Petrov-Galerkin Proper Generalized Decomposition strategies for convection-diffusion problems

Treball realitzat per:
Rafel Perelló i Ribas

Dirigit per:
Sergio Zlotnik Martínez
Pedro Díez Mejía

Màster en:
Mètodes Numèrics en Enginyeria

Barcelona, juny de 2020

Departament d'Enginyeria Civil i Ambiental

TREBALL FINAL DE MÀSTER

Petrov-Galerkin Proper Generalized Decomposition strategies for convection-diffusion problems

by
Rafel Perelló i Ribas

Thesis submitted in partial fulfillment of the requirements for the degree of
Master on Numerical Methods in Engineering

Universitat Politècnica de Catalunya
International Centre for Numerical Methods in Engineering

June 2020

Abstract

The present thesis explores the capabilities of a dual based Petrov-Galerkin Proper Generalised Decomposition (PGD) method [3] to solve non self-adjoint parametric partial differential equations (PDE). The (consistent) Galerkin PGD method is used as reference. The Petrov-Galerkin PGD method is formulated as a Galerkin PGD making possible a non-intrusive implementation.

Then, different problems are introduced and solved using the Petrov-Galerkin PGD methodology by separating the domain in space and time. In particular, a transient advection-diffusion problem is solved using a streamline upwind Petrov-Galerkin (SUPG) stabilisation. We also study a transient advection-diffusion problem where the Dirichlet/Neumann splitting of the boundary conditions (BC) is time depending.

The traditional fixed-point algorithm to solve the rank-one problem lacks of convergence for certain conditions. We introduce several alternative approaches to solve this inconvenient.

Finally, we present a few examples of transient parametric PDE solved using the introduced methodology.

TABLE OF CONTENTS

Chapter 1

Introduction

In the industry environment, over the past years, there has been an increasing interest in obtaining solutions of parametric partial differential equations. The applications of parametric PDEs include design optimisation and simulation of stochastic processes among others. The major limitation of parametric PDEs is the exponential increase of the computational cost with respect to the number of parameters.

In this context, different model reduction techniques have been designed. Proper Generalised Decomposition is widely used to obtain a priori separated approximations of high dimensional PDEs. The PGD methodology has been successfully applied to many different problems. However, conventional PGD algorithms fail to compute efficient approximations of transient problems, i.e. problems where the time dimension is treated as an additional parametric dimension. The reason of this phenomenon relies on the fact that in transient problems, the associated bilinear form is not a valid inner product.

In the present thesis we focus on solving problems in the space-time domain performing a tensor product separation of the functions of space and time. In particular, we solve the transient advection-diffusion equation. For that, we use the Petrov-Galerkin PGD method introduced by A. Nouy [3]. However, in the context of order reduction techniques, the motivation to separate the spatial and temporal dimensions is not, in general, to solve transient PDEs involving just a space-time domain but parametric transient PDEs over a high dimensional domain. For this reason, all methodologies presented are formulated to include the possibility of an arbitrary number of separated dimensions.

1.1 Contributions and outline

The main contributions of this thesis are:

– Implementation of the Petrov-Galerkin PGD algorithm as a Galerkin PGD method using a (relatively) non-intrusive methodology.

– Implementation of the update procedure as a Galerkin PGD method using a non-intrusive methodology.

– Stabilisation of the separated solution in advection dominated problems.

– Introduction of a method for solving the problem of time depending (or, more generally, parameter depending) boundary condition (BC) type. That is, in this problem, the part of the boundary where essential (and natural) BCs are imposed depends on time (or the parameters).

– Stabilisation of oscillations in the time dimension arising in time depending BC type problems.

– Different alternative algorithms to compute the rank-one solution of the PGD algorithm.

We consider a PGD method as non-intrusive if the rank-one problems that appear in the execution of the algorithm can be solved by a rank-one Galerkin PGD solver without any modification.

The thesis is organised six chapters (excluding the introduction and conclusions). In chapter 2 the problem is precisely described, as well as the algorithm used to compute the full order solution for reference purposes.

Chapters 3 and 4 introduce the formulation of the Galerkin and Petrov-Galerkin PGD methods. It is also presented how to compute the contribution of previously computed modes and an algorithm to update the previous modes of some previously defined dimensions. In addition, chapter 4 presents the formulation of the Petrov-Galerkin method as a Galerkin one by enlarging the space of functions.

In chapter 5 a SUPG stabilisation is described to stabilise the spatial dimension in advection dominated problems, and the time dimension in problems where Dirichlet BCs are enforced at some points of the boundary that were initially free.

