}\hat{\mathbf{q}} + \sigma(\underline{\boldsymbol{\delta}} - \hat{\mathbf{q}}\hat{\mathbf{q}})) \cdot \nabla_{\mathbf{q}}\psi \right), \tag{4.1}$$

where for simplicity we have used the notation $\mathbf{q} = \mathbf{q}_1$ and $\hat{\mathbf{q}}$ denotes the unit vector in the direction of $\mathbf{q}$. Boundary and initial conditions as well as the relevant configuration space will be specified later.

In equation (4.1), $\psi = \psi(\mathbf{q}, t)$ represents a probability distribution function (pdf) of the probability of observing different configurations of the spring. We also define

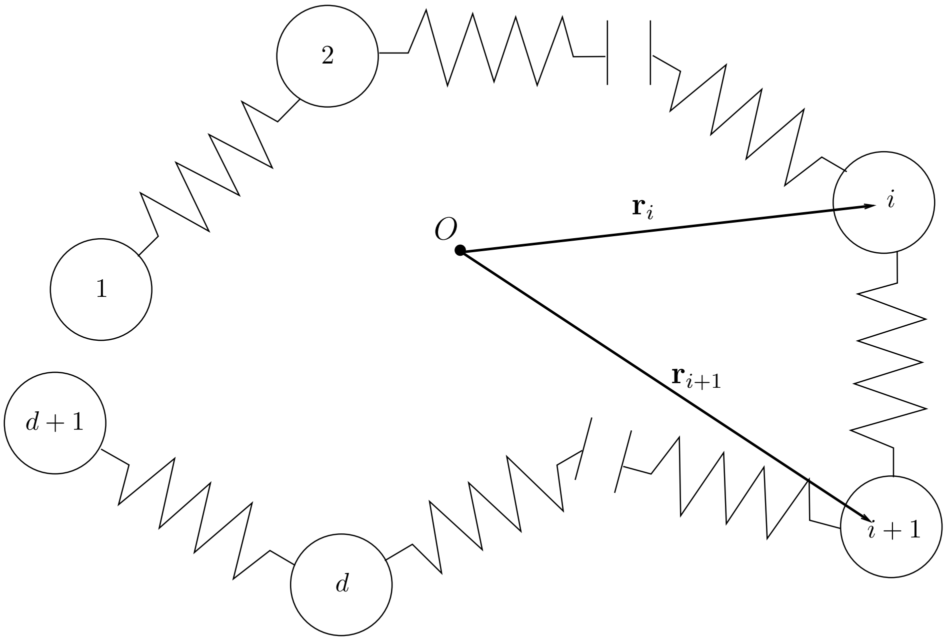


Figure 4.1: A General $(d+1)$-Bead-Spring Model

$\underline{\boldsymbol{\kappa}} = \nabla \mathbf{u}$, which is the given velocity gradient at the centre of mass of the dumbbell, $\mathbf{F}$, which is the force law used for the spring, We, which is the Weissenberg number and σ, which is a parameter related to anisotropic effects. Here $\underline{\boldsymbol{\delta}}$ simply refers to the unit tensor.

The velocity gradient, $\underline{\boldsymbol{\kappa}} = \underline{\boldsymbol{\kappa}}(t)$, in the configurational Fokker-Planck equation (4.1) is independent of $\mathbf{x}$ (the physical coordinate). For this reason it is not suitable for coupling with a macroscale continuum model such as Navier-Stokes since it assumes that the flow is globally homogeneous. Furthermore, we will assume the velocity gradient satisfies $\mathrm{Tr}(\underline{\boldsymbol{\kappa}}) = 0$ (i.e. the fluid is incompressible).

Equation (4.1) is derived from the equation of motion in which three contributions to the force are considered:

1. The hydrodynamic drag force on the beads due to the solvent.

2. The intramolecular force between the two beads (i.e. the spring force).

3. Brownian motion.

It is also possible to include external forces such as gravity but for simplicity we shall assume there are no external forces.

Furthermore, to keep things simple we assume that the fluid isotropic. This assumption is equivalent to setting the parameter $\sigma = 1$, where for anisotropic fluids we have $\sigma < 1$. This assumption is valid when we are considering a dilute polymer. If one were to consider concentrated polymers or polymer melts then these should be considered to be anisotropic due to the interaction between different molecules. Note that if one wished to model anisotropic fluids then models based on reptation such as the Doi-Edwards model [58] are preferred over bead-spring models. The PGD has also been applied to such reptation models by Mokdad et al. [103].

This simplifying assumption ($\sigma = 1$) reduces the Fokker-Planck equation for a dumbbell (4.1) to the following equation:

$$\frac{\partial \psi}{\partial t} = -\nabla_{\mathbf{q}} \cdot \left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}\psi - \frac{1}{2\,\mathrm{We}}\mathbf{F}(\mathbf{q})\psi - \frac{1}{2\,\mathrm{We}}\nabla_{\mathbf{q}}\psi \right). \tag{4.2}$$

This then extends to the following equation based on the multi-bead-spring model in Figure 4.1 by (see e.g. [100]):

$$\frac{\partial \psi}{\partial t} = -\sum_{i=1}^{d} \nabla_{\mathbf{q}_i} \cdot \left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}_i\psi - \frac{1}{4\,\mathrm{We}} \sum_{j=1}^{d} A_{i,j} \Big(\mathbf{F}(\mathbf{q}_j)\psi - \nabla_{\mathbf{q}_j}\psi \Big) \right), \tag{4.3}$$

where A is the $d \times d$ Rouse matrix [115]:

$$A = \begin{pmatrix} 2 & -1 & 0 & \cdots & \cdots & 0 \\ -1 & 2 & -1 & \ddots & 0 & \vdots \\ 0 & -1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -1 & 0 \\ \vdots & 0 & \ddots & -1 & 2 & -1 \\ 0 & \cdots & \cdots & 0 & -1 & 2 \end{pmatrix}.$$

4.2.1 The FENE Model

The FENE model is incorporated into the Fokker-Planck equation by the relevant choice of the force law $\mathbf{F}$. For example, in the simple case of Hookean springs, one would use the dimensionless force law:

$$\mathbf{F}(\mathbf{q}) = \mathbf{q},$$

However, as we have mentioned before using this model allows the springs to extend infinitely. The FENE spring force law, first introduced by Warner in 1972 [129], is given by:

$$\mathbf{F}(\mathbf{q}) = \frac{\mathbf{q}}{1 - (q^2/b)},$$

where $q = |\mathbf{q}|$ and where $\sqrt{b} > 0$ is the dimensionless maximum extensibility of the spring. It is clear from this that if a spring were able to extend to $q = \sqrt{b}$ then the spring force would blow up, therefore preventing this from happening. The natural configuration space, Q, for the FENE model such that $\mathbf{q}_i \in Q$ for $i = 1, \ldots, d$, is:

$$Q = \{\mathbf{q} \in \mathbb{R}^n \; : \; |\mathbf{q}| < \sqrt{b}\},$$

where n is the physical dimension of the problem i.e. in 2D Q represents a disc of radius $\sqrt{b}$ and in 3D a sphere of radius $\sqrt{b}$. Note that each of the springs in the chain does not necessarily have the same maximum extensibility $\sqrt{b}$. Therefore for each spring vector $\mathbf{q}_i$, $i = 1, \dots, d$, we define a maximum extensibility $\sqrt{b_i}$ and associated force law:

$$\mathbf{F}_i(\mathbf{q}_i) = \frac{\mathbf{q}_i}{1 - (q_i^2/b_i)},$$

and configuration spaces:

$$Q_i = \{\mathbf{q}_i \in \mathbb{R}^n \ : \ |\mathbf{q}_i| < \sqrt{b_i}\}.$$

We can then define the full configuration space of our FENE bead-spring chain by:

$$\mathcal{Q} = Q_1 \times \cdots \times Q_d.$$

Existence of a unique solution to a stochastic approach to the modelling of FENE polymers when the maximum extensibility is greater than two was proven by Jourdain et al. [80]. This stochastic approach corresponds to the deterministic approach we are considering and hence we shall assume that $b_i \geq 2$, for $i = 1, \dots, d$, from here on.

We now wish to impose boundary conditions and an initial condition on the problem (4.3). These need to be chosen in order to satisfy the following normality condition:

$$\int_{\mathcal{Q}} \psi(\mathbf{q}_1, \dots, \mathbf{q}_d, t) d\mathcal{Q} = 1, \quad \forall t \in [0, T], \tag{4.4}$$

which is needed since ψ is a probability density function. A second property of a probability density function that ψ is required to satisfy is that of non-negativity:

$$\psi(\mathbf{q}_1, \dots, \mathbf{q}_d, t) \geq 0, \quad \forall (\mathbf{q}_1, \dots, \mathbf{q}_d, t) \in \mathcal{Q} \times [0, T]. \tag{4.5}$$

In order to ensure these properties are satisfied we shall assume we have chosen an initial condition, $\psi^0(\mathbf{q}_1, \dots, \mathbf{q}_d) = \psi(\mathbf{q}_1, \dots, \mathbf{q}_d, 0)$, which satisfies non-negativity as well as the normality condition:

$$\int_{\mathcal{Q}} \psi^0(\mathbf{q}_1, \dots, \mathbf{q}_d) d\mathcal{Q} = 1. \tag{4.6}$$

We can then impose the following no-flux boundary conditions on (4.3) to ensure the normality condition (4.4) holds (see e.g. Knezevic [82]):

$$\left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}_i \psi - \frac{1}{4\,\mathrm{We}} \sum_{j=1}^{d} A_{i,j} \left(\mathbf{F}_j(\mathbf{q}_j) \psi - \nabla_{\mathbf{q}_j} \psi \right) \right) \cdot \mathbf{n}_{\partial Q_i} = 0 \quad \text{on} \quad \partial_i \mathcal{Q} \times (0, T], \tag{4.7}$$

where $\partial_i \mathcal{Q} = Q_1 \times \cdots \times \partial Q_i \times \cdots \times Q_d$ and where $\mathbf{n}_{\partial Q_i}$ denotes the unit normal to

∂Q_i for $i = 1, \ldots, d$. Indeed, if we integrate equation (4.3) over configuration space, $\mathcal{Q}$, and apply the divergence theorem we obtain:

$$\frac{\partial}{\partial t}\int_{\mathcal{Q}} \psi \; d\mathcal{Q} = -\sum_{i=1}^{d}\int_{\partial_i \mathcal{Q}} \left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}_i \psi - \frac{1}{4\,\mathrm{We}}\sum_{j=1}^{d} A_{i,j}\Big(\mathbf{F}_j(\mathbf{q}_j)\psi - \nabla_{\mathbf{q}_j}\psi\Big)\right) \cdot \mathbf{n}_{\partial Q_i} \; d\partial_i \mathcal{Q},$$

Imposing the no-flux boundary conditions (4.7) yields:

$$\frac{\partial}{\partial t}\int_{\mathcal{Q}} \psi(\mathbf{q}_1, \ldots, \mathbf{q}_d, t) \; d\mathcal{Q} = 0 \Longrightarrow \int_{\mathcal{Q}} \psi(\mathbf{q}_1, \ldots, \mathbf{q}_d, t) \; d\mathcal{Q} = c \quad \forall t \in [0, T],$$

for some constant c. Using the normalised initial condition ψ^0 in (4.6) we can then deduce that $c = 1$, hence the normality condition (4.4) holds.

The non-negativity condition, (4.5), has been proven to hold by Knezevic and Süli for the weak formulation of the Fokker-Planck equation provided the initial condition satisfies non-negativity as well (see Lemma 3.3 in [84]). Unfortunately this property is no longer guaranteed to hold at the discrete level for a numerical solution. However, numerical experiments in [84] strongly suggest that it is satisfied if the approximation space is sufficiently refined, at least in the case of the spectral Galerkin method employed in said paper. It is also unclear whether non-negativity is satisfied for a PGD approximation of the pdf, ψ. Hence we will also need to perform some numerical experiments to determine whether or not this is the case.

The specific choice of the initial condition ψ^0 is dependent on the specific problem that one is trying to model. For our purposes we will use the initial condition used by Ammar et al. [8,9] in their work on the PGD applied to the Fokker-Planck equation. As such, we assume that the system is evolving from equilibrium state at which point we have a zero velocity gradient ($\underline{\boldsymbol{\kappa}} = \underline{\mathbf{0}}$). Hence the initial distribution, ψ^0, is the solution of the following steady state problem:

$$\sum_{i=1}^{d} \nabla_{\mathbf{q}_i} \cdot \left(\sum_{j=1}^{d} A_{i,j}\Big(\mathbf{F}_j(\mathbf{q}_j)\psi^0 - \nabla_{\mathbf{q}_j}\psi^0\Big)\right) = 0 \quad \text{in} \;\; \mathcal{Q}, \tag{4.8}$$

$$\left(\sum_{j=1}^{d} A_{i,j}\Big(\mathbf{F}_j(\mathbf{q}_j)\psi^0 - \nabla_{\mathbf{q}_j}\psi^0\Big)\right) \cdot \mathbf{n}_{\partial Q_i} = 0 \quad \text{on} \;\; \partial_i \mathcal{Q} \;\; (i = 1, \ldots, d). \tag{4.9}$$

We can solve this equation by first rewriting $(\mathbf{F}(\mathbf{q}_j)\psi^0 - \nabla_{\mathbf{q}_j}\psi^0)$, $j = 1, \ldots, d$, under a single derivative. To do this we use the following integrating factors:

$$I_j(\mathbf{q}_j) = \exp\left(\int \mathbf{F}(\mathbf{q}_j) \cdot \mathbf{dq}_j\right) = \exp\left(\int \frac{\mathbf{q}_j}{1 - (q_j^2/b_j)} \cdot \mathbf{dq}_j\right) = C\left(\frac{1}{1 - (q_j^2/b_j)}\right)^{\frac{b_j}{2}}.$$

for some constant C. By multiplying and dividing through by the integrating factor,

$I_j(\mathbf{q}_j)$, we can rewrite problem (4.8)-(4.9) in the following way:

$$\sum_{i=1}^{d} \nabla_{\mathbf{q}_i} \cdot \left(\sum_{j=1}^{d} A_{i,j} \left(\frac{1}{I_j(\mathbf{q}_j)} \nabla_{\mathbf{q}_j} I_j(\mathbf{q}_j) \psi^0 \right) \right) = 0 \quad \text{in} \quad \mathcal{Q}, \tag{4.10}$$

$$\left(\sum_{j=1}^{d} A_{i,j} \left(\frac{1}{I_j(\mathbf{q}_j)} \nabla_{\mathbf{q}_j} I_j(\mathbf{q}_j) \psi^0 \right) \right) \cdot \mathbf{n}_{\partial Q_i} = 0 \quad \text{on} \quad \partial_i \mathcal{Q} \quad (i = 1, \ldots, d). \tag{4.11}$$

One can then deduce that this is satisfied when:

$$\psi^0(\mathbf{q}_1, \ldots, \mathbf{q}_d) = C \prod_{k=1}^{d} \frac{1}{I_k(\mathbf{q}_k)},$$

since then we have that:

$$\nabla_{\mathbf{q}_j} I_j(\mathbf{q}_j) \psi^0 = \mathbf{0}, \quad \forall j = 1, \ldots, d.$$

The constant C can then be found by imposing the normality condition (4.6). Hence we finally have the following initial condition for the Fokker-Planck equation for the FENE model:

$$\psi^0(\mathbf{q}_1, \ldots, \mathbf{q}_d) = \prod_{i=1}^{d} \frac{(1 - (q_i^2/b_i))^{\frac{b_i}{2}}}{\int_{Q_i} (1 - (q_i^2/b_i))^{\frac{b_i}{2}} \, dQ_i} =: M(\mathbf{q}_1, \ldots, \mathbf{q}_d). \tag{4.12}$$

Note that ψ^0 also satisfies non-negativity (4.5) and hence is a suitable choice of initial condition. The function (4.12) is known as the (normalised) Maxwellian for the FENE model (see e.g. Barrett and Süli [15]), hence the prescribed notation M. The Maxwellian has a much more significant role in the Fokker-Planck equation than just providing a suitable initial condition. Indeed, much in the same way as we did to obtain equations (4.10)-(4.11) we can rewrite the Fokker-Planck equation (4.3) and boundary condition (4.7) as:

$$\frac{\partial \psi}{\partial t} = -\sum_{i=1}^{d} \nabla_{\mathbf{q}_i} \cdot \left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}_i \psi - \frac{1}{4\,\mathrm{We}} \sum_{j=1}^{d} A_{i,j} \left(M \nabla_{\mathbf{q}_j} \left(\frac{\psi}{M} \right) \right) \right) \quad \text{in} \quad \mathcal{Q} \times (0, T], \tag{4.13}$$

$$\left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}_i \psi - \frac{1}{4\,\mathrm{We}} \sum_{j=1}^{d} A_{i,j} \left(M \nabla_{\mathbf{q}_j} \left(\frac{\psi}{M} \right) \right) \right) \cdot \mathbf{n}_{\partial Q_i} = 0 \quad \text{on} \quad \partial_i \mathcal{Q} \times (0, T], \tag{4.14}$$

for $i = 1, \ldots, d$. This transformation was used by Barrett and Süli in order to overcome analytical difficulties related to the unbounded convection term, $\mathbf{F}$, in the standard formulation. For our purposes this transformation provides a platform from which a PGD algorithm can be proven to converge as was done by Figueroa and Süli [69]. We shall give some details of this in a later section. As a result of this we shall now only be considering the Fokker-Planck equation in the Maxwellian transformed form above.