Chapter 6 states the problem of time/parameter dependent BC type. The corresponding separated form is also presented.

Finally, in chapter 7 some algorithms for solving the rank-one problem are introduced. These algorithms improve the convergence properties of the traditional fixed-point method.

In all chapters where different methodologies are introduced, examples are included.

Chapter 2

Problem statement

In this chapter we state precisely the problem to solve in the foregoing of the thesis. For the sake of generality, all introduced methods assume an arbitrary number of separated dimensions and arbitrary bilinear and linear separable forms on the full order space. For this reason, we introduce the abstract formulation of the problem as well of the methodologies if they make sense. Then, we particularise to the example of the transient advection-diffusion problem.

2.1 Abstract formulation

Let $\{\mathcal{V}_\alpha\}_{\alpha\in A}$ be a family of Hilbert spaces over $\mathbb{R}$ (define the number of dimensions $n_D := |A| < \infty$). Let define the tensor product space as $\mathcal{V} := \bigotimes_{\alpha\in A}\mathcal{V}_\alpha$. We want to solve the following

Problem 1: Assume $B(\cdot,\cdot)$ is a continuous bilinear form on $\mathcal{V}$ and $L(\cdot)$ is a continuous linear functional on $\mathcal{V}$. In addition assume that both have the following separability properties:

$$B\Big(\bigotimes_{\alpha\in A} u_\alpha, \bigotimes_{\alpha\in A} v_\alpha\Big) = \sum_{m\in M_B}\prod_{\alpha\in A} B^m_\alpha(u_\alpha, v_\alpha),$$
$$L\Big(\bigotimes_{\alpha\in A} u_\alpha\Big) = \sum_{m\in M_L}\prod_{\alpha\in A} L^m_\alpha(u_\alpha).$$

Where $|M_B| < \infty$ and $|M_L| < \infty$ are the number of modes of the bilinear and linear form, B^m_α is a continuous bilinear form on $\mathcal{V}_\alpha$ and L^m_α is a continuous linear functional on $\mathcal{V}_\alpha$. Then, find $u \in \mathcal{V}$ such that

$$B(u,v) = L(v) \quad \forall v \in \mathcal{V}. \tag{2.1}$$

It is assumed that the problem is well posed, i.e. u exists, is unique and depends continuously on L. However, for the sake of reducing the computational power needed to solve the problem, we restrict the study of algorithms providing separated approximations of the solution. That is, we focus on algorithms providing an approximation of the solution u of the form

$$u \approx \sum_{m=1}^{M}\bigotimes_{\alpha\in A} u^m_\alpha.$$

Remark 2.1: The tensor product of two Hilbert spaces is symmetric up to a natural isomorphism. For this reason, we identify the tensor product of different spaces (and of their elements) without regarding the order in which the operation is performed.

2.2 Advection-diffusion formulation

For the sake of clarity, we present as example the problem of the transient advection-diffusion problem in conservative form.

2.2.1 Strong form

Problem 2: Let define the spatial domain as $\Omega = (0,1) \times (0,1)$ and the time interval as $T = (0,1)$. Then, find $u(x,t) \in C^1\big(T; C^2(\Omega)\big)$ that solves

$$\begin{aligned} u_t + \nabla \cdot (cu - \nu \nabla u) &= f \quad \text{in } \Omega \times T, \\ u &= u_D \quad \text{on } \Gamma_D \times T,\ \Gamma_D \subseteq \partial\Omega, \\ (cu - \nu\nabla u) \cdot n &= h \quad \text{on } \Gamma_N \times T,\ \Gamma_N = \partial\Omega \backslash \Gamma_D, \\ u &= u_0 \quad \text{on } \Omega \times \{0\}. \end{aligned} \tag{2.2}$$

Here and through all the thesis, the advection velocity is chosen to be divergence-free. This means that the conservative and non-conservative form are identical and can be used indistinguishable except in the treatment of the Neumann BCs.

2.2.2 Weak form

As usual, we reformulate the problem in weak form; we refer to [2]. Let define the Sobolev space of spatial functions as

$$\mathcal{V}_S = \mathcal{H}_1^D(\Omega) = \big\{ v \in \mathcal{H}_1(\Omega) : v|_{\Gamma_D} = 0 \big\}. \tag{2.3}$$

The space of weak solutions of the PDE is

$$\mathcal{V} = \big\{ u \in \mathcal{L}^2(T; \mathcal{V}_S) : u_t \in \mathcal{L}^2(T; \mathcal{V}_S') \big\}. \tag{2.4}$$

Let v_D be an arbitrary function in $\mathcal{L}^2(T; \mathcal{H}_1(\Omega))$ such that $(v_D)_t \in \mathcal{L}^2(T; \mathcal{H}_1'(\Omega))$ and its trace on Γ_N agrees with the Dirichlet BC, i.e. $v_D(t)|_{\Gamma_N} = u_D(t)$ for almost every (a.e.) $t \in T$. We decompose the solution of the PDE as $u^* = u + v_D$. Where u is the solution to the problem with homogeneous Dirichlet BC taking into account the contribution of the function v_D. We say that $u \in \mathcal{V}$ is a weak solution of such a problem if

$$B(u,v) = L(v) \quad \forall v \in \mathcal{V}. \tag{2.5}$$

Where

$$\begin{aligned} B(u,v) &= \int_{\Omega\times T} u_t v + (c \cdot \nabla u) v \; d(\Omega \times T) + \nu \nabla u \cdot \nabla v + \int_\Omega u(0) v(0) \; d\Omega, \\ L(v) &= \int_{\Omega\times T} f v \; d(\Omega\times T) + \int_{\Gamma_N \times T} h v \; d(\partial\Omega \times T) + \int_\Omega u_0 v(0) \; d\Omega - B(v_D, v). \end{aligned}$$

Note that the initial condition is imposed in weak form. Using the standard PGD methods, one can impose the initial condition in strong form. However, as we note in chapter 5, the Petrov-Galerkin defines a dual problem that can be naturally interpreted as a backward in time advection-diffusion PDE if the initial condition is imposed weakly.

Before continuing, we say a word on the space $\mathcal{V}$. It consists on functions such that for a.e. $t \in T$ they assign a function in the space of spatial functions, i.e. $\mathcal{V} \ni v = (t \mapsto v(t) \in \mathcal{V}_S)$. Such functions must be square integrable; that is

$$\int_T \|v(t)\|_{\mathcal{V}_S}^2 \, dT < \infty.$$

To ensure that the term $\int_{\Omega\times T} u_t v d(\Omega \times T)$ is bounded, the additional restriction on the time derivative of u is imposed, i.e. $u_t \in \mathcal{L}^2(T; \mathcal{V}_S')$ and we identify such term with

$$\int_T \langle u_t(t), v(t) \rangle_{\mathcal{V}_S', \mathcal{V}_S} dT.$$

Where $\langle \cdot, \cdot \rangle_{\mathcal{V}_S', \mathcal{V}_S}$ denotes the duality evaluation in $\mathcal{V}_S$. That is, we consider $u_t(t)$ as a distribution in Ω.

It can be shown that $\mathcal{V} \hookrightarrow C(T; \mathcal{L}^2(\Omega))$. This ensures that the terms $u(0)$ and $v(0)$ are well defined and B and L are bounded. The regularities of each parameter needed for the weak problem to be well posed are

$$\begin{aligned} c &\in \mathcal{L}^\infty(T; \mathcal{L}^\infty(\Omega)), \\ \nu &\in \mathcal{L}^\infty(T; \mathcal{L}^\infty(\Omega)), \\ f &\in \mathcal{L}^2(T; \mathcal{V}_S'), \\ u_D &\in \mathcal{L}^2(T; \mathcal{H}^{1/2}(\partial\Omega)), \\ h &\in \mathcal{L}^2(T; \mathcal{L}^2(\partial\Omega)), \\ u_0 &\in \mathcal{L}^2(\Omega). \end{aligned}$$

For the sake of simplicity, in the first chapters we restrict the study to the case of the homogeneous Dirichlet problem, i.e. $\Gamma_D = \partial\Omega,\ u_D = 0$.

In chapter 6 we study the problem with more complex BC. In particular, we formulate the problem with both Dirichlet and Neumann non-homogeneous BC. We also study the problem of changing Dirichlet-Neumann BCs as a function of time. That is, the boundary is still partitioned with a Dirichlet and Neumann part but this splitting is time dependent. This peculiar problem arises commonly in engineering applications. A discussion about the well posedness of such a problem is out of the scope of the thesis, so we shall assume it is well posed.

2.2.3 Space-time separation

The space $\mathcal{V}$ has no obvious way to be expressed as the tensor product of a space of functions in space and a space of functions in time. For this reason we substitute the space $\mathcal{V}$ for a complete subspace of itself. This step does not have any affect on the problem if the parameters are regular enough. To show that, we remit to a regularity theorem.