4.2.2 Implementation of the PGD

As with our previous work with the PGD one might expect that we seek a separated representation of the pdf, ψ, of the form:

$$\psi(\mathbf{q}_1, \ldots, \mathbf{q}_d, t) \approx \sum_{j=1}^{J} \prod_{i=1}^{d} \prod_{k=1}^{n} Q_{i,j}^{(k)}(q_i^{(k)}) T_j(t),$$

where we have separated all the components of the end-to-end spring vectors $\mathbf{q}_i = (q_i^{(1)}, \ldots, q_i^{(n)})^T$, $i = 1, \ldots, d$. However, there are two problems with using a separated representation of this form. Firstly, since Q_i, $i = 1, \ldots, d$, are balls, the domain of the problem is not defined in a hyper-rectangle which is the natural setting for the application of PGD algorithms. Secondly, we would also require the Maxwellian to be a function that is separable in the components of each $\mathbf{q}_i$, $i = 1, \ldots, d$. This is clearly not the case for the FENE Maxwellian (4.12). Both these problems could be remedied by converting to polar/spherical coordinates. Indeed, consider, for example, a problem defined in 2D ($n = 2$). In polar coordinates each spring vector $\mathbf{q}_i$ can be expressed in terms of radial and angular components (r_i, θ_i) where we would now seek a separated representation of the form:

$$\psi(\mathbf{q}_1, \ldots, \mathbf{q}_d, t) \approx \sum_{j=1}^{J} \prod_{i=1}^{d} R_{i,j}(r_i) \Theta_{i,j}(\theta_i) T_j(t).$$

The change of coordinates maps the balls Q_i to rectangles $[0, \sqrt{b_i}] \times [0, 2\pi]$, $i = 1, \ldots, d$, and hence the domain of the problem becomes a hyper-rectangle. Furthermore, since $r_i = |\mathbf{q}_i| = q_i$, then the FENE Maxwellian (4.12) now reads:

$$M(\mathbf{q}_1, \ldots, \mathbf{q}_d) = \prod_{k=1}^{d} \frac{(1 - (r_k^2/b_k))^{\frac{b_k}{2}}}{\int_{Q_k} (1 - (q_k^2/b_k))^{\frac{b_k}{2}} \, dQ_k}.$$

This is now a function of the radial terms only and more importantly it now possesses a rank-one separated form. Unfortunately, this introduces additional complications in that the change of coordinates introduces singularities at the poles $r_i = 0$, $i = 1, \ldots, d$. Normally what is done for problems of this type is that additional so called pole conditions are added to ensure regularity of the solution at the poles (see e.g. Shen [122]). However, it is unclear in the PGD how these should be imposed and furthermore it is unclear how the periodicity in the θ_i directions can be imposed. For these reasons we will instead consider a separated representation of the form:

$$\psi(\mathbf{q}_1, \ldots, \mathbf{q}_d, t) \approx \sum_{j=1}^{J} \prod_{i=1}^{d} Q_{i,j}(\mathbf{q}_i) T_j(t),$$

where $\mathbf{q}_i \in Q_i \subset \mathbb{R}^n$ for $i = 1, \ldots, d$, where in practice n is a moderate dimension (at most 3). Since the dimension of each of the separated variables is still moderate it means that the PGD will still provide an exceptional amount of computational saving. This separated representation makes use of the d-term cartesian product space $\mathcal{Q}$ and hence it is practical to implement in the PGD. We also note that the FENE Maxwellian M has a rank-one separated representation of this form:

$$M(\mathbf{q}_1, \ldots, \mathbf{q}_d) = \prod_{i=1}^{d} M_i(\mathbf{q}_i),$$

where M_i, $i = 1, \ldots, d$, are known as the partial FENE Maxwellians defined by:

$$M_i(\mathbf{q}_i) = \frac{(1 - (q_i^2/b_i))^{\frac{b_i}{2}}}{\int_{Q_i} (1 - (q_i^2/b_i))^{\frac{b_i}{2}} \, dQ_i}.$$

A second consideration is that of the inclusion of time in the separated representation. Indeed, while the transient space-time problem has been considered in the PGD by Ammar et al. [9] it requires the solution of a parabolic problem and one can experience stability issues related to the inclusion of time derivative. A second possibility is to use an incremental time discretisation where one would typically employ a backwards Euler scheme so that at the $(k+1)^{\text{st}}$ time step we have:

$$\frac{\partial \psi}{\partial t} \approx \frac{\psi^{k+1} - \psi^k}{\Delta t}, \tag{4.15}$$

where $\psi^k = \psi(\mathbf{q}_1, \ldots, \mathbf{q}_d, t^k)$ denotes the solution obtained at the k^{th} time step and Δt is the size of the time step such that $t^k = k\Delta t$ for $k = 0, \ldots, T/\Delta t$. This leads to a series of elliptic problems where at each time step we seek a separated representation of the form:

$$\psi^k(\mathbf{q}_1, \ldots, \mathbf{q}_d) \approx \sum_{j=1}^{J} \prod_{i=1}^{d} Q_{i,j}(\mathbf{q}_i).$$

This has the advantage that it avoids any stability issues related to the parabolic nature of the fully time dependent problem and furthermore yields a problem with which an associated PGD algorithm can be proven to converge.

One of the main focuses of this thesis has been the convergence of PGD algorithms. It is for this reason that we consider the second of these options (i.e. the time stepping scheme) since in this case it is possible to associate the problem with a series of greedy algorithms at each time step. It is then possible to prove convergence of these greedy algorithms as was considered by Figueroa and Süli [69]. In the following section we derive the weak formulation for the Fokker-Planck equation and show how to formulate the problem so that convergence of a PGD algorithm can be proven.

4.2.3 Convergence of Fokker-Planck PGD Algorithms

As mentioned earlier a full space-time PGD requires the solution of a parabolic problem. Proofs of convergence for Galerkin PGDs only cover self-adjoint elliptic problems. To this end we will employ the previously mentioned backward Euler scheme (4.15) which will yield a semi-discrete Fokker-Planck equation which involves a series of elliptic problems. It is also possible to formulate this as a series of self-adjoint elliptic problems by treating the convective terms explictly in time. Stability of this semi-discretisation has been proven in several different norms by Knezevic and Süli [84].

Before deriving the weak formulation of the semi-discrete Fokker-Planck equation we introduce the following Maxwellian weighted Sobolev space (see [84]):

$$H^1(\mathcal{Q};M) := \left\{ \varphi \in L^2(\mathcal{Q}) \ : \ \|\varphi\|^2_{H^1(\mathcal{Q};M)} := \int_{\mathcal{Q}} \left(|\varphi|^2 + \sum_{i=1}^{d} |\nabla^i_M \varphi|^2 \right) \, d\mathcal{Q} < \infty \right\},$$

where we use the notation that:

$$\nabla^i_M \varphi := \sqrt{M} \nabla_{\mathbf{q}_i} \left(\frac{\varphi}{\sqrt{M}} \right), \quad \text{for} \quad i = 1, \ldots, d.$$

We can then derive a weak formulation of the problem, following [84], by multiplying the Maxwellian transformed Fokker-Planck equation (4.13) by a test function ψ^*/M, where $\psi^*/\sqrt{M} \in H^1_0(\mathcal{Q};M)$, and then integrating over configuration space $\mathcal{Q}$. Finally, we apply the backward Euler scheme to obtain the following semi-discrete weak formulation of the Fokker-Planck equation:

For each $k = 0, \ldots, T/\Delta t - 1$, find ψ^{k+1}, where $\psi^{k+1}/\sqrt{M} \in H^1(\mathcal{Q};M)$, such that:

$$\begin{aligned} &\frac{1}{\Delta t} \int_{\mathcal{Q}} \frac{\psi^{k+1}\psi^*}{M} \, d\mathcal{Q} - \sum_{i=1}^{d} \int_{\mathcal{Q}} (\underline{\boldsymbol{\kappa}}^{k+1} \cdot \mathbf{q}_i)\psi^{k+1} \cdot \nabla_{\mathbf{q}_i} \left(\frac{\psi^*}{M} \right) d\mathcal{Q} \\ &+ \frac{1}{4\,\mathrm{We}} \sum_{i,j=1}^{d} A_{i,j} \int_{\mathcal{Q}} M \nabla_{\mathbf{q}_j} \left(\frac{\psi^{k+1}}{M} \right) \cdot \nabla_{\mathbf{q}_i} \left(\frac{\psi^*}{M} \right) d\mathcal{Q} = \frac{1}{\Delta t} \int_{\mathcal{Q}} \frac{\psi^k \psi^*}{M} \, d\mathcal{Q}, \end{aligned}$$

for all ψ^*, where $\psi^*/\sqrt{M} \in H^1_0(\mathcal{Q};M)$, where $\underline{\boldsymbol{\kappa}}^{k+1} = \underline{\boldsymbol{\kappa}}(t^{k+1})$, and where we have made use of the divergence theorem where relevant. If we now use the substitutions $\hat{\psi}^k = \psi^k/\sqrt{M}$ and $\hat{\psi}^* = \psi^*/\sqrt{M}$ then we can write the above weak formulation as:

For each $k = 0, \dots, T/\Delta t - 1$, find $\hat{\psi}^{k+1} \in H^1(\mathcal{Q}; M)$ such that:

$$\frac{1}{\Delta t}\int_{\mathcal{Q}} \hat{\psi}^{k+1}\hat{\psi}^* \, d\mathcal{Q} - \sum_{i=1}^{d}\int_{\mathcal{Q}} (\underline{\boldsymbol{\kappa}}^{k+1} \cdot \mathbf{q}_i)\hat{\psi}^{k+1} \cdot \nabla_M^i \hat{\psi}^* \, d\mathcal{Q}$$
$$+ \frac{1}{4\,\mathrm{We}}\sum_{i,j=1}^{d} A_{i,j}\int_{\mathcal{Q}} \nabla_M^j \hat{\psi}^{k+1} \cdot \nabla_M^i \hat{\psi}^* \, d\mathcal{Q} = \frac{1}{\Delta t}\int_{\mathcal{Q}} \hat{\psi}^k \hat{\psi}^* \, d\mathcal{Q},$$

for all $\hat{\psi}^* \in H_0^1(\mathcal{Q}; M)$. Note that, in these weak formulations, the boundary condition (4.14) has not been explicitly imposed. The reason for this is that when $b \geq 2$ (which is an assumption we previously made due to Jourdain et al. [80]) the boundary condition (4.14) for the weak problem becomes redundant (see e.g. Liu and Liu [98]). However, the Maxwellian weighted Sobolev space we have considered has the interesting property that $H^1(\mathcal{Q}; M) = H_0^1(\mathcal{Q}; M)$ (see e.g. [84]). This means that solutions to the weak problem above are forced to satisfy a homogeneous Dirichlet boundary condition on $\partial\mathcal{Q}$. Intuitively this makes sense since we expect the probability that a spring becomes fully extended to be zero.

We now introduce the following bilinear forms to simplify notation:

$$a(\psi, \psi^*) := \frac{1}{4\,\mathrm{We}}\sum_{i,j=1}^{d} A_{i,j}\int_{\mathcal{Q}} \nabla_M^j \psi \cdot \nabla_M^i \psi^* \, d\mathcal{Q}, \tag{4.16}$$

$$b(\psi, \psi^*; \underline{\boldsymbol{\kappa}}) := \sum_{i=1}^{d}\int_{\mathcal{Q}} (\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}_i)\psi \cdot \nabla_M^i \psi^* \, d\mathcal{Q}. \tag{4.17}$$

The weak formulation of the Maxwellian transformed Fokker-Planck equation can then be written as: For each $k = 0, \dots, T/\Delta t - 1$, find $\hat{\psi}^{k+1} \in H^1(\mathcal{Q}; M)$ such that:

$$\frac{1}{\Delta t}\langle \hat{\psi}^{k+1}, \hat{\psi}^* \rangle + a(\hat{\psi}^{k+1}, \hat{\psi}^*) - b(\hat{\psi}^{k+1}, \hat{\psi}^*; \underline{\boldsymbol{\kappa}}^{k+1}) = \frac{1}{\Delta t}\langle \hat{\psi}^k, \hat{\psi}^* \rangle, \quad \forall \hat{\psi}^* \in H_0^1(\mathcal{Q}; M).$$

In order to prove convergence of a PGD algorithm for this problem we require that it is symmetric (recall the proof of Falcó and Nouy [66]). Unfortunately, this is not true due to the presence of the convective term $b(\cdot, \cdot)$. However, we can remedy this by adjusting our scheme so that this convective term is treated explicitly in time and moved to the right hand side. This leads to the following problem: For each $k = 0, \dots, T/\Delta t - 1$, find $\hat{\psi}^{k+1} \in H^1(\mathcal{Q}; M)$ such that:

$$\frac{1}{\Delta t}\langle \hat{\psi}^{k+1}, \hat{\psi}^* \rangle + a(\hat{\psi}^{k+1}, \hat{\psi}^*) = b(\hat{\psi}^k, \hat{\psi}^*; \underline{\boldsymbol{\kappa}}^k) + \frac{1}{\Delta t}\langle \hat{\psi}^k, \hat{\psi}^* \rangle, \quad \forall \hat{\psi}^* \in H_0^1(\mathcal{Q}; M). \tag{4.18}$$

We can then associate this problem with a greedy algorithm based on the minimisation of the functional:

$$\mathcal{J}_k(\hat{\psi}^{k+1}) = \frac{1}{2}\left(\frac{1}{\Delta t}\langle \hat{\psi}^{k+1}, \hat{\psi}^{k+1} \rangle + a(\hat{\psi}^{k+1}, \hat{\psi}^{k+1})\right) - b(\hat{\psi}^k, \hat{\psi}^{k+1}; \underline{\boldsymbol{\kappa}}^k) - \frac{1}{\Delta t}\langle \hat{\psi}^k, \hat{\psi}^{k+1} \rangle, \tag{4.19}$$

for each $k = 0, \ldots, T/\Delta t - 1$. A proof of convergence for this greedy algorithm can be found in the paper of Figueroa and Süli [69]. We will not outline the complete proof here but it is essentially the verification of the assumptions given in the paper of Cancès et al. [37] for the Maxwellian weighted Sobolev space $H^1(\mathcal{Q}; M)$.