Theorem 1: Assume that c and ν are smooth and constant in time; $u_0 \in \mathcal{H}_2(\Omega)$ and is compatible with the essential BC in the sense that $u_0 \in \mathcal{H}_1^0(\Omega)$; and $f, f_t \in \mathcal{L}^2(T; \mathcal{L}^2(\Omega))$. Then, $u_t \in \mathcal{L}^2(T; \mathcal{V}_S)$. [2]

With the assumption that $u_t \in \mathcal{L}^2(T; \mathcal{V}_S)$ the space of weak solutions can be written as $\mathcal{H}_1(T; \mathcal{V}_S(\Omega))$ which is straightforward to express as a tensor product space. What shows the theorem is that, for most practical applications, we can expect the same convergence properties with the tensor product formulation as for other classical discretisations techniques.

From now on we shall substitute the function space by

$$\mathcal{V} = \mathcal{H}_1(T; \mathcal{H}_1^D(\Omega)) \cong \mathcal{H}_1(T) \otimes \mathcal{H}_1^D(\Omega) =: \mathcal{V}_T \otimes \mathcal{V}_S. \tag{2.6}$$

Remark 2.2: The substitution of $\mathcal{V}$ by a complete subspace of itself has no effect on the discretised problem. That is, at the discretisation step one substitutes $\mathcal{V}$ with a finite dimensional subspace that for all common methods of discretisation is a subspace of the slightly smaller space $\mathcal{H}_1(T) \otimes \mathcal{H}_1^D(\Omega)$.

Remark 2.3: For the discrete case, substitute $\mathcal{V}_T$ and $\mathcal{V}_S$ for their discretisations, i.e. a finite dimensional subspace of themselves. Note that discontinuous functions in time discretisations are not allowed; this means that the problem is be global in time. Most common time discretisations techniques use the fact that information cannot propagate backwards in time to avoid solving global space-time problems. However, as we shall see, in the PGD context, one cannot avoid to solve global problems in time (but one has not to solve global problems in space-time).

Let denote for simplicity the tensor product of to elements $\omega \otimes \lambda$ by $\omega\lambda$. The separation of the forms reads

$$\begin{aligned}B(\omega\lambda,\omega^*\lambda^*) &= B^e_S(\omega,\omega^*)B^e_T(\lambda,\lambda^*) + B^d_S(\omega,\omega^*)B^d_T(\lambda,\lambda^*) + B^a_S(\omega,\omega^*)B^a_T(\lambda,\lambda^*)\\ &\quad + B^0_S(\omega,\omega^*)B^0_T(\lambda,\lambda^*) = \int_\Omega \omega\omega^*\ d\Omega \int_T \lambda_t\lambda^* dT + \int_\Omega \nu_S \nabla\omega\cdot\nabla\omega^*\ d\Omega \int_T \nu_T\lambda\lambda^*\ dT\\ &\quad + \int_\Omega (c_S\cdot\nabla\omega)\omega^*\ d\Omega \int_T c_T\lambda\lambda^*\ dT + \int_\Omega \omega\omega^*\ d\Omega\cdot\big(\lambda(0)\lambda^*(0)\big),\end{aligned}$$

$$L(\omega\lambda) = L^f_S(\omega)L^f_T(\lambda) + L^0_S(\omega)L^0_T(\lambda) = \int_\Omega f_S\omega\ d\Omega\ \int_T f_T\lambda\ dT + \int_\Omega u_0\omega\ d\Omega\cdot\lambda(0).$$

Again, $\lambda(0)$ is well defined as $\mathcal{H}_1(T)\hookrightarrow C(T)$.

Here we have assumed that all parameters c,ν,f are rank-one tensors, e.g. $c=c_S\otimes c_T$. In case that the separation of the parameters cannot be performed in one mode, multiple modes must be added to the forms. In case that the parameters are not separable using a finite number of terms (or the terms required are too high for computational purposes), they shall be substituted by some approximation using a low number of modes; we refer to [5].

2.3 Numerical examples: Reference solution

2.3.1 Convegence of PGD methods

To study the convergence of the formulated PGD methods, different numerical examples have been proposed. To obtain a measure of the error of the PGD schemes, we solve the problem using the standard Galerkin approach (global in space-time) and use that solution as reference.

The reference solution u is computed as the solution to the full order discretised problem (2.5). That is, it is computed in the discrete space $\mathcal{V}$ without using any separation scheme. As all discretisation methods, this solution has an error compared to the solution of the non-discrete weak problem. However, this error is not interesting for us. The PGD methods approximate the solution u of the discretised space and not the solution of the non-discrete problem. The errors we are interested in are the norms of the differences between u and the corresponding PGD method. Following the same criterion used by Nouy in his paper, we use the $\mathcal{L}_2(T;\mathcal{L}_2(\Omega))$ norm to measure the error, i.e. $\|e\|^2_{\mathcal{L}_2(T;\mathcal{L}_2(\Omega))} = \int_T \|e(t)\|^2_{\mathcal{L}_2(\Omega)}\, dT$.

The reference solution is computed in the space $\mathcal{V}_S\otimes\mathcal{V}_T$ (where $\mathcal{V}_S$ and $\mathcal{V}_T$ are finite-dimensional), the same one used in the PGD algorithms. This means that there exists a sequence of functions of finite number of modes such that it converges in the $\mathcal{H}_1(S)\otimes\mathcal{H}^D_1(T)$ norm to the full order solution; and a fortiori it converges in the $\mathcal{L}_2(\Omega)\otimes\mathcal{L}_2(T)$. The methods we present, compute an approximation by sequentially adding rank-one modes to the approximation (and maybe updating some of the dimensions). We expect that the sequence defined by the sequential addition of modes converges towards the reference solution (but not necessarily towards the solution of the non-discrete problem).

We know that the PGD methods are consistent in the sense that if the full order solution can be expressed in a separable way, then the PGD formulation using the full order solution as previous modes returns the same reference solution, i.e. the full solution is a fixed point of the PGD iterations. What it is unknown is under which conditions this fixed-point is stable and the PGD algorithm is convergent towards such a fixed-point. In latter chapters we show some examples of non-convergent solutions. They fail for two reasons: they diverge (cf. chapter 5) or they converge to a fixed-point that is not the reference solution (cf. chapter 6).

2.3.2 Computation of the full order solution

Now we expose how to compute the solution of the full order problem in the tensor product space. First, we use the separability properties of the forms and the fact that every dimension α is finite dimensional and define the matrix and vector of each mode and dimension as

$$\begin{aligned}\left[\boldsymbol{K}_\alpha^m\right]_{i,j} &= B_\alpha^m(N_j^\alpha, N_i^\alpha), \quad \alpha \in \{S,T\},\ m \in \{e,d,a,0\},\\ \left[\boldsymbol{f}_\alpha^m\right]_i &= L_\alpha^m(N_i^\alpha), \quad \alpha \in \{S,T\},\ m \in \{f,0\}.\end{aligned}$$

Where (N_i^α) denotes a basis for the corresponding space.

The matrix and vector associated to the system are defined as the sum of all the modes of the separated forms. That is

$$\begin{aligned}\boldsymbol{K}_{full} &= \sum_{m\in M_B} \big(\bigotimes_{\alpha\in A} \boldsymbol{K}_\alpha^m\big),\\ \boldsymbol{f}_{full} &= \sum_{m\in M_f} \big(\bigotimes_{\alpha\in A} \boldsymbol{f}_\alpha^m\big).\end{aligned}$$

For computational purposes we define the following isomorphism:

$$\begin{aligned}\phi : \mathcal{V} &\to \mathbb{R}^{(n_{DoF})} =: \mathcal{V}_{full},\\ \bigotimes_{\alpha\in A} N_{i_\alpha}^\alpha &\mapsto e_{\psi\left((i_\alpha)_{\alpha\in A}\right)}.\end{aligned}$$

$$\begin{aligned}\psi : \prod_{\alpha\in A}\{1,2,...,n_{DoF_\alpha}\} &\to \{1,2,...,n_{DoF}\},\\ \left((i_\alpha)_{\alpha\in A}\right) &\mapsto \sum_{\alpha\in A} i_\alpha \cdot \prod_{\substack{\beta\leqslant\alpha\\ \beta\neq\alpha}} n_{DoF_\beta}.\end{aligned}$$

Where $n_{DoF} = \prod_{\alpha\in A} n_{DoF_\alpha}$ and $(e_i)_{i\in\{1,...,n_{DoF}\}}$ form a basis on $\mathcal{V}_{full}$. Here $(N_{i_\alpha}^\alpha)_{\alpha\in A}$ is a tuple of arbitrary basis elements on every discretised space $\mathcal{V}_\alpha$. Note that we have introduced a total order relation in the set A of separated dimensions of the domain.