Numerical results of Knezevic [82] suggest that this semi-implicit time discretisation is less stable than the fully-implicit backward Euler scheme. A second option that would provide us with a problem with which we are able to prove convergence of associated PGD algorithms and which is fully-implicit would be to consider a least-squares PGD algorithm for the Fokker-Planck equation. A first order reformulation of the Fokker-Planck equation (4.13) could be made by introducing the following additional dependent variables:

$$\boldsymbol{\chi}_j = -M\nabla_{\mathbf{q}_j}\left(\frac{\psi}{M}\right), \quad j = 1, \ldots, d.$$

This yields the following problem:

$$\begin{aligned} \frac{\partial \psi}{\partial t} = -\sum_{i=1}^{d} \nabla_{\mathbf{q}_i} \cdot \left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}_i \psi + \frac{1}{4\,\mathrm{We}} \sum_{j=1}^{d} A_{i,j} \boldsymbol{\chi}_j \right) \quad &\text{in} \quad \mathcal{Q} \times (0, T], \\ \frac{\boldsymbol{\chi}_j}{M} + \nabla_{\mathbf{q}_j}\left(\frac{\psi}{M}\right) = \mathbf{0} \quad &\text{in} \quad \mathcal{Q} \times (0, T], \quad (j = 1, \ldots, d), \\ \left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}_i \psi + \frac{1}{4\,\mathrm{We}} \sum_{j=1}^{d} A_{i,j} \boldsymbol{\chi}_j \right) \cdot \mathbf{n}_{\partial Q_i} = 0 \quad &\text{on} \quad \partial_i \mathcal{Q} \times (0, T], \quad (i = 1, \ldots, d). \end{aligned}$$

This is essentially a high dimensional time-dependent convection-diffusion equation. Hence we could obtain a homogeneous elliptic system by employing a time integration scheme and by adding the additional redundant equations:

$$\nabla_{\mathbf{q}_j} \times \frac{\boldsymbol{\chi}_j}{M} = \mathbf{0}, \quad \text{in} \quad \mathcal{Q} \times (0, T], \quad (j = 1, \ldots, d).$$

However, there are a number of complications with considering a least-squares PGD of this problem:

1. If d is large the number of additional dependent variables we need also becomes large and hence we would require the solution of much larger linear systems.

2. It is not clear how we can add additional redundant boundary conditions that complement the no-flux boundary condition to obtain a homogeneous elliptic system.

3. From our previous work on least-squares PGDs for the convection-diffusion equation we have seen that algorithms can perform very poorly when the convective term dominates. This would be a problem for us when we have a high Weissenberg number or large velocity gradients.

4. The ADN theory does not provide energy balances in non-standard norms such as the norm on Maxwellian weighted Sobolev space $H^1(\mathcal{Q};M)$ which may prove to be the most convenient functional space for this problem.

For these reasons we believe that, for the time being, a Galerkin PGD algorithm based on the semi-implicit problem (4.18) is the preferable option. Although it would certainly be of future interest to see how well a least-squares PGD algorithm for the Fokker-Planck equation would perform if one were to employ an appropriate stabilisation such as the one presented by Chen et al. [45], where great care is taken to ensure a homogeneous elliptic system is obtained in the relevant norms.

A final consideration before applying this scheme to an example is that of the preservation of the properties of a probability density function within the PGD. For the non-negativity property (4.5) we have already stated that there will be no guarantee that it is satisfied in the PGD and therefore this is something we will need to check via numerical experiments. As for the normality property (4.4) this has been proven to be satisfied for the weak formulation of the Maxwellian transformed Fokker-Planck equation by Knezevic and Süli [84]. To further ensure it is satisfied in the PGD we would require that $\sqrt{M} \in \operatorname{Span}(\mathcal{S}_1)$ where $\mathcal{S}_1$, as before, is the set of all rank-one tensors:

$$\mathcal{S}_1 := \left\{ \psi = \bigotimes_{i=1}^{d} \psi_i \ : \ (\psi_1, \ldots, \psi_d) \in H^1(Q_1; M_1) \times \cdots \times H^1(Q_d; M_d) \right\},$$

where $H^1(Q_i; M_i)$, $i = 1, \ldots, d$, are the partial Maxwellian weighted Sobolev spaces:

$$H^1(Q_i; M_i) := \left\{ \varphi \in L^2(Q_i) \ : \ \|\varphi\|^2_{H^1(Q_i;M_i)} := \int_{Q_i} \left(|\varphi|^2 + |\nabla_{M_i} \varphi|^2 \right) \, dQ_i < \infty \right\},$$

and where we have used the notation:

$$\nabla_{M_i} \varphi := \sqrt{M_i} \nabla \left(\frac{\varphi}{\sqrt{M_i}} \right), \quad \text{for} \quad i = 1, \ldots, d.$$

The reason for this requirement is that in the Galerkin PGD the test functions, ψ^*, are chosen such that $\psi^* \in \operatorname{Span}(\mathcal{S}_1)$. In particular they are chosen to be rank-d tensors of the form (see Ammar et al. [8]):

$$\psi^* = \sum_{j=1}^{d} \bigotimes_{\substack{i=1 \\ i \neq j}}^{d} \psi_i \otimes \psi_j^*.$$

The Galerkin PGD algorithm is then a problem of the form: For each $k =$

$0, \ldots, T/\Delta t - 1$ and each $j = 1, \ldots, J_k$, find $\hat{\psi}_j^{k+1} \in \mathcal{S}_1$ such that:

$$\frac{1}{\Delta t}\langle \hat{\psi}_j^{k+1} + \sum_{i=1}^{j-1} \hat{\psi}_i^{k+1}, \hat{\psi}^* \rangle + a(\hat{\psi}_j^{k+1} + \sum_{i=1}^{j-1} \hat{\psi}_i^{k+1}, \hat{\psi}^*) = b(\hat{\psi}^k, \hat{\psi}^*; \underline{\boldsymbol{\kappa}}^k) + \frac{1}{\Delta t}\langle \hat{\psi}^k, \hat{\psi}^* \rangle,$$

for all $\hat{\psi}^* \in \mathrm{Span}(\mathcal{S}_1)$, where $\hat{\psi}^k \in \mathrm{Span}(\mathcal{S}_1)$ denotes a rank-J_k tensor of the form:

$$\hat{\psi}^k = \sum_{j=1}^{J_k} \hat{\psi}_j^k, \quad \text{where } \hat{\psi}_j^k \in \mathcal{S}_1 \;\; (j = 1, \ldots, J_k).$$

If we have that $\sqrt{M} \in \mathrm{Span}(\mathcal{S}_1)$ then we can take the test function above to be $\hat{\psi}^* = \sqrt{M}$. Noting that $\sqrt{M} \in \mathrm{Ker}(\nabla_M^i)$ for each $i = 1, \ldots, d$ this then yields:

$$\langle \hat{\psi}_j^{k+1} + \sum_{i=1}^{j-1} \hat{\psi}_i^{k+1}, \sqrt{M} \rangle = \langle \hat{\psi}^k, \sqrt{M} \rangle.$$

By induction, we can deduce that:

$$\int_{\mathcal{Q}} \hat{\psi}^k(\mathbf{q}_1, \ldots, \mathbf{q}_d)\sqrt{M(\mathbf{q}_1, \ldots, \mathbf{q}_d)}\; d\mathcal{Q} = \int_{\mathcal{Q}} \hat{\psi}^0(\mathbf{q}_1, \ldots, \mathbf{q}_d)\sqrt{M(\mathbf{q}_1, \ldots, \mathbf{q}_d)}\; d\mathcal{Q},$$

or in terms of the original dependent variables:

$$\int_{\mathcal{Q}} \psi^k(\mathbf{q}_1, \ldots, \mathbf{q}_d)\; d\mathcal{Q} = \int_{\mathcal{Q}} \psi^0(\mathbf{q}_1, \ldots, \mathbf{q}_d)\; d\mathcal{Q}.$$

Therefore the normality condition (4.4) is satisfied under the assumption that we have chosen an initial condition ψ^0 which satisfies the normality condition (4.6). Conveniently, trivially we have that $\sqrt{M} \in \mathrm{Span}(\mathcal{S}_1)$ for the FENE Maxwellian since $M \in \mathcal{S}_1$ then $\sqrt{M} \in \mathcal{S}_1 \subset \mathrm{Span}(\mathcal{S}_1)$. This means that the normality condition is preserved in the PGD, at least for the continuous, nonlinear problem. For an actual implementation of the PGD we additionally require that normality is preserved for the discrete problem as well as in the linearisation.

To ensure normality is preserved in the discrete problem we require that $\sqrt{M}$ is in our chosen discretisation space. One situation in which $\sqrt{M}$ is ensured to be in our chosen discretisation space is when the maximum extensibility of each spring satisfies $b_i = 4m_i$ for some $m_i \in \mathbb{N}$, $i = 1, \ldots, d$. Upon converting to polar/spherical coordinates we then have that:

$$\sqrt{M(\mathbf{q}_1, \ldots, \mathbf{q}_d)} = \prod_{i=1}^{d} \frac{(1 - (r_i^2/4m_i))^{m_i}}{\sqrt{\int_{Q_i}(1 - (q_i^2/4m_i))^{2m_i}\; dQ_i}}.$$

Notice that $\sqrt{M}$ is now a polynomial of degree $2m_i$ in each radial component r_i, $i = 1, \ldots, d$. Hence if we use a spectral element method in the radial directions with polynomials of degree at least $2m_i$, $i = 1, \ldots, d$ then we can ensure that $\sqrt{M}$ is in our discretisation space. In the case when the maximum extensibility of the springs is such that $b_i \neq 4m_i$ for any $m_i \in \mathbb{N}$, $i = 1, \ldots, d$, then one can enrich the discrete basis by including the component of $\sqrt{M}$ which is orthogonal to all elements of the discretisation space. This was successfully implemented by Knezevic and Süli [84] using a standard mesh-based method where the PGD was not employed.

Unfortunately, there is no way to guarantee that normality will be preserved in the linearisation and this is something that will need to be checked in the numerical experiments. We will now provide details showing how our PGD algorithm for the solution of the Fokker-Planck equation in configuration space is implemented by considering a simple example in 1D.

4.2.4 Numerical Experiments

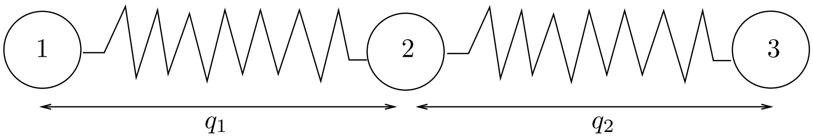
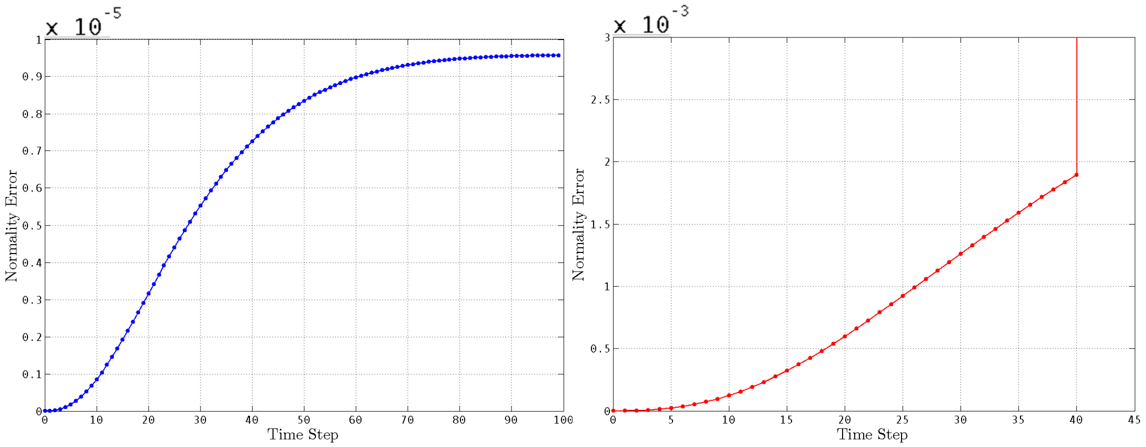


Figure 4.2: A 1D 3-Bead-Spring Model

Consider the solution of a Fokker-Planck equation for the configuration space given by the bead-spring model in Figure 4.2. In this model we assume that the beads are aligned in the y and z spatial coordinates such that the spring vectors reduce to dimensionless scalar extensions q_1 and q_2 as pictured above. Therefore this problem is a spatially 1D problem defined in the configuration space $\mathcal{Q} = [-\sqrt{b_1}, \sqrt{b_1}] \times [-\sqrt{b_2}, \sqrt{b_2}]$. This greatly simplifies the problem since we do not need to worry about converting to polar/spherical coordinates.

In the following numerical experiments we will continue to use a spectral element method for discretisation. It was shown by Knezevic [82] that $H_0^1(\mathcal{Q}) \subset H^1(\mathcal{Q}; M)$ hence we can choose a standard discretisation space which is a subset of $H_0^1(\mathcal{Q})$ with it trivially also being a subset of the Maxwellian weighted Hilbert space $H^1(\mathcal{Q}; M)$.

We begin by checking whether normality is preserved when we are in the situation $b_1 = 4m_1$ and $b_2 = 4m_2$ for $m_1, m_2 \in \mathbb{N}$. To test this we considered the particular case when $m_1 = m_2 = 4$. We then used polynomials of degree $N = 8 = 2m_1 = 2m_2$ on $K_x = K_y = 3$ elements for each scalar spring coordinate seeking rank $J = 1$ PGD approximations. In 1D the velocity gradient reduces to a scalar time dependent value $\kappa(t)$. In this experiment we have taken $\kappa(t) = 0.1, 0.3$ and $\mathrm{We} = 1$ to minimise the error associated with a convection dominated problem. The time step size was chosen to be $\Delta t = 0.05$.



(a) $\kappa(t) = 0.1$

(b) $\kappa(t) = 0.3$

Figure 4.3: Normality Preservation

Figure 4.3 shows the increasing error in the normality of the numerical solution as the number of time steps is increased. In Figure 4.3(a) we notice that normality is preserved to a reasonable degree of accuracy but unfortunately not quite within computer precision as we would expect. In Figure 4.3(b) the situation is worse still where after the 40th time step we find the error in the normality jumps to 1 (beyond the scale of the graph). The reason for this jump is that the solution obtained at this time step is identically zero to computer precision. Clearly, in this case, normality is not sufficiently preserved in the PGD. However, there is a simple way of fixing the solution to ensure that it satisfies the normality condition. Indeed, it is easy to see from the Fokker-Planck equation (4.13)-(4.14) that if ψ is a solution then $c\psi$ is also a solution for all $c \in \mathbb{R}$. Therefore upon obtaining a PGD solution to the weak problem (4.18), $\hat{\psi}^{k+1}$, we can fix normality by multiplying our solution by the constant:

$$c_k = \frac{1}{\int_{\mathcal{Q}} \sqrt{M}\hat{\psi}^{k+1}\, d\mathcal{Q}} \tag{4.20}$$

for each $k = 0, \ldots, T/\Delta t - 1$. This can only be done if we have that $\int_{\mathcal{Q}} \sqrt{M}\hat{\psi}^{k+1}\, d\mathcal{Q} \neq 0$. This should only be the case when we obtain a solution $\hat{\psi}^{k+1} \equiv 0$ as we experienced in Figure 4.3(b). From this Figure we see that, even if we normalise at each step we can still suddenly obtain a zero solution in a single step. This problem seemed to occur more often the more convection dominated the problem became. In this case we can try to impose the normality implicitly. Ammar et al. [9] imposed normality by using a Lagrangian multiplier. This could be employed by augmenting the variational problem, which is the minimisation of the functional (4.19), in the following way: For each $k = 0, \ldots, T/\Delta t - 1$ find $\hat{\psi}^{k+1} \in H^1(\mathcal{Q}; M)$

and $\lambda \in \mathbb{R}$ such that:

$$(\hat{\psi}^{k+1}, \lambda) = \arg \min_{\phi \in H^1(\mathcal{Q};M)} \max_{\mu \in \mathbb{R}} \left(\mathcal{J}_k(\phi) + \mu \left(1 - \int_{\mathcal{Q}} \sqrt{M}\phi \, d\mathcal{Q} \right) \right).$$

Unfortunately this means that the problem can no longer be expressed as a minimisation problem and hence we cannot prove convergence of an associated greedy algorithm. For this reason we instead take inspiration from our previous work on least-squares PGDs and augment our variational problem in the following way: For each $k = 0, \ldots, T/\Delta t - 1$ find $\hat{\psi}^{k+1} \in H^1(\mathcal{Q}; M)$ such that:

$$\hat{\psi}^{k+1} = \arg \min_{\phi \in H^1(\mathcal{Q};M)} \left(\mathcal{J}_k(\phi) + \gamma \left| 1 - \int_{\mathcal{Q}} \sqrt{M}\phi \, d\mathcal{Q} \right|^2 \right),$$

for some adjustable constant $\gamma > 0$.