It is straightforward to obtain the representation of the matrix $\boldsymbol{K}_{full}$ in $\mathcal{V}_{full}$. That is, the map between the images of ϕ:

$$[\boldsymbol{K}_{full}^\phi]_{i,j} = \sum_{m\in M_B}\prod_{\alpha\in A}[\boldsymbol{K}_\alpha^m]_{(\psi^{-1}(i))_\alpha,(\psi^{-1}(j))_\alpha}, \quad i,j \in \{1,...,n_{DoF}\}.$$

In order to reduce the bandwidth of $\boldsymbol{K}_{full}^\phi$ the optimal way to define the order relation in A is to order the elements in reverse order of the maximum bandwidth of all the matrices. That is, if $\alpha \leqslant \beta$ then

$$\max_{m\in M_B} \text{Bandwidth}(\boldsymbol{K}_\alpha^m) \geqslant \max_{m\in M_B} \text{Bandwidth}(\boldsymbol{K}_\beta^m).$$

This implies that usually the space of spatial functions is the least element. The representation of the force vector is obvious from the definition of the isomorphism.

$$\boldsymbol{f}_{full}^\phi = \phi(\boldsymbol{f}_{full}).$$

Chapter 3

Galerkin PGD

In this chapter we introduce the formulation of the Galerkin PGD. This method is similar to the classical PGD algorithms. However, in the PGD context it is usual to treat the parametric dimensions in a non-consistent way [4]. That is, as the matrices resulting in the discretisation of the parametric dimensions are mass matrices, they are diagonalised. We are interested in non self-adjoint problems where time plays a role as a parametric dimension. The discretisation of such operators cannot be diagonalised. For this reason, we treat all dimensions in a consistent way.

The methods presented in this section are taken from [3]. There, the formulation of the Galerkin method is presented for the case of space-time separation. It is also introduced a method for updating the time dimension and both space-time dimensions after the end of each mode. Here we present the generalisation of the methods to an arbitrary number of dimensions.

3.1 Rank-one approximation

3.1.1 Abstract formulation

The Galerkin rank-one solution is defined as a tuple $(u_\alpha)_{\alpha\in A} \in \bigoplus_{\alpha\in A} \mathcal{V}_\alpha$ that approximates the solution as $u \approx u_{R1} = \bigotimes_{\alpha\in A} u_\alpha$ such that

$$B(u_{R1}, v) = L(v) \quad \forall v \in \mathcal{V}^{test} = \sum_{i\in A} \mathcal{V}_i^{test}. \tag{3.1}$$

Where

$$\mathcal{V}_i^{test} = \Big\{ v \in \mathcal{V} : v = v_i \otimes \bigotimes_{\alpha\in A\setminus\{i\}} u_\alpha,\ v_i \in \mathcal{V}_i \Big\}.$$

Using the fact that both forms are linear in v, the nonlinear condition can be enforced as a coupled system of n_D equations. The equation for the i-th dimension is

$$B(u_{R1}, v) = L(v) \quad \forall v \in \mathcal{V}_i^{test}. \tag{3.2}$$

Different methods for computing the solution $(u_\alpha)_{\alpha\in A}$ are presented in chapter 7. We briefly explain here the classical fixed-point algorithm. To start, one choses an arbitrary initial guess. Then, pick a dimension i. Considering all functions of the rest of dimensions as knowns, equation (3.2) becomes a linear equation. The solution of that linear system is used as the new guess for the function of the i dimension of the rank-one approximation. Repeating the process for all dimensions $i \in A$ the whole rank-one function is updated concluding one iteration. The iterations are repeated until the fixed point is achieved (up to a prescribed tolerance).

3.1.2 Advection-diffusion formulation

The Galerkin rank-one approximation of the transient advection-diffusion problem is to find $(\omega, \lambda) \in \mathcal{V}_S \oplus \mathcal{V}_T$ such that

$$B(\omega\lambda, \omega^*\lambda + \omega\lambda^*) = L(\omega^*\lambda + \omega\lambda^*) \quad \forall(\omega^*, \lambda^*) \in \mathcal{V}_S \oplus \mathcal{V}_T. \tag{3.3}$$

Alternatively, we rewrite the non-linear equation as a coupled system of two equations.

$$B(\omega\lambda, \omega^*\lambda) = L(\omega^*\lambda) \quad \forall\omega^* \in \mathcal{V}_S, \tag{3.4}$$
$$B(\omega\lambda, \omega\lambda^*) = L(\omega\lambda^*) \quad \forall\lambda^* \in \mathcal{V}_T. \tag{3.5}$$

3.2 Additional modes

3.2.1 Abstract formulation

To obtain a higher accuracy, the solution can be approximated as the sum of different rank-one solutions, i.e. $u \approx u_M = \sum_{m=1}^{M} u_{R1}^m$. Assume that M modes have been computed and it is desired to add an additional mode to improve the approximation. This can be stated as problem (3.1) with a modification of the linear functional L. That is, the contribution to the residual of the previous modes has to be subtracted from the functional. So the equation to solve is now

$$B(u_{R1}^{M+1}, v) = L(v) - \tilde{L}_{u_M}(v) \quad \forall v \in \mathcal{V}^{test}. \tag{3.6}$$

We assume that the map $u_M \mapsto \tilde{L}_{u_M}$ is linear and bounded so the contribution functional can be written as a bilinear continuous form: $\tilde{L}_{u_M}(v) = \tilde{B}(u_M, v)$. Note that in general $\tilde{B} = B$ but this is not be the case in the Petrov-Galerkin scheme, for instance. We may further assume that $\tilde{B}$ is separable. That is

$$\tilde{B}\Big(\bigotimes_{\alpha\in A} u_\alpha, \bigotimes_{\alpha\in A} v_\alpha\Big) = \sum_{m\in M_{\tilde{B}}} \prod_{\alpha\in A} \tilde{B}_\alpha^m(u_\alpha, v_\alpha).$$

3.2.2 Advection-diffusion formulation

For the standard PGD Galerkin formulation, the term involving the previous terms is just the same bilinear form to subtract the contribution to the residual of the already computed modes. Let $u_M = \sum_{m=1}^{M} \omega^m \lambda^m$ represent the previously computed modes. Then the new mode (ω, λ) is computed as the solution of

$$B(\omega\lambda, \omega^*\lambda + \omega\lambda^*) = L(\omega^*\lambda + \omega\lambda^*) - B(u_M, \omega^*\lambda + \omega\lambda^*) \quad \forall(\omega^*, \lambda^*) \in \mathcal{V}_S \oplus \mathcal{V}_T. \tag{3.7}$$

3.3 Update

3.3.1 Abstract formulation

The previous algorithm is not optimal in the sense that modes are computed sequentially in an uncoupled form. In this section we present a method that permits the coupling in some sense of the different modes. The procedure is the following:

First, we split the family of Hilbert spaces $\{\mathcal{V}_\alpha\}_{\alpha\in A}$ in two subfamilies $\{\mathcal{V}_\beta\}_{\beta\in B}$ and $\{\mathcal{V}_\gamma\}_{\gamma\in G}$ such that G is nonempty, $G \subseteq A$ and $B = A \backslash G$.

Then, after an additional mode has been computed, we perform an update that consists in recomputing the functions of all modes in the dimensions $\gamma \in G$ such that they satisfy a multiple Galerkin orthogonality.

Suppose that an approximation u_M is given as an approximation to the solution u. By virtue of the splitting of the dimensions in B and G we can rewrite u_M as

$$u_M = \sum_{m=1}^{M} \bigotimes_{\alpha\in A} u_\alpha^m = \sum_{m=1}^{M} \bigotimes_{\beta\in B} u_\beta^m \otimes \bigotimes_{\gamma\in G} u_\gamma^m.$$

Then the problem is: given $(u_\beta^m)_{(\beta,m)\in B\times\{1,\dots,M\}}$, find $(\bar{u}_\gamma^m)_{(\gamma,m)\in G\times\{1,\dots,M\}}$ such that

$$B(\bar{u}_M, v) = L(v) \quad \forall v \in \sum_{\substack{i\in G \\ m\in\{1,\dots,M\}}} \mathcal{V}_{i,m}^{test}. \tag{3.8}$$

Where

$$\bar{u}_M = \sum_{m=1}^{M} \bigotimes_{\beta\in B} u_\beta^m \otimes \bigotimes_{\gamma\in G} \bar{u}_\gamma^m,$$
$$\mathcal{V}_{i,m}^{test} = \Big\{ v \in \mathcal{V} : v = v_i^m \otimes \bigotimes_{\beta\in B} u_\beta^m \otimes \bigotimes_{\gamma\in G\setminus\{i\}} \bar{u}_\gamma^m,\ v_i^m \in \mathcal{V}_i \Big\}.$$

This nonlinear problem can be reformulated as a coupled system of $|G|\cdot M$ linear equations in a similar way that for the Galerkin method. With this, the equation corresponding to the i-th dimension and the m mode is: given $\{u_\gamma^n\}_{(\gamma,n)\in(G\times(1,\dots,M))\setminus\{i,m\}}$, find $u_i^m \in \mathcal{V}_i$ such that

$$B(\bar{u}_M, v) = L(v) \quad \forall \mathcal{V}_{i,m}^{test}. \tag{3.9}$$

In order to reduce the number of equations of this system an alternative approach is used to obtain a system of $|G|$ equations. However, the spaces of each of these equations is enlarged by a factor M. To do so, we shall first define such enlarged spaces:

$$\bar{\mathcal{V}}_\gamma = (\mathcal{V}_\gamma)^M,\ \gamma \in G, \tag{3.10}$$
$$\bar{\mathcal{V}} = \bigotimes_{\gamma\in G} \bar{\mathcal{V}}_\gamma. \tag{3.11}$$

We also define the projection operator $\bar{P}_\gamma^m : \bar{\mathcal{V}}_\gamma \to \mathcal{V}_\gamma$ as the map returning the m component of an element in $\bar{\mathcal{V}}_\gamma$.