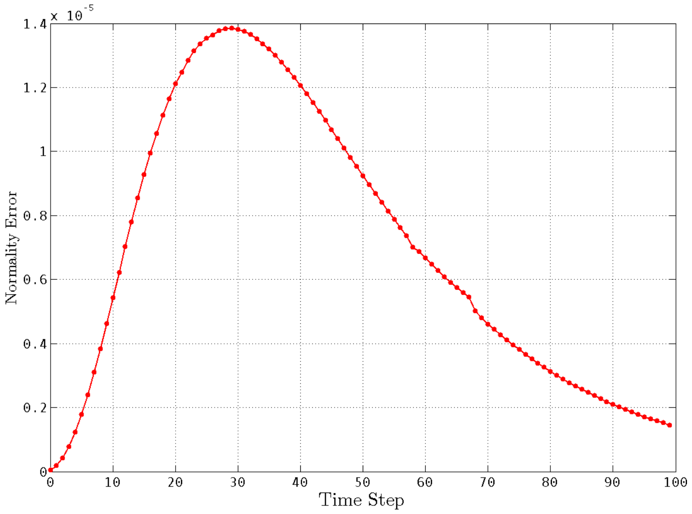


Figure 4.4: $\kappa(t) = 0.3$

In Figure 4.4 we have applied this least-squares imposition of the normality to the same problem as in Figure 4.3(b). This time we find that the normality no longer jumps to 1 as the solution does not collapse to zero. The normality is preserved to a reasonable degree but again we find that it is not quite within computer precision. This can be improved by increasing the constant γ but choosing this too large can have adverse effects on the qualitative results of the solution. Instead, we can now fix normality using the constant multiple (4.20) since we no longer obtain a solution with zero integral.

We must emphasise that this implicit least-squares imposition of the normality should only be used when we are in the situation in Figure 4.3(b) where we have a zero solution. The reason for this is that imposing the normality in this way leads to a significantly slower algorithm since the matrices in the linear systems become full. Indeed, in Figure 4.5 we have plotted the CPU times for increasing time steps, for $\kappa(t) = 0.1$ and $J = 7$, both with and without the implicit least-squares normality imposition. From this we can see that the CPU time is twice as long

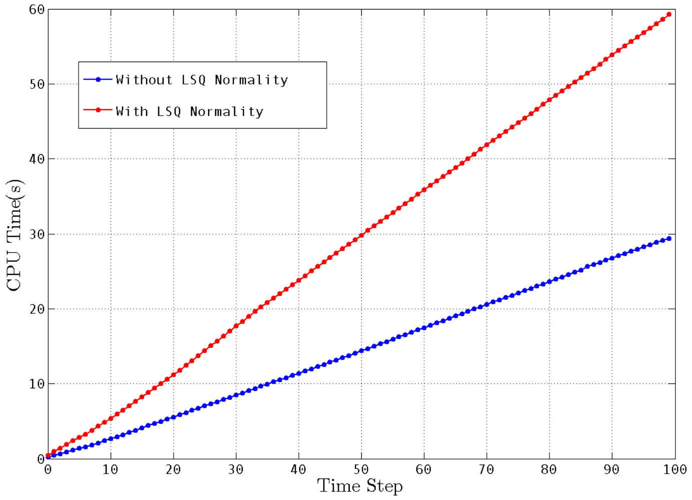
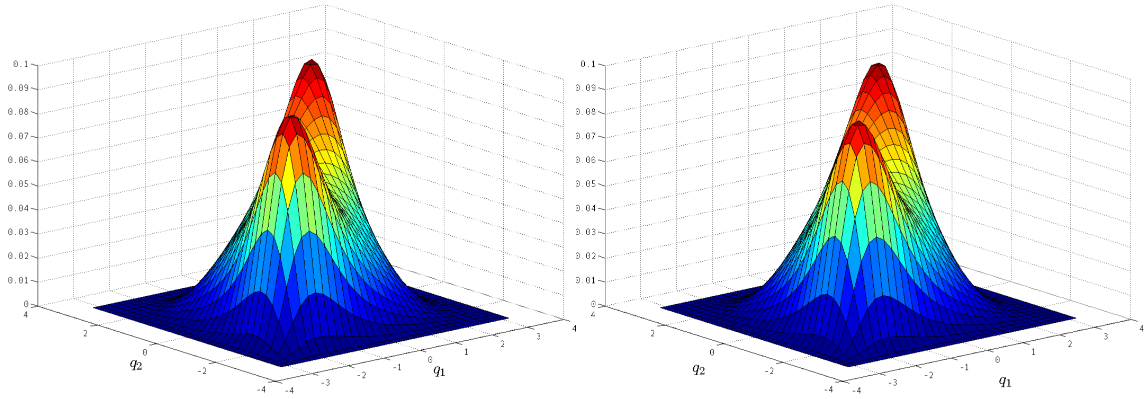


Figure 4.5: Comparison of CPU Times

when imposing the normality implicitly in this way.

So far we have only considered the preservation of normality in our PGD algorithms and we still need to verify the solutions that we obtain. To do this we compare our results to those of Ammar et al. [8] for the problem based on the same bead-spring configuration in Figure 4.2. Their results were themselves verified against a standard finite element mesh-based approach to the same problem. The authors considered the standard definition of the steady state Fokker-Planck equation, without the Maxwellian weighting, with maximum extensibility $\sqrt{b_1} = \sqrt{b_2} = \sqrt{10}$, velocity gradient $\kappa = \sqrt{2}/4$, and Weissenberg number We = 1.



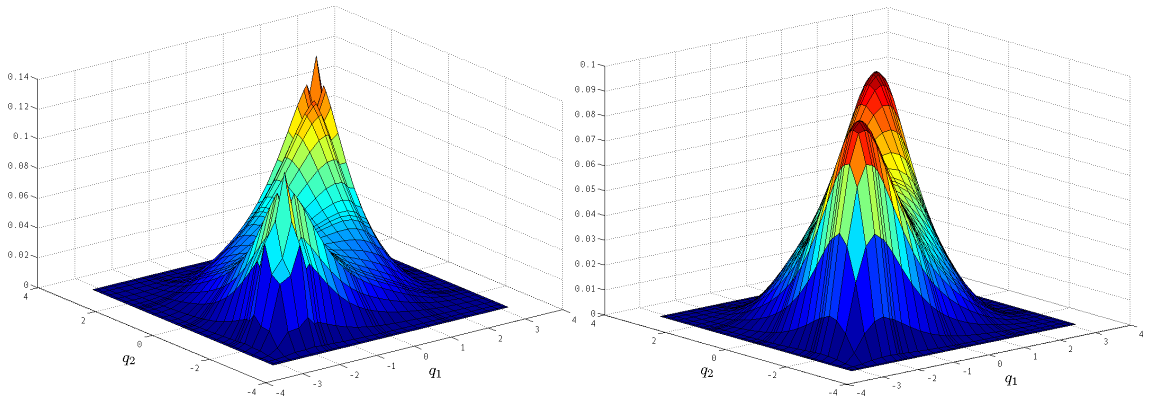
(a) Without Least-Squares Normality

(b) With Least-Squares Normality

Figure 4.6: PGD Approximation of ψ

In Figure 4.6 we have plotted the PGD approximations obtained for this problem after $T = 200$ time steps, with a step size of $\Delta t = 0.05$, using a discretisation with linear basis functions ($N = 1$) over $K_x = K_y = 30$ elements in each scalar spring direction. We have plotted the results both with the least-squares imposition of the normality and without and they are both in excellent agreement with the results of Ammar et al. [8]. The rank of the PGD approximation used in Figures 6(a) and 6(b) was $J = 10$ and $J = 15$, respectively. Additional PGD basis functions were required when using the least-squares imposition of the normality in order to obtain a solution of the same quality as in Figure 4.6(a) which highlights another disadvantage of using this approach. In the results of Ammar et al. they used a rank $J = 7$ PGD approximation to obtain a solution of the same quality. This is most likely because they considered a PGD algorithm with a projective step, the likes of which can be associated with an orthogonal greedy algorithm, and it has been observed that this can provide better rates of convergence in the rank.

Note that in Figure 4.6 we used linear elements. Figure 4.7(a) shows the result when we used higher order spectral elements, in this case $N = 8$ on $K_x = K_y = 4$ elements in each scalar spring direction. From this plot we can see that the solution obtained using the higher order elements appears to be very poor. The reason for this is that since we are using fewer elements and the problem is slightly convection dominated then spurious oscillations are introduced.



(a) Without SU

(b) With SU

Figure 4.7: PGD Approximation of ψ

In order to stabilise our solution we employed streamline upwinding (SU) and the result of this can be seen in Figure 4.7(b) which now agrees excellently with the results of Ammar et al. [8]. Streamline upwinding (see Brooks and Hughes [33]) is essentially a method which introduces an additional artificial diffusion term in order to balance out the convection. For the Maxwellian transformed Fokker-Planck equation (4.13)-(4.14) this artificial diffusion term is given by:

$$c(\hat{\psi}^{k+1}, \hat{\psi}^*; \underline{\boldsymbol{\kappa}}^{k+1}) = \sum_{i,j=1}^{d} \int_{\mathcal{Q}} \frac{\tau}{M} \nabla_{\mathbf{q}_i} \cdot (\underline{\boldsymbol{\kappa}}^{k+1} \cdot \mathbf{q}_i \sqrt{M} \hat{\psi}^{k+1}) \nabla_{\mathbf{q}_j} \cdot (\underline{\boldsymbol{\kappa}}^{k+1} \cdot \mathbf{q}_j \sqrt{M} \hat{\psi}^*) \, d\mathcal{Q},$$

where τ is the stabilisation parameter. While this may not seem very rigorous this additional term is derived by considering the adjusted test function:

$$\tilde{\psi}^* := \frac{1}{M}\left(\psi^* + \tau \sum_{j=1}^{d} \nabla_{\mathbf{q}_j} \cdot (\underline{\boldsymbol{\kappa}}^{k+1} \cdot \mathbf{q}_j \psi^*) \right),$$

instead of the usual test function ψ^*/M where $\psi^*/\sqrt{M} \in H^1(\mathcal{Q}; M)$. This adjusted test function is then only applied to the convective term in the Fokker-Planck equation which, upon the change of variables, yields the artificial diffusion term, $c(\hat{\psi}^{k+1}, \hat{\psi}^*; \underline{\boldsymbol{\kappa}}^{k+1})$, as given above. Of course a more consistent stabilisation method would be to apply this adjusted test function to the whole equation not just the convective term. This, however, leads to second order derivatives appearing in the weak formulation of the problem which means we would require a conforming discretisation space that is C^1-continuous which is not the case for standard finite/spectral elements over element edges. The streamline upwind Petrov-Galerkin (SUPG) method [33] addresses this issue by only calculating the additional stabilisation terms on element interiors. The SUPG has been applied to the PGD by González et al. [72]. However, for our means, streamline upwinding appears to be sufficient to obtain accurate results as can be seen from Figure 4.7(b).

As for the convergence of a PGD algorithm for the streamline upwinded problem, we can associate this with a greedy algorithm based on the minimisation of the adjusted functionals:

$$\tilde{\mathcal{J}}_k(\hat{\psi}^{k+1}) := \mathcal{J}_k(\hat{\psi}^{k+1}) + \frac{1}{2}c(\hat{\psi}^{k+1}, \hat{\psi}^{k+1}; \underline{\boldsymbol{\kappa}}^{k+1}),$$

for $k = 0, \ldots, T/\Delta t - 1$. While convergence of this greedy algorithm can be proven, to prove that it converges to the solution of the Fokker-Planck equation is a more delicate matter related to the consistency of the stabilisation.

Finally, an important part of streamline upwinding is the choice of the stabilisation parameter τ. For linear elements there is a known choice of τ which yields nodally accurate results [33]. Unfortunately, for higher order methods, such as the spectral methods we have considered, no such choice of τ is known. In his thesis, Chauvière [43] considered stabilised methods for spectral elements and the stabilisation parameter in this case was chosen to be $\tau = 1/N^2$ which we have also employed. One could employ more specialised methods of stabilisation such as the locally-upwinded spectral technique (LUST) developed by Owens et al. [108] for the stabilisation of spectral element methods but we do not want to dwell too much on the choice of discretisation and rather we are more interested in how the PGD performs as an algorithm in itself. We do, however, note that the convergence of the streamline upwinded problem appears to be slower in the rank since a rank $J = 15$ approximation was required to obtain the level of accuracy displayed in Figure 4.7(b). For this reason we believe that linear finite elements are probably more well suited to this problem and we will continue to use these from here on.

4.2.5 Concluding Remarks

In this section we have considered a PGD approximation of the Maxwellian tranformed Fokker-Planck equation defined in configuration space. We have shown that normality is not preserved in the PGD but have presented alternative methods of ensuring normality. We did, however, note that non-negativity of the solution was consistently preserved in our results. Results have also been verified against previous work on the solution of the Fokker-Planck equation and stabilisation via streamline upwinding has been employed where necessary. The ultimate goal is to couple the Fokker-Planck equation with a macroscopic continuum model. However, as mentioned previously, the Fokker-Planck equation defined in configuration space is not suitable for this coupling since it assumes that the flow is spatially homogeneous i.e. the velocity gradient $\underline{\boldsymbol{\kappa}}$ does not depend on the physical coordinate $\mathbf{x}$. In the following section we present the fully non-homogeneous Fokker-Planck equation which does take into account the movement of the centres of mass of the bead-spring chains within physical space.

4.3 The Full Fokker-Planck Equation

The fully non-homogeneous Fokker-Planck equation for the bead-spring configuration given in Figure 4.1 is given by (see e.g. Figueroa and Süli [69]):

$$\begin{aligned}\frac{\partial \psi}{\partial t} = &-\sum_{i=1}^{d} \nabla_{\mathbf{q}_i} \cdot \left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}_i \psi - \frac{1}{4\,\mathrm{We}} \sum_{j=1}^{d} A_{i,j} \left(M \nabla_{\mathbf{q}_j} \left(\frac{\psi}{M} \right) \right) \right) \\ &+ \frac{(l_0/L_0)^2}{4\,\mathrm{We}(d+1)} \Delta_{\mathbf{x}} \psi - \mathbf{u} \cdot \nabla_{\mathbf{x}} \psi \quad \text{in} \quad \Omega \times \mathcal{Q} \times (0,T], \end{aligned} \tag{4.21}$$

$$\left(\frac{(l_0/L_0)^2}{4\,\mathrm{We}(d+1)} \nabla_{\mathbf{x}} \psi - \psi \mathbf{u} \right) \cdot \mathbf{n}_{\partial\Omega} = 0 \quad \text{on} \quad \partial\Omega \times \mathcal{Q} \times (0,T], \tag{4.22}$$

$$\left(\underline{\boldsymbol{\kappa}} \cdot \mathbf{q}_i \psi - \frac{1}{4\,\mathrm{We}} \sum_{j=1}^{d} A_{i,j} \left(M \nabla_{\mathbf{q}_j} \left(\frac{\psi}{M} \right) \right) \right) \cdot \mathbf{n}_{\partial Q_i} = 0 \quad \text{on} \quad \Omega \times \partial_i \mathcal{Q} \times (0,T], \tag{4.23}$$

for $i = 1, \ldots, d$, where $\mathbf{x} \in \Omega$ denotes the physical coordinate, $\mathbf{u}(\mathbf{x}, t)$ denotes the macroscopic velocity of the flow and l_0 and L_0 denote the characteristic length-scale of the spring and macroscopic length, respectively. In this version of the Fokker-Planck equation the pdf, $\psi = \psi(\mathbf{x}, \mathbf{q}_1, \ldots, \mathbf{q}_d, t)$, depends not only on time and spring configuration but also on physical space. Furthermore, the velocity gradient, $\underline{\boldsymbol{\kappa}}(\mathbf{x}, t) = \nabla_{\mathbf{x}} \mathbf{u}(\mathbf{x}, t)$, now also depends on physical space as well as time. We have also assumed that the flow is incompressible so that $\nabla_{\mathbf{x}} \cdot \mathbf{u} = 0$ and hence $\mathrm{Tr}(\underline{\boldsymbol{\kappa}}) = 0$. The above problem also includes the initial condition $\psi^0(\mathbf{x}, \mathbf{q}_1, \ldots, \mathbf{q}_d) = \psi(\mathbf{x}, \mathbf{q}_1, \ldots, \mathbf{q}_d, 0)$.

The full Fokker-Planck equation (4.21) can be derived as the forward Kolmogorov equation [87] (a deterministic parabolic PDE which describes the evolution of a pdf for a particular stochastic process) when the stochastic process relates to the random movement of the springs in the bead-spring chain pictured in Figure 4.1 as well as the random movement of their centres of mass in physical space. A detailed explanation of this derivation can be found in the paper of Barrett and Süli [16]. A stochastic approach equivalent to a generalisation of the full Fokker-Planck equation (4.21) based on a modified Brownian configuration field was also provided by Schieber [118]. Note that the Fokker-Planck equation in configuration space can also be derived analogously (see e.g. Knezevic [82]).

Since ψ is a pdf we once again require non-negativity:

$$\psi(\mathbf{x}, \mathbf{q}_1, \ldots, \mathbf{q}_d, t) \geq 0, \quad \forall(\mathbf{x}, \mathbf{q}_1, \ldots, \mathbf{q}_d, t) \in \Omega \times \mathcal{Q} \times [0, T],$$

as well as the normality condition:

$$\int_{\Omega \times \mathcal{Q}} \psi(\mathbf{x}, \mathbf{q}_1, \ldots, \mathbf{q}_d, t) d\Omega d\mathcal{Q} = 1, \quad \forall t \in [0, T].$$

As was the case for the Fokker-Planck equation in configuration space these properties of the pdf are imposed by selecting an initial condition that satisfies both the properties and by the choice of the boundary conditions. Indeed, integrating (4.21) over $\Omega \times \mathcal{Q}$ and employing the divergence theorem yields:

$$\frac{\partial}{\partial t} \int_{\Omega \times \mathcal{Q}} \psi \ d\Omega d\mathcal{Q} = \int_{\mathcal{Q}} \int_{\partial\Omega} \left(\frac{(l_0/L_0)^2}{4\,\mathrm{We}(d+1)} \nabla_{\mathbf{x}} \psi - \psi \mathbf{u} \right) \cdot \mathbf{n}_{\partial\Omega} \ d\partial\Omega d\mathcal{Q}. \tag{4.24}$$

where the terms involving derivatives in configuration space have been left out since from the previous section we know they do not contribute to the right hand side. Note that the divergence theorem is applied to the convective term, $\mathbf{u} \cdot \nabla_{\mathbf{x}} \psi$, by noticing that for incompressible flow we have that $\mathbf{u} \cdot \nabla_{\mathbf{x}} \psi = \nabla_{\mathbf{x}} \cdot (\psi \mathbf{u})$. Using the boundary condition (4.22) on $\partial\Omega$ we obtain:

$$\frac{\partial}{\partial t} \int_{\Omega \times \mathcal{Q}} \psi \ d\Omega d\mathcal{Q} = 0,$$

hence, with a choice of initial condition that satisfies normality, we obtain:

$$\int_{\Omega \times \mathcal{Q}} \psi \ d\Omega d\mathcal{Q} = \int_{\Omega \times \mathcal{Q}} \psi^0 \ d\Omega d\mathcal{Q} = 1, \quad \forall t \in [0, T].$$

Note that, for simplicity, we will assume that we have an enclosed flow so that the macroscopic boundary condition is given by $\mathbf{u} \cdot \mathbf{n}_{\partial\Omega} = 0$. This reduces the boundary condition on physical space (4.22) in our Fokker-Planck model to $\nabla_{\mathbf{x}} \psi \cdot \mathbf{n}_{\partial\Omega} = 0$.

In almost all practical applications one would have that $L_0 >> l_0$ hence the contribution from the centre of mass diffusion term, $\frac{(l_0/L_0)^2}{4\,\mathrm{We}(d+1)}\Delta_{\mathbf{x}}\psi$, is very small. Indeed, Bhave et al. [19] estimated that, for a macroscopic length scale of 1cm, the size of this diffusion coefficient would be in the range of $10^{-7} - 10^{-9}$. In fact in many applications of the fully non-homogeneous Fokker-Planck equation this term is left out for this reason (see e.g. Chauvière and Lozinski [44]). This simplifying assumption was rigorously justified by Barrett and Süli [14]. The centre of mass diffusion term was first included in the full Fokker-Planck equation by El-Kareh and Leal [62]. In this paper the authors hoped that the additional diffusion would help to stabilise the solution. In a similar vein we include the centre of mass diffusion for the time being since in the operator splitting scheme it means we need to solve an, albeit highly convection dominated, convection-diffusion equation rather than a purely hyperbolic problem. This point will become clearer in the upcoming section on the operator splitting scheme. The full Fokker-Planck equation including centre of mass diffusion was also considered by Lozinski et al. [99] for the reason that this term is not always negligible, for example in flow through a very narrow tube. Existence of a solution has also been proven for the system (4.21)-(4.23) including the centre of mass diffusion and coupled with the Navier-Stokes equations by Barrett and Süli [14–16].

4.3.1 Weak Formulation

Before deriving the weak formulation of the full Fokker-Planck equation we introduce the following relevant space:

$$\mathcal{X} := \Big\{ \varphi \in L^2(\Omega \times \mathcal{Q}) \ : \ (\phi(\mathbf{x})) \in L^2(\Omega; H^1(\mathcal{Q}; M)) \cap H^1(\Omega; L^2(\mathcal{Q})) \Big\},$$

where $(\phi(\mathbf{x}))(\mathbf{q}) := \varphi(\mathbf{x}, \mathbf{q})$ are a family of functions in configuration space that are parametrised in physical space defined in the Bochner space given above.

To derive the weak formulation of the full Fokker-Planck equation we multiply (4.21) by a test function ψ^*/M in a suitable function space and then integrate over $\Omega \times \mathcal{Q}$. As with the Fokker-Planck equation in configuration space we then employ the change of variables $\hat{\psi} = \psi/\sqrt{M}$ yielding the following weak formulation: Find $\hat{\psi} \in L^2([0,T]; \mathcal{X})$ such that:

$$\frac{\partial}{\partial t}\langle \hat{\psi}, \hat{\psi}^* \rangle + a(\hat{\psi}, \hat{\psi}^*) - b(\hat{\psi}, \hat{\psi}^*; \underline{\boldsymbol{\kappa}}) + \tilde{a}(\hat{\psi}, \hat{\psi}^*) + \tilde{b}(\hat{\psi}, \hat{\psi}^*; \mathbf{u}) = 0, \tag{4.25}$$

for all $\hat{\psi}^* \in \mathcal{X}$, where $a(\cdot, \cdot)$ and $b(\cdot, \cdot; \underline{\boldsymbol{\kappa}})$ are the bilinear forms given by (4.16)-(4.17) except integrated over $\Omega \times \mathcal{Q}$ (rather than just over configuration space) and where:

$$\tilde{a}(\psi, \psi^*) := \frac{(l_0/L_0)^2}{4\,\mathrm{We}(d+1)} \int_{\Omega \times \mathcal{Q}} \nabla_{\mathbf{x}} \psi \cdot \nabla_{\mathbf{x}} \psi^* \; d\Omega d\mathcal{Q}, \tag{4.26}$$

$$\tilde{b}(\psi, \psi^*; \mathbf{u}) := \int_{\Omega \times \mathcal{Q}} (\mathbf{u} \cdot \nabla_{\mathbf{x}} \psi) \psi^* \; d\Omega d\mathcal{Q}. \tag{4.27}$$

Note that, unlike in the weak formulation of the Fokker-Planck equation in configuration space where we considered a semi-discrete problem, we have not yet employed a time integration scheme. The reason for this is that the time integration scheme is part of the operator splitting scheme that we will describe in detail the next section. Note that it will become apparent that this operator splitting scheme is not suitable to be used when employing a PGD approximation in the high-dimensional configuration space. It is the aim of the next section to rigorously explain why this is the case but the method will not be employed.

4.3.2 Operator Splitting Scheme

Operator splitting schemes were originally constructed when computer memory was not as abundant as it is today and even solving problems in 2D was a computational challenge (e.g. see Douglas and Dupont [59]). Operator splitting schemes are also referred to as alternating direction schemes. However, we prefer to use the former as it avoids any confusion with the alternating directions fixed point algorithm that is used in the linearisation of the PGD. The basic idea of operator splitting schemes is to separate a time dependent problem into several lower dimensional time dependent problems where, in each of the low dimensional problems, derivatives in only one variable appear. More formally, consider the following problem in 2D:

$$\frac{\partial u}{\partial t} + (L_x + L_y)u = 0, \quad \text{in} \quad \Omega = \Omega_x \times \Omega_y, \tag{4.28}$$

where $u = u(x, y, t)$, and L_x and L_y are differential operators in x and y respectively. There are a number of ways of applying operator splitting but the one we are concerned with is fractional time stepping. This yields the following operator splitting for the problem (4.28):

$$\frac{u^{k+1/2} - u^k}{\Delta t} + L_x u^{k+1/2} = 0, \tag{4.29}$$

$$\frac{u^{k+1} - u^{k+1/2}}{\Delta t} + L_y u^{k+1} = 0, \tag{4.30}$$

where $u^k = u(x, y, t^k)$ denotes the solution obtained at the k^{th} time step such that $t^k = k\Delta t$ for $k = 0, \ldots, T/\Delta t$ and similarly for the fractional time steps, $u^{k+1/2} = u(x, y, t^{k+1/2})$ where $t^{k+1/2} = (k + 1/2)\Delta t$ for $k = 0, \ldots, T/\Delta t - 1$.

It is also possible to apply a weak formulation to the operator splitting steps (4.29)-

(4.30). To demonstrate this we shall consider the simple example of the heat equation in 2D whereby $L_x = \frac{\partial^2}{\partial x^2}$ and $L_y = \frac{\partial^2}{\partial y^2}$. We begin by considering the x-direction for $k = 0$ (4.29) by multiplying by a test function, $u^*(x, y)$, and integrating over Ω which upon applying the divergence theorem (assuming we have homogeneous Dirichlet boundary conditions on $\partial\Omega$) yields:

$$\frac{1}{\Delta t}\int_{\Omega} u^{1/2}u^* \, d\Omega - \int_{\Omega} \frac{\partial u^{1/2}}{\partial x}\frac{\partial u^*}{\partial x} \, d\Omega = \frac{1}{\Delta t}\int_{\Omega} u^0 u^* \, d\Omega,$$

where $u^0(x, y)$ is some given initial condition. We then select test functions, $u^*(x, y) = u^*_x(x)h_i(y)$, where $h_i(y)$, $i = 1, \ldots, N_y$, are some basis functions such that $h_i(y_j) = \delta_{i,j}$ for quadrature points y_j, $j = 1, \ldots, N_y$. This enables us to apply numerical integration in the y variable yielding:

$$\sum_{j=1}^{N_y} w_j \delta_{i,j} \left(\frac{1}{\Delta t}\int_{\Omega_x} u_j^{1/2}(x)u^*_x(x) \, dx - \int_{\Omega_x} \frac{du_j^{1/2}(x)}{dx}\frac{du^*_x(x)}{dx} \, dx \right)$$
$$\approx \sum_{j=1}^{N_y} w_j \delta_{i,j} \frac{1}{\Delta t}\int_{\Omega_x} u^0(x, y_j)u^*_x(x) \, dx,$$

for $i = 1, \ldots, N_y$, where $u_j^{1/2}(x) := u^{1/2}(x, y_j)$, $j = 1, \ldots, N_y$. Equivalently, we have:

$$\frac{1}{\Delta t}\int_{\Omega_x} u_i^{1/2}(x)u^*_x(x) \, dx - \int_{\Omega_x} \frac{du_i^{1/2}(x)}{dx}\frac{du^*_x(x)}{dx} \, dx \approx \frac{1}{\Delta t}\int_{\Omega_x} u^0(x, y_i)u^*_x(x) \, dx,$$

for $i = 1, \ldots, N_y$. Note that here '≈' denotes equality up to quadrature error. This step of the operator splitting scheme then amounts to solving N_y 1-dimensional problems in x with solutions $u_i^{1/2}(x)$, $i = 1, \ldots, N_y$. Note that these N_y problems are all completely uncoupled which lends themselves perfectly to parallel computing. While this would not be important for this simple example it would yield a significant improvement in computational time for the full Fokker-Planck equation. Indeed, the parallel implementation of this was considered in detail by Knezevic and Süli [83].

Returning our attention to the example, we now need to solve in the y-direction. This is done analogously to the previous step by multiplying (4.30) by test functions $u^*(x, y) = h_i(x)u^*_y(y)$, $i = 1, \ldots, N_x$, where $h_i(x_j) = \delta_{i,j}$ for quadrature points x_j, $j = 1, \ldots, N_x$. We then integrate over Ω and apply numerical integration in the x variable. This yields the following:

$$\frac{1}{\Delta t}\int_{\Omega_y} u_i^1(y)u^*_y(y) \, dy - \int_{\Omega_y} \frac{du_i^1(y)}{dy}\frac{du^*_y(y)}{dy} \, dy \approx \frac{1}{\Delta t}\int_{\Omega_y} u^{1/2}(x_i, y)u^*_y(y) \, dy,$$

for $i = 1, \ldots, N_x$, where $u_i^1(y) := u^1(x_i, y)$. The issue is now that the function $u^{1/2}(x_i, y)$ is not known for all $y \in \Omega_y$. However, upon discretisation in y we can write:

$$u^{1/2}(x_i, y) \approx \sum_{j=1}^{N_y} u^{1/2}(x_i, y_j)h_j(y) = \sum_{j=1}^{N_y} u_j^{1/2}(x_i)h_j(y),$$

where $u_j^{1/2}(x)$, $j = 1, \dots, N_y$ were the functions evaluated from the previous step of the operator splitting scheme. Therefore the second step of the operator splitting scheme amounts to solving N_x 1-dimensional problems in y with solutions $u_i^1(y)$, $i = 1, \dots, N_x$. As with the previous step these problems are completely decoupled from one another. The scheme then continues in this way for $k = 1, \dots, T/\Delta t - 1$.

It is also possible to derive an operator splitting scheme by starting with a weak formulation of the problem which was first considered by Douglas and Dupont [59]. This technique was also employed by Knezevic and Süli [83] for the full Fokker-Planck equation when the configurational convective term, $b(\hat{\psi}, \hat{\psi}^*; \underline{\boldsymbol{\kappa}})$, in (4.25) is solved explicitly in time. This enabled the authors to provide convergence estimates for an operator splitting scheme for this problem and prove stability estimates for the scheme for the full Fokker-Planck equation with the configurational convective term treated both explicitly and implicitly. The aforementioned paper was the first in depth numerical analysis of the operator splitting scheme applied to the full Fokker-Planck equation but it had previously been applied to this problem by Chauvière and Lozinski [44].

The main aim of applying this operator splitting scheme to the full Fokker-Planck equation is to decouple the physical space problem from the configuration space problem. The reason we are able to do this is that the terms involving derivatives in physical space are separate from those involving configurational derivatives. Indeed, the reason we do not apply an operator splitting technique to separate all the spring vectors, for example, is that there are mixed derivatives involving these terms in the Fokker-Planck equation (i.e. the terms associated with the off diagonal entries of the Rouse matrix $A_{i,j}$, $i, j = 1, \dots, d$). So, while this operator splitting scheme can alleviate a certain degree of the curse of dimensionality, we still need to make use of the PGD for the potentially high-dimensional problem in configuration space. Unfortunately, the operator splitting scheme does not appear to be applicable when employing a PGD in part of the operator splitting. We believe this problem may have been overlooked when such a method was proposed in the theoretical paper of Figueroa and Süli [69] (although the proof of convergence of the configurational Fokker-Planck equation in said paper is still perfectly valid). To demonstrate this issue we proceed with an operator splitting for the full Fokker-Planck equation. Note that the scheme we present here differs slightly from the one proposed by Figueroa and Süli [69] but the underlying concept is the same.

The fractional time stepping operator splitting scheme we consider for the full

Fokker-Planck equation (4.25) is as follows: For $k = 0, \dots, T/\Delta t - 1$ find $\hat{\psi}^{k+1} \in \mathcal{X}$ such that:

$$\frac{1}{\Delta t}\langle \hat{\psi}^{k+1/2}, \hat{\psi}^* \rangle + \tilde{a}(\hat{\psi}^{k+1/2}, \hat{\psi}^*) + \tilde{b}(\hat{\psi}^{k+1/2}, \hat{\psi}^*; \mathbf{u}^{k+1/2}) = \frac{1}{\Delta t}\langle \hat{\psi}^{k}, \hat{\psi}^* \rangle, \tag{4.31}$$

$$\frac{1}{\Delta t}\langle \hat{\psi}^{k+1}, \hat{\psi}^* \rangle + a(\hat{\psi}^{k+1}, \hat{\psi}^*) = b(\hat{\psi}^{k+1/2}, \hat{\psi}^*; \underline{\boldsymbol{\kappa}}^{k+1/2}) + \frac{1}{\Delta t}\langle \hat{\psi}^{k+1/2}, \hat{\psi}^* \rangle, \tag{4.32}$$

for all $\hat{\psi}^* \in \mathcal{X}$, where $\mathbf{u}^{k+1/2}(\mathbf{x}) = \mathbf{u}(\mathbf{x}, t^{k+1/2})$ and $\underline{\boldsymbol{\kappa}}^{k+1/2}(\mathbf{x}) = \underline{\boldsymbol{\kappa}}(\mathbf{x}, t^{k+1/2})$.

This operator splitting is then solved analogously to the heat equation we presented earlier. Indeed, for the $\mathbf{x}$-direction, (4.31), we make the choice of test functions $\hat{\psi}^* = \hat{\psi}^*_{\mathbf{x}}(\mathbf{x}) h_{i_1}(\mathbf{q}_1) \times \cdots \times h_{i_d}(\mathbf{q}_d)$ where $h_{i_n}(\mathbf{q}_n^{(j_n)}) = \delta_{i_n, j_n}$ $(i_n = 1, \dots, N_{\mathbf{q}_n})$ for quadrature points $\mathbf{q}_n^{(j_n)}$ $(j_n = 1, \dots, N_{\mathbf{q}_n})$ for all $n = 1, \dots, d$. We then apply numerical quadrature over configuration space, $\mathcal{Q}$, yielding:

$$\frac{1}{\Delta t}\int_\Omega \hat{\psi}^{k+1/2}_{i_1,\dots,i_d}(\mathbf{x})\hat{\psi}^*_{\mathbf{x}}(\mathbf{x})\; d\Omega + \frac{(l_0/L_0)^2}{4\,\mathrm{We}(d+1)}\int_\Omega \nabla_{\mathbf{x}}\hat{\psi}^{k+1/2}_{i_1,\dots,i_d}(\mathbf{x}) \cdot \nabla_{\mathbf{x}}\hat{\psi}^*_{\mathbf{x}}(\mathbf{x})\; d\Omega$$
$$+ \int_\Omega \big(\mathbf{u}^{k+1/2}(\mathbf{x}) \cdot \nabla_{\mathbf{x}}\hat{\psi}^{k+1/2}_{i_1,\dots,i_d}(\mathbf{x})\big)\hat{\psi}^*_{\mathbf{x}}(\mathbf{x})\; d\Omega \approx \frac{1}{\Delta t}\int_\Omega \hat{\psi}^{k}_{i_1,\dots,i_d}(\mathbf{x})\hat{\psi}^*_{\mathbf{x}}(\mathbf{x})\; d\Omega, \tag{4.33}$$

where $\hat{\psi}^{k+1/2}_{i_1,\dots,i_d}(\mathbf{x}) := \hat{\psi}^{k+1/2}(\mathbf{x}, \mathbf{q}_1^{(i_1)}, \dots, \mathbf{q}_d^{(i_d)})$ (and similarly for $\hat{\psi}^{k}_{i_1,\dots,i_d}(\mathbf{x})$) for $k = 0, \dots, T/\Delta t - 1$. This then amounts to solving $N_{\mathbf{q}} = \prod_{n=1}^d N_{\mathbf{q}_n}$ convection-diffusion equations. However, $N_{\mathbf{q}}$ grows exponentially as the dimension, d, of the problem increases. This means that implementing the operator splitting scheme in this way does not alleviate the curse of dimensionality for problems with high-dimensional configuration spaces at all. Furthermore, these convection-diffusion equations yield $\hat{\psi}^{k+1/2}$ evaluated at the $N_{\mathbf{q}}$ quadrature points in configuration space whereas we require the coefficients of the PGD modes for a PGD approximation of $\hat{\psi}^{k+1/2}$ in configuration space. To this end one could consider the following separated representation in configuration space:

$$\hat{\psi}^{k+1/2}(\mathbf{x}, \mathbf{q}_1, \dots, \mathbf{q}_d) \approx \sum_{j=1}^{J}\prod_{i=1}^{d} Q_{n,j}(\mathbf{x}, \mathbf{q}_n).$$

Then we have that:

$$\hat{\psi}^{k+1/2}_{i_1,\dots,i_d}(\mathbf{x}) \approx \sum_{j=1}^{J}\prod_{n=1}^{d} Q_{n,j}(\mathbf{x}, \mathbf{q}_n^{(i_n)}),$$

where

$$Q_{n,j}(\mathbf{x}, \mathbf{q}_n^{(i_n)}) = \sum_{j_n=1}^{N_{\mathbf{q}_n}} \alpha_{j,j_n}(\mathbf{x}) h_{j_n}(\mathbf{q}_n^{(i_n)}) = \alpha_{j,i_n}(\mathbf{x}),$$

where α_{j,i_n}, $j = 1, \dots, J$, $i_n = 1, \dots, N_{\mathbf{q}_n}$, $n = 1, \dots, d$, are the coefficients of the PGD modes which would need to be found in order to evaluate the right hand side of the **q**-direction step (4.32) with the PGD. We then have that:

$$\hat{\psi}^{k+1/2}_{i_1,\ldots,i_d}(\mathbf{x}) \approx \sum_{j=1}^{J} \prod_{n=1}^{d} \alpha_{j,i_n}(\mathbf{x}). \tag{4.34}$$

However, if we insert (4.34) into (4.33) and individually seek solutions, $\alpha_{j,i_n}(\mathbf{x})$, then we are left with an ill-posed problem. Indeed, it is easy to see that we do not have uniqueness since any solution $\alpha_{j,i_n}(\mathbf{x})$ could arbitrarily be swapped with $\alpha_{j,i_n^*}(\mathbf{x})$ for $i_n \neq i_n^*$.

To summarise: Operator splitting schemes are perfectly suited for solving the full Fokker-Planck equation when the configuration space is of moderate dimension (e.g. [44,83]). However, when the configuration space is sufficiently high-dimensional that it warrants the application of the PGD to alleviate the curse of dimensionality, then it becomes unclear, if not impossible, to apply such an operator splitting scheme. Essentially the reason for this is that, upon the application of numerical integration in configuration space, the information about which PGD mode coefficient is associated to which spring vector direction is lost.

4.3.3 Implementation of the PGD

Since it is unclear how to apply an operator splitting scheme together with a PGD in configuration space then we need an alternative method to alleviate the curse of dimensionality for this problem. The alternative method we propose here is to include the physical variable in the PGD. Indeed, since we are already using a PGD in configuration space it is a simple extension to include physical space as well. Hence we begin by once again employing a backward Euler scheme. Furthermore, we shall now adopt the common simplifying assumption of removing the centre-of-mass diffusion term. Note that we had originally included this term since it meant we had to solve an elliptic convection-diffusion equation instead of a hyperbolic transport equation in the physical direction step of the operator splitting scheme. Since we are no longer using this scheme we do not gain any significant benefit from including this term. We now present two schemes (**I** & **II**) with semidiscrete weak formulations: For $k = 0, \ldots, T/\Delta t - 1$ find $\hat{\psi}^{k+1} \in \mathcal{X}$ such that:

$$\textbf{(I)} \quad \begin{aligned} &\frac{1}{\Delta t}\langle \hat{\psi}^{k+1}, \hat{\psi}^* \rangle + a(\hat{\psi}^{k+1}, \hat{\psi}^*) - b(\hat{\psi}^{k+1}, \hat{\psi}^*; \underline{\boldsymbol{\kappa}}^{k+1}) \\ &\qquad\qquad\qquad + \tilde{b}(\hat{\psi}^{k+1}, \hat{\psi}^*; \mathbf{u}^{k+1}) = \frac{1}{\Delta t}\langle \hat{\psi}^{k}, \hat{\psi}^* \rangle, \end{aligned}$$

$$\textbf{(II)} \quad \frac{1}{\Delta t}\langle \hat{\psi}^{k+1}, \hat{\psi}^* \rangle + a(\hat{\psi}^{k+1}, \hat{\psi}^*) = b(\hat{\psi}^{k}, \hat{\psi}^*; \underline{\boldsymbol{\kappa}}^{k}) - \tilde{b}(\hat{\psi}^{k}, \hat{\psi}^*; \mathbf{u}^{k}) + \frac{1}{\Delta t}\langle \hat{\psi}^{k}, \hat{\psi}^* \rangle,$$

for all $\hat{\psi}^* \in \mathcal{X}$.

Scheme **(I)** is a fully implicit scheme whereas **(II)** is semi-implicit with the two convective terms treated explicitly in time. We have considered these two schemes since for **(II)** we are solving a series of elliptic self-adjoint problems and hence it should be possible to prove convergence of an associated PGD provided that $\mathscr{X}$ satisfies the required assumptions given by Cancès et al. [37]. However, it is not clear how stable this scheme will be. Therefore we also consider a the fully implicit scheme **(I)** which should be considerably more stable than the semi-implicit scheme but on the other hand does not lead to a self-adjoint problem and so convergence of a PGD for this scheme cannot be proven.

In both schemes we seek an approximate separated representation of the solution at each timestep of the form:

$$\hat{\psi}^k(\mathbf{x}, \mathbf{q}_1, \ldots, \mathbf{q}_d) \approx \sum_{j=1}^{J} X_j(\mathbf{x}) \prod_{i=1}^{d} Q_{i,j}(\mathbf{q}_i).$$

It could also be possible to separate the individual components of $\mathbf{x}$ depending on the geometry of Ω.

4.4 Conclusions and Further Work

In this chapter we have have investigated the application of a Galerkin progressive PGD algorithm to the Maxwellian transformed Fokker-Planck equation. Following Figueroa and Süli [69], this allowed us to prove convergence of the PGD algorithm when a semi-implicit time integration scheme was used, where the convective term was solved explicitly in time. Numerical results for the Fokker-Planck equation defined purely in configuration space were supplied and we demonstrated that within the alternating direction linearisation step of the PGD algorithm it was not possible to guarantee preservation of the normality condition required for probability distribution functions. We provided a simple solution to ensure normality which worked very effectively provided that the numerical solution was not identically zero. In the case when this approach was not applicable we proposed an implicit least-squares imposition of the normality. This not only effectively imposed normality but also maintained the proof of convergence for this PGD algorithm by preserving the convexity of the Rayleigh-Ritz functional $\mathscr{J}$. We also introduced streamline upwinding into our PGD approximation and demonstrated that this very effectively stabilised the approximations we obtained.

We also investigated the application of a PGD algorithm to the fully non-homogeneous Fokker-Planck equation defined in both physical and configuration space. We began by following a scheme proposed in the paper of Figueroa and Süli [69] which used an operator splitting scheme to separate the physical space problem from the configuration space problem. One could then directly use the methodology for the configuration space problem in Section 4.2 coupled with a suitable solver in physical space. Unfortunately, we showed that such an operator splitting scheme is not suitable to be applied in conjunction with the PGD as we either need to solve a number of physical space problems which grows exponentially with the dimension of the configuration space or we are left with an ill-posed problem. To this end we proposed using a PGD approximation of the full Fokker-Planck equation which uses a separated representation in both the configuration variables and the physical variable. In particular, we presented two schemes based on this concept: A fully implicit scheme and a semi-implicit scheme. It remains as further work to couple these schemes for approximating the solution to the full Fokker-Planck equation to macroscopic flow problems. Furthermore, it remains to validate the performance of the schemes and to compare their convergence and stability properties.

Another area for further work on this subject is to make use of the theory developed in Chapter 3 to construct an efficient least-squares PGD algorithm for the solution of the Fokker-Planck equation. This would enable one to use a fully implicit time integration scheme while still being in a position where convergence of the PGD algorithm can be proved. We previously listed a number of issues which need to be addressed before pursuing this approach: Firstly we need to find a way to effectively stabilise least-squares PGD approximations of convection dominated convection-diffusion equations and secondly we need to carefully derive the energy balances in an ad-hoc manner to obtain balances in the Maxwellian weighted Sobolev spaces. It would then certainly be of interest to us to see how well such a least-squares PGD algorithm would perform for this problem.

Chapter 5

Conclusions and Further Work

In this thesis we have thoroughly investigated proper generalised decompositions. We firstly introduced the progressive Galerkin PGD and presented numerical results for both the Poisson and Stokes problems. A spectral element discretisation was employed and the expected optimal rates of convergence were observed provided that the rank of the PGD approximation was sufficiently high. We further demonstrated how the PGD could be extended to problems with non-homogeneous Dirichlet boundary conditions and described how it could be extended to problems defined in different geometries. We also reviewed two unique approaches to proving convergence of greedy algorithms which could be associated with these PGD algorithms under certain assumptions. We noted that a progressive Galerkin PGD algorithm for the Poisson equation satisfied these assumptions whereas the Stokes problem did not due to its weak coercivity. Furthermore, we showed that stability conditions related to the weak coercivity (i.e. the LBB condition in the Stokes problem) were not guaranteed to be satisfied in the PGD. Our numerical experiments reflected this observation by the inconsistency and unreliability of the PGD algorithm for the Stokes problem.

In Chapter 3 we investigated PGD algorithms based on least-squares formulations rather than Galerkin formulations. This concept was very similar to that of minimal residual PGD algorithms although we made the point of using the least-squares PGD terminology since we wanted to highlight that these algorithms were based on rigorously defined least-squares principles. Research into least-squares PGDs was motivated by the two issues we encountered with a Galerkin PGD algorithm for the Stokes problem: lack of a proof of convergence and no guarantee of LBB-like stability. A least-squares formulation of the problem provides a strongly coercive setting instead of the weakly coercive Galerkin formulation which was exactly the source of these two issues. Before developing least-squares PGD algorithms for the Stokes problem we considered both the Poisson and convection-diffusion equations. In the case of the Poisson problem we developed least-squares PGD algorithms based on a homogeneous elliptic and non-homogeneous elliptic first-order reformulation of the problem. We demonstrated the superiority of the algorithms based on homogeneous elliptic formulations and, in particular, showed that we could obtain comparable rates of convergence with a Galerkin formulation of the same problem. For the convection-diffusion equation we noted a severe degradation of the convergence rates when the magnitude of the convection term was increased. We left it as further work to apply or develop efficient methods of stabilisation for a least-squares PGD formulation of convection dominated problems. Finally, we developed least-squares PGDs based on homogeneous elliptic and non-homogeneous elliptic formulations of the Stokes problem and made the same conclusions as for the Poisson problem. In this case the homogeneous elliptic formulations provided a least-squares PGD which yielded significantly better rates of convergence that the non-homogeneous elliptic equivalents. We also successfully applied this algorithm to the simple benchmark problem of the flow of a Newtonian fluid in a lid driven cavity.

The final work of this thesis concentrated on the specific application of the PGD to the kinetic theory modeling of complex flows. In particular, we considered a deterministic approach rather than the more widely applied stochastic treatment of kinetic theory models. This requires the solution of the deterministic (but potentially high-dimensional) Fokker-Planck equation. The PGD is particularly well-suited for this problem since it considers a separated representation of the unknown field which significantly increases the tractability of high-dimensional problems. We began by considering the Fokker-Planck equation defined only in configuration space which only models the evolution of a probability distribution function governing the possible configurations of a FENE bead-spring chain model of a polymer chain. It does not take into consideration the movement of the centres of mass of the bead-spring chains in physical space and hence it is not suitable to be coupled with a macroscopic flow problem. It is still instructive to consider the problem and it is certainly still challenging as the high-dimensionality of the problem largely comes from this configuration space. A Maxwellian weighted reformulation of the Fokker-Planck was considered and results were presented in the case of a two spring-three bead chain model in one dimension. We developed methods for imposing the normality condition which is a required property of probability distribution functions and also stabilised our approximations by employing streamline upwinding. We then considered the fully non-homogeneous Fokker-Planck equation which does take into account the movement of the centres of mass of the bead-spring chains and hence is defined in both configuration and physical space. We showed that an operator splitting scheme, which is often employed for this problem, is not suitable when using a PGD approximation in configuration space. Instead, we developed two schemes based on a PGD algorithm that seeks a separated representation both in its configurational variables and the physical variable which remain to be thoroughly tested.

At the end of Chapters 3 and 4 we presented a number of future developments of this work specific to the subject of each chapter. In this concluding chapter we provide some thoughts on potential future developments in the much wider scope of the proper generalised decomposition and its applications. The first thing to note is that PGDs are quickly becoming widely employed in a huge variety of applications in computational mathematics and engineering, which we made apparent in Section 1.3. The theoretical understanding of these algorithms, however, lags behind the applications and progress needs to be made to accommodate the increasing number of applications.

One of the overriding themes of this thesis has been the convergence of PGD algorithms. As we stated earlier in Section 2.4.1, we were able to prove convergence of a greedy algorithm where at each stage a minimisation problem is solved. In our implementation of the PGD it is the Euler-Lagrange equations associated with this minimisation that is solved since it is significantly cheaper. However, since we are seeking solutions in the nonlinear manifold, $\mathcal{S}_1$, of rank-one tensors this is not equivalent to solving the minimisation problem. A big question is then: is it possible to prove convergence of PGD algorithms when at each iteration we solve the Euler-Lagrange equation. Le Bris et al. [95] showed that this assumption was enough to prove that the PGD algorithm converged but not necessarily that it converged to the correct solution. A further consideration is when solving a problem that does not possess such an equivalent minimisation problem such as the non-symmetric convection-diffusion equation. In this case it is not clear how to even define a greedy algorithm let alone prove convergence thereof. However, Galerkin PGD algorithms have still been developed for problems of this type which still yield promising results that may imply that they do converge (e.g. the fully implicit or space-time treatment of the Fokker-Planck equation by Ammar et al. [8,9]).

There are also other aspects of the actual implementation of PGD algorithms that are not accounted for in the theory. Firstly, it is not yet clear what role the discretisation has to play in the convergence of PGD algorithms. Secondly, it is not clear how the alternating directions fixed point linearisation will effect convergence. Getting a better understanding of this could lead to much more efficient discretisations and potentially an alternative improved linearisation.

A final theoretical development that needs to be made is a better understanding of the convergence rates of PGD algorithms. This would not only provide a great deal of insight into how these algorithms work but it would also lead to practical developments such as extremely accurate error estimators. These could be used as global stopping criterion or used to design efficient adaptive strategies.

Of course, there is always more room for further applications of PGD algorithms.

One which is of particular interest is the application to Schrödinger's equation in quantum chemistry. The dimension of this problem depends on the number of particles you choose to model interacting with each other. If one were able to use the PGD to solve the very high-dimensional problems one obtains with a complex system of particles then it could reveal new understanding into the workings of the universe. Chinesta et al. [49] had previously attempted an application of the PGD to this problem but found difficulty in applying Pauli's exclusion principle. In order to apply this, the number of PGD modes one needs to include effectively increases factorially with the dimension of the problem due to the inclusion of Slater determinants. This problem is also shared by the previously mentioned post-Hartree Fock method. Finding a way to efficiently impose Pauli's exclusion principle could have a huge impact on the numerical approximation of Schrödinger's equation.

There are also a huge number of potential applications of the PGD to parametrised models. We had previously mentioned that it is already being employed in the, currently very popular, area of simulation of surgery. It also shows great promise for simulating any number of problems in computational engineering in close to real time. It seems that the potential areas of application for the PGD are practically endless and I, personally, am very excited to see what impact this new method will have on computational science in the future.

Bibliography

[1] S. Agmon, A. Douglis, and L. Nirenberg, *Estimates near the boundary for solutions of elliptic partial differential equations satisfying general boundary conditions I.*, Commun. Pur. Appl. Math., 12 (1959), pp. 623–727.

[2] ——, *Estimates near the boundary for solutions of elliptic partial differential equations satisfying general boundary conditions II.*, Commun. Pur. Appl. Math., 17 (1964), pp. 35–92.

[3] A. Ammar, F. Chinesta, and E. Cueto, *Coupling finite elements and proper generalized decompositions*, Int. J. Mult. Comp. Eng., 9 (2011), pp. 17–33.

[4] A. Ammar, F. Chinesta, E. Cueto, and M. Doblaré, *Proper generalized decomposition of time-multiscale models*, Int. J. Numer. Meth. Eng., 90 (2012), pp. 569–596.

[5] A. Ammar, F. Chinesta, P. Díez, and A. Huerta, *An error estimator for separated representations of highly multidimensional models*, Comput. Methods Appl. Mech. Engrg., 199 (2010), pp. 1872–1880.

[6] A. Ammar, F. Chinesta, and A. Falcó, *On the convergence of a greedy rank-one update algorithm for a class of linear systems*, Arch. Comput. Methods Eng., 17 (2010), pp. 473–486.

[7] A. Ammar, A. Huerta, F. Chinesta, E. Cueto, and A. Leygue, *Parametric solutions involving geometry: A step towards efficient shape optimization*, Comput. Methods Appl. Mech. Engrg., 268 (2014), pp. 178–193.

[8] A. Ammar, B. Mokdad, F. Chinesta, and R. Keunings, *A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids*, J. Non-Newtonian Fluid Mech., 139 (2006), pp. 153–176.

[9] ——, *A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. Part II: Transient simulation using space-time separated representations*, J. Non-Newtonian Fluid Mech., 144 (2007), pp. 98–121.

[10] D. N. Arnold, R. S. Falk, and R. Winther, *Differential complexes and stability of finite element methods I. the de Rham complex*, IMA V. Math., 142 (2006), pp. 23–46.

[11] A. K. Aziz, R. B. Kellogg, and A. B. Stephens, *Least squares methods for elliptic systems*, Math. Comput., 44 (1985), pp. 53–70.

[12] I. Babuška, *Error-bounds for finite element method*, Numer. Math., 16 (1971), pp. 322–333.

[13] I. Babuška and B. Q. Guo, *The h, p and h-p version of the finite element method: basis theory and applications*, Adv. Eng. Softw., 15 (1992), pp. 159–174.

[14] J. W. Barrett and E. Süli, *Existence of global weak solutions to some regularized kinetic models for dilute polymers*, Multiscale Model. Sim., 6 (2007), pp. 506–546.

[15] ——, *Existence and equilibration of global weak solutions to kinetic models for dilute polymers I: Finitely extensible nonlinear bead-spring chains*, Math. Mod. Meth. Appl. S., 21 (2011), pp. 1211–1289.

[16] ——, *Existence of global weak solutions to finitely extensible nonlinear bead-spring chain models for dilute polymers with variable density and viscosity*, J. Differ. Equations, 253 (2012), pp. 3610–3677.

[17] B. C. Bell and K. S. Surana, *A space-time coupled p-version least-squares finite element formulation for unsteady fluid dynamics problems*, Int. J. Numer. Meth. Eng., 37 (1994), pp. 3545–3569.

[18] G. Beylkin and M. J. Mohlenkamp, *Numerical operator calculus in higher dimensions*, Proc. Natl. Acad. Sci., 99 (2002), pp. 10246–10251.

[19] A. V. Bhave, R. C. Armstrong, and R. A. Brown, *Kinetic theory and rheology of dilute, nonhomogeneous polymer solutions*, J. Chem. Phys., 95 (1991), pp. 2988–3000.

[20] M. Billaud-Friess, A. Nouy, and O. Zahm, *A tensor approximation method based on ideal minimal residual formulations for the solution of high-dimensional problems*, ESAIM-Math. Model. Num., 48 (2014), pp. 1777–1806.

[21] R. B. Bird, C. F. Curtiss, R. C. Armstrong, and O. Hassager, *Dynamics of Polymeric Liquids: Volume 2 Kinetic Theory*, John Wiley & Sons, New York, 1987.

[22] P. B. Bochev, *Analysis of least-squares finite element methods for the Navier-Stokes equations*, SIAM J. Numer. Anal., 34 (1997), pp. 1817–1844.

[23] ——, *Least-squares finite element methods for first-order elliptic systems*, Int. J. Num. Anal. Mod., 1 (2004), pp. 49–64.

[24] P. B. Bochev and M. D. Gunzburger, *Analysis of least-squares finite element methods for the Stokes equations*, Math. Comput., 63 (1994), pp. 479–506.

[25] ——, *Least-Squares Finite Element Methods*, Springer-Verlag, New York, 2009.

[26] B. Bognet, F. Bordeu, F. Chinesta, A. Leygue, and A. Poitou, *Advanced simulation of models defined in plate geometries: 3D solutions with 2D computational complexity*, Comput. Methods Appl. Mech. Engrg., 201–204 (2012), pp. 1–12.

[27] J. H. Bramble, R. D. Lazarov, and J. E. Pasciak, *A least-squares approach based on a discrete minus one inner product for first order systems*, Tech. Rep. 94-32, Mathematical Science Insitute, Cornell University, 1994.

[28] J. H. Bramble and A. H. Schatz, *Rayleigh-Ritz-Galerkin methods for Dirichlet's problem using subspaces without boundary conditions*, Commun. Pur. Appl. Math., 23 (1970), pp. 653–675.

[29] ——, *Least squares methods for* $2m$*th order elliptic boundary-value problems*, Math. Comput., 25 (1971), pp. 1–32.

[30] F. Brezzi, *On the existence, uniqueness and approximation of saddle-point problems arising from Lagrangian multipliers*, ESAIM-Math. Model. Num., 8 (1974), pp. 129–151.

[31] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer-Verlag, New York, 1991.

[32] F. Brezzi, J. Rappaz, and P. A. Raviart, *Finite dimensional approximation of nonlinear problems. Part I: Branches of nonsingular solutions*, Numer. Math., 36 (1980), pp. 1–25.

[33] A. N. Brooks and T. J. R. Hughes, *Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations*, Comput. Methods Appl. Mech. Engrg., 32 (1982), pp. 199–259.

[34] H. J. Bungartz and M. Griebel, *Sparse grids*, Acta Numerica, 13 (2004), pp. 1–123.

[35] J. Burkardt, M. Gunzburger, and H. Lee, *POD and CVT-based reduced-order modeling of Navier-Stokes flows*, Comput. Methods Appl. Mech. Engrg., 196 (2006), pp. 337–355.

[36] Z. Cai, T. A. Manteuffel, and S. F. McCormick, *First-order system least squares for the Stokes equations, with applications to linear elasticity*, SIAM J. Numer. Anal., 34 (1997), pp. 1727–1741.

[37] E. Cancès, V. Ehrlacher, and T. Lelièvre, *Convergence of a greedy algorithm for high-dimensional convex nonlinear problems*, Math. Models Methods Appl. Sci., 21 (2011), pp. 2433–2467.

[38] ——, *Greedy algorithms for high-dimensional non-symmetric linear problems*, ESAIM: Proc., 41 (2013), pp. 95–131.

[39] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods: Fundamentals in Single Domains*, Springer-Verlag, Berlin, 2006.

[40] C. Chancellor, A. Ammar, F. Chinesta, M. Magnin, and O. Roux, *Linking discrete and stochastic models: The chemical master equation as a bridge between process hitting and proper generalized decomposition*, Lect. Notes. Comput. Sc., 8130 (2013), pp. 50–63.

[41] C. L. Chang, *Finite element approximation for grad-div type systems in the plane*, SIAM J. Numer. Anal., 29 (1992), pp. 452–461.

[42] A. Chatterjee, *An introduction to the proper orthogonal decomposition*, Current Science, 78 (2000), pp. 808–817.

[43] C. Chauvière, *Stabilized spectral element methods for the simulation of viscoelastic flows*, PhD thesis, EPFL, 2001.

[44] C. Chauvière and A. Lozinski, *Simulation of dilute polymer solutions using a Fokker-Planck equation*, Comput. Fluids, 33 (2004), pp. 687–696.

[45] H. Chen, G. Fu, J. Li, and W. Qiu, *First order least square method with weakly imposed boundary condition for convection dominated diffusion problems*, Comput. Math. Appl., 68 (2014), pp. 1635–1652.

[46] F. Chinesta, A. Ammar, and E. Cueto, *On the use of proper generalized decompositions for solving the multidimensional chemical master equation*, Eur. J. Comput. Mech., 19 (2010), pp. 53–64.

[47] ——, *Proper generalized decomposition of multiscale models*, Int. J. Numer. Meth. Eng., 83 (2010), pp. 1114–1132.

[48] ——, *Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models*, Arch. Comput. Methods Eng., 17 (2010), pp. 327–350.

[49] F. Chinesta, A. Ammar, and P. Joyot, *The nanometric and micrometric scales of the structure and mechanics of materials revisited: An introduction to the challenges of fully deterministic numerical descriptions*, Int. J. Multiscale Comput. Eng., 6 (2008), pp. 191–213.

[50] F. Chinesta, A. Ammar, A. Leygue, and R. Keunings, *An overview of the proper generalized decomposition with applications in computational rheology*, J. Non-Newtonian Fluid Mech., 166 (2011), pp. 578–592.

[51] F. Chinesta and E. Cueto, *PGD-Based Modeling of Materials, Structures and Processes*, Springer, 2014.

[52] F. Chinesta, R. Keunings, and A. Leygue, *The Proper Generalized Decomposition for Advanced Numerical Simulations*, Springer, 2013.

[53] F. Chinesta, P. Ladevèze, and E. Cueto, *A short review on model order reduction based on proper generalized decomposition*, Arch. Comput. Methods Eng., 18 (2011), pp. 395–404.

[54] F. Chinesta, A. Leygue, F. Bordeu, J. V. Aguado, E. Cueto, D. González, I. Alfaro, A. Ammar, and A. Huerta, *PGD-based computational vademecum for efficient design, optimization and control*, Arch. Comput. Methods Eng., 20 (2013), pp. 31–59.

[55] A. Cohen, *A Padé approximant to the inverse Langevin function*, Rheol. Acta, 30 (1991), pp. 270–273.

[56] V. de Silva and L.-H. Lim, *Tensor rank and ill-posedness of the best low-rank approximation problem*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 1084–1127.

[57] R. A. DeVore and V. N. Temlyakov, *Some remarks on greedy algorithms*, Adv. Comput. Math., 5 (1996), pp. 173–187.

[58] M. Doi and S. F. Edwards, *Dynamics of concentrated polymer systems*, J. Chem. Soc., Faraday Trans. 2, 74 (1978), pp. 1789–1832.

[59] J. Douglas and T. Dupont, *Alternating-direction Galerkin methods on rectangles*, in Numerical Solution of Partial Diferential Equations II, B. Hubbard, ed., SYNSPADE 1970, Academic Press Inc., 1971, pp. 133–214.

[60] A. Dumon, C. Allery, and A. Ammar, *Proper general decomposition (PGD) for the resolution of Navier-Stokes equations*, J. Comput. Phys., 230 (2011), pp. 1387–1407.

[61] C. Eckart and G. Young, *The approximation of one matrix by another of lower rank*, Psychometrika, 1 (1936), pp. 211–218.

[62] A. W. El-Kareh and L. G. Leal, *Existence of solutions for all Deborah numbers for a non-Newtonian model modified to include diffusion*, J. Non-Newtonian Fluid Mech., 33 (1989), pp. 257–287.

[63] H. Elman, D. Silvester, and A. Wathen, *Finite Elements and Fast Iterative Solvers*, Oxford University Press, Oxford, 2005.

[64] A. Falcó, *Algorithms and numerical methods for high dimensional financial market models*, Rev. Econ. Financ., 20 (2010), pp. 51–68.

[65] A. Falcó, L. Hilario, N. Montés, and M. C. Mora, *Numerical strategies for the Galerkin-proper generalized decomposition method*, Math. Comp. Mod., 57 (2013), pp. 1694–1702.

[66] A. Falcó and A. Nouy, *A Proper Generalized Decomposition for the solution of elliptic problems in abstract form by using a functional Eckart-Young approach*, J. Math. Anal. Appl., 376 (2011), pp. 469–480.

[67] ——, *Proper generalized decomposition for nonlinear convex problems in tensor Banach spaces*, Numer. Math., 121 (2012), pp. 503–530.

[68] J. M. Fiard, T. A. Manteuffel, and S. F. McCormick, *First-order system least squares (FOSLS) for convection-diffusion problems: Numerical results*, SIAM J. Sci. Comput., 19 (1998), pp. 1958–1979.

[69] L. E. Figueroa and E. Süli, *Greedy approximation of high-dimensional Ornstein-Uhlenbeck operators*, Found. Comput. Math., 12 (2012), pp. 573–623.

[70] D. González, I. Alfaro, C. Quesada, E. Cueto, and F. Chinesta, *Computational vademecums for the real-time simulation of haptic collision between nonlinear solids*, Comput. Methods Appl. Mech. Engrg., 283 (2015), pp. 210–223.

[71] D. González, A. Ammar, F. Chinesta, and E. Cueto, *Recent advances on the use of separated representations*, Int. J. Numer. Meth. Engng., 81 (2010), pp. 637–659.

[72] D. González, E. Cueto, F. Chinesta, P. Díez, and A. Huerta, *Streamline upwind/Petrov-Galerkin-based stabilization of proper generalized decompositions for high-dimensional advection-diffusion equations*, Int. J. Numer. Meth. Eng., 94 (2013), pp. 1216–1232.

[73] W. J. Gordon and C. A. Hall, *Transfinite element methods: Blending-function interpolation over arbitrary curved element domains*, Numer. Math., 21 (1973), pp. 109–129.

[74] P. Grisvard, *Elliptic Problems in Nonsmooth Domains*, Pitman, Boston, 1985.

[75] J. L. Guermond, *A finite element technique for solving first-order PDEs in L^p*, SIAM J. Numer. Anal., 42 (2004), pp. 714–737.

[76] D. Rh. Gwynllyw and T. N. Phillips, *On the enforcement of the zero mean pressure condition in the spectral element approximation of the Stokes problem*, Comput. Methods Appl. Mech. Engrg., 195 (2006), pp. 1027–1049.

[77] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimisation Algorithms I.*, Springer-Verlag, Berlin, 1993.

[78] P.-W. Hsieh and S.-Y. Yang, *A novel least-squares finite element method enriched with residual-free bubbles for solving convection-dominated problems*, SIAM J. Sci. Comput., 32 (2010), pp. 2047–2073.

[79] B.-N. Jiang, *The Least-Squares Finite Element Method: Theory and Applications in Computational Fluid Dynamics and Electromagnetics*, Springer-Verlag, Berlin, 1998.

[80] B. Jourdain, T. Lelièvre, and C. Le Bris, *Existence of solution for a micro-macro model of polymeric fluid: the FENE model*, J. Funct. Anal., 209 (2004), pp. 162–193.

[81] R. Keunings, *Micro-macro methods for the multiscale simulation of viscoelastic flow using molecular models of kinetic theory*, in Rheology Reviews, D. M. Binding and K. Walters, eds., British Society of Rheology, 2004, pp. 67–98.

[82] D. J. Knezevic, *Analysis and implementation of numerical methods for simulating dilute polymeric fluids*, PhD thesis, Oxford University, 2008.

[83] D. J. Knezevic and E. Süli, *A heterogeneous alternating-direction method for a micro-macro dilute polymeric fluid model*, ESAIM-Math. Model. Num., 43 (2009), pp. 1117–1156.

[84] ——, *Spectral Galerkin approximation of Fokker-Planck equations with unbounded drift*, ESAIM-Math. Model. Num., 43 (2009), pp. 445–485.

[85] T. G. Kolda, *Orthogonal tensor decompositions*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 243–255.

[86] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Rev., 51 (2009), pp. 455–500.

[87] A. N. Kolmogorov, *Über die analytischen methoden in der wahrscheinlichkeitsrechnung*, Math. Ann., 104 (1931), pp. 415–458.

[88] S. V. Konyagin and V. N. Temlyakov, *Rate of convergence of pure greedy algorithm*, East. J. Approx., 5 (1999), pp. 493–499.

[89] P. Ladevèze, *Nonlinear computational structural mechanics: New approaches and non-incremental methods of calculation*, Springer, Berlin, 1999.

[90] P. Ladevèze and L. Chamoin, *On the verification of model reduction methods based on the proper generalized decomposition*, Comput. Methods Appl. Mech. Engrg., 200 (2011), pp. 2032–2047.

[91] P. Ladevèze, J.-C. Passieux, and D. Néron, *The LATIN multiscale computational method and the Proper Generalized Decomposition*, Comput. Methods Appl. Mech. Engrg., 199 (2010), pp. 1287–1296.

[92] O. A. Ladyzhenskaya, *The Mathematical Theory of Viscous Incompressible Flows*, Gordon and Breech, New York, 1969.

[93] M. Laso and H. C. Öttinger, *Calculation of viscoelastic flow using molecular models: the CONNFESSIT approach*, J. Non-Newtonian Fluid Mech., 47 (1993), pp. 1–20.

[94] P. D. Lax and A. N. Milgram, *Parabolic equations*, in Contributions to the theory of partial differential equations, Annals of Mathematics Studies, no. 33, Princeton University Press, Princeton, N. J., 1954, pp. 167–190.

[95] C. Le Bris, T. Lelièvre, and Y. Maday, *Results and questions on a nonlinear approximation approach for solving high-dimensional partial differential equations*, Constr. Approx., 30 (2009), pp. 621–651.

[96] G. M. Leonenko and T. N. Phillips, *On the solution of the Fokker-Planck equation using a high-order reduced basis approximation*, Comput. Methods Appl. Mech. Engrg., 199 (2009), pp. 158–168.

[97] ——, *The prediction of plane Couette flow for a FENE fluid using a reduced basis approximation of the Fokker-Planck equation*, Int. J. Mult. Comp. Eng., 9 (2011), pp. 73–88.

[98] C. Liu and H. Liu, *Boundary conditions for the microscopic FENE models*, SIAM J. Appl. Math., 68 (2008), pp. 1304–1315.

[99] A. Lozinski, R. G. Owens, and J. Fang, *A Fokker-Planck-based numerical method for modelling non-homogeneous flows of dilute polymeric solutions*, J. Non-Newtonian Fluid Mech., 122 (2004), pp. 273–286.

[100] A. Lozinski, R. G. Owens, and T. N. Phillips, *The Langevin and Fokker-Planck equations in polymer rheology*, in Handbook of Numerical Analysis, Special Volume: Numerical Methods for Non-Newtonian Fluids, P. G.

Ciarlet, R. Glowinski, and J. Xu, eds., vol. 16, Elsevier B.V., North-Holland, 2011, pp. 211–303.

[101] Y. Maday, A. T. Patera, and E. M. Rønquist, *The $P_N - P_{N-2}$ method for the approximation of the Stokes problem*, Technical Report 92025, Laboratoire d'Analyse Numérique, Université Pierre et Marie Curie, 1992.

[102] N. Moës, J. Dolbow, and T. Belytschko, *A finite element method for crack growth without remeshing*, Int. J. Numer. Meth. Engng., 46 (1999), pp. 131–150.

[103] B. Mokdad, E. Pruliere, A. Ammar, and F. Chinesta, *On the simulation of kinetic theory models of complex fluids using the Fokker-Planck approach*, Appl. Rheol., 17 (2007), pp. 1–14.

[104] S. Niroomandi, D. González, I. Alfaro, F. Bordeu, A. Leygue, E. Cueto, and F. Chinesta, *Real-time simulation of biological soft tissues: a PGD approach*, Int. J. Numer. Meth. Biomed. Engng., 29 (2013), pp. 586–600.

[105] A. Nouy, *A generalized spectral decomposition technique to solve a class of linear stochastic partial differential equations*, Comput. Methods Appl. Mech. Engrg., 196 (2007), pp. 4521–4537.

[106] ——, *A priori model reduction through proper generalized decomposition for solving time-dependent partial differential equations*, Comput. Methods Appl. Mech. Engrg., 199 (2010), pp. 1603–1626.

[107] J. G. Oldroyd, *On the formulation of rheological equations of state*, Proc. Roy. Soc. Lond. A., 200 (1950), pp. 523–541.

[108] R. G. Owens, C. Chauvière, and T. N. Phillips, *A locally-upwinded spectral technique (LUST) for viscoelastic flows*, J. Non-Newtonian Fluid Mech., 108 (2002), pp. 49–71.

[109] R. G. Owens and T. N. Phillips, *Computational Rheology*, Imperial College Press, London, 2002.

[110] A. T. Patera, *A spectral element method for fluid dynamics: Laminar flow in a channel expansion*, J. Comput. Phys., 54 (1984), pp. 468–488.

[111] M. M. J. Proot, *The Least-Squares Spectral Element Method*, PhD thesis, Delft University of Technology, 2003.

[112] M. M. J. Proot and M. I. Gerritsma, *A least-squares spectral element formulation for the Stokes problem*, J. Sci. Comput., 17 (2002), pp. 285–296.

[113] ——, *Least-squares spectral elements applied to the Stokes problem*, J. Comput. Phys., 181 (2002), pp. 454–477.

[114] E. Pruliere, F. Chinesta, and A. Ammar, *On the deterministic solution of multidimensional parametric models using the proper generalized decomposition*, Math. Comput. Simul., 81 (2010), pp. 791–810.

[115] P. E. Rouse, *A theory of the linear viscoelastic properties of dilute solutions of coiling polymers*, J. Chem. Phys., 21 (1953), pp. 1272–1280.

[116] V. L. Rvachev and T. I. Sheiko, *R-functions in boundary value problems in mechanics*, Appl. Mech. Rev., 48 (1995), pp. 151–189.

[117] V. L. Rvachev, T. I. Sheiko, V. Shapiro, and I. Tsukanov, *Transfinite interpolation over implicitly defined sets*, Comput. Aided Geom. D., 18 (2001), pp. 195–220.

[118] J. D. Schieber, *Generalized Brownian configuration fields for Fokker-Planck equations including center-of-mass diffusion*, J. Non-Newtonian Fluid Mech., 135 (2006), pp. 179–181.

[119] H. R. Schwarz, *Finite Element Methods*, Academic Press, London, 1988.

[120] P. N. Shankar, *Slow Viscous Flows: Qualitative Features and Quantitative Analysis Using Complex Eigenfunction Expansions*, Imperial College Press, London, 2007.

[121] E. S. G. Shaqfeh and R. P. Jagadeeshan, *International Workshop on Mesoscale and Multiscale Description of Complex Fluids (IWMMCOF'06), Prato, Italy, July 5-8, 2006*, Appl. Rheol., 16 (2006), pp. 340–341.

[122] J. Shen, *Efficient spectral-Galerkin methods III: Polar and cylindrical geometries*, SIAM J. Sci. Comput., 18 (1997), pp. 1583–1604.

[123] C. D. Sherrill and H. F. Schaefer, *The Configuration Interaction Method: Advances in highly correlated approaches*, Adv. Quantum Chem., 34 (1999), pp. 143–269.

[124] L. Sirovich, *Turbulence and the dynamics of coherent structures, I-III*, Quart. J. Appl. Math, 45 (1987), pp. 561–590.

[125] L. Tamellini, O. le Maître, and A. Nouy, *Model reduction based on proper generalized decomposition for the stochastic steady incompressible Navier-Stokes equations*, SIAM J. Sci. Comput., 36 (2014), pp. A1089–A1117.

[126] R. Temam, *Navier-Stokes Equations: Theory and Numerical Analysis*, North Holland, Amsterdam, 1977.

[127] V. N. Temlyakov, *The best m-term approximation and greedy algorithms*, Adv. Comput. Math., 8 (1998), pp. 249–265.

[128] L. N. Trefethen and D. Bau, *Numerical Linear Algebra*, SIAM, Philadelphia, PA, 1997.

[129] H. R. Warner, *Kinetic theory and rheology of dilute suspensions of finitely extendible dumbbells*, Ind. Eng. Chem. Fund., 11 (1972), pp. 379–387.

[130] A. Yeckel and J. J. Derby, *On setting a pressure datum when computing incompressible flows*, Int. J. Numer. Meth. Fluids, 29 (1999), pp. 19–34.