Defining the adequate separable bilinear form $\bar{B}$ and linear functional $\bar{L}$ on $\bar{\mathcal{V}}$ we can reformulate the problem such that it adopts the form of a Galerkin PGD formulation. The problem consists in finding $(\bar{u}_\gamma)_{\gamma\in G} \in \bigoplus_{\gamma\in G} \bar{\mathcal{V}}_\gamma$ such that

$$\bar{B}(\bar{u}^G, \bar{v}) = \bar{L}(\bar{v}) \quad \forall \bar{v} \in \bar{\mathcal{V}}^{test} = \sum_{i\in G} \bar{\mathcal{V}}_i^{test}. \tag{3.12}$$

Where

$$\bar{u}^G = \bigotimes_{\gamma\in G} \bar{u}_\gamma,$$
$$\bar{\mathcal{V}}_i^{test} = \Big\{ \bar{v} \in \bar{\mathcal{V}} : \bar{v} = \bar{v}_i \otimes \bigotimes_{\gamma\in G\setminus\{i\}} \bar{u}_\gamma,\ \bar{v}_i \in \bar{\mathcal{V}}_i \Big\}.$$

Then, the problem is solved by assigning to $\bar{u}_\gamma^m$ the element $\bar{P}_\gamma^m \bar{u}_\gamma$.

The forms that ensure the equivalence of the problem are

$$\bar{B}\Big(\bigotimes_{\gamma\in G} \bar{u}_\gamma, \bigotimes_{\gamma\in G} \bar{v}_\gamma \Big) = \sum_{m_u,m_v=1}^{M} \sum_{m_B\in M_B} \prod_{\beta\in B} \sigma_{\beta,m_B}^{m_u,m_v} \cdot \prod_{\gamma\in G} \bar{B}_{\gamma,m_B}^{m_u,m_v}(\bar{u}_\gamma, \bar{v}_\gamma), \tag{3.13}$$
$$\bar{L}\Big(\bigotimes_{\gamma\in G} \bar{v}_\gamma \Big) = \sum_{m_v=1}^{M} \sum_{m_L\in M_L} \prod_{\beta\in B} \sigma_{\beta,m_L}^{m_v} \cdot \prod_{\gamma\in G} \bar{L}_{\gamma,m_L}^{m_v}(\bar{v}_\gamma). \tag{3.14}$$

Where

$$\sigma_{\beta,m_B}^{m_u,m_v} = B_\beta^{m_B}(u_\beta^{m_u}, u_\beta^{m_v}), \tag{3.15}$$
$$\sigma_{\beta,m_L}^{m_v} = L_\beta^{m_L}(u_\beta^{m_v}), \tag{3.16}$$
$$\bar{B}_{\gamma,m_B}^{m_u,m_v}(\bar{u}_\gamma, \bar{u}_\gamma) = B_\gamma^{m_B}(\bar{P}_\gamma^{m_u}\bar{u}_\gamma, \bar{P}_\gamma^{m_v}\bar{v}_\gamma), \tag{3.17}$$
$$\bar{L}_{\gamma,m_L}^{m_v}(\bar{v}_\gamma) = L_\gamma^{m_L}(\bar{P}_\gamma^{m_v}\bar{v}_\gamma). \tag{3.18}$$

Forms B and L have not been defined on the whole space $\bar{\mathcal{V}}$ but only on a subset of it. We complete their definition using linearity and density arguments, i.e. extending them such that they preserve linearity and continuity. The problem to solve is to find the rank-one Galerkin approximation $\bar{u}^G$ where the number of dimensions has been reduced to $|G|$ although each dimension size has been multiplied by M. Note also that the number of modes of B and L have been multiplied by M^2 and M.

Now we prove that solving problem (3.8) is equivalent to solving (3.12). Equation (3.12) can be stated as

$$\bar{B}\Big(\bigotimes_{\gamma\in G}\bar{u}_\gamma, \sum_{i\in G}\bar{v}_i\otimes\bigotimes_{\gamma\in G\setminus\{i\}}\bar{u}_\gamma\Big) = \bar{L}\Big(\sum_{i\in G}\bar{v}_i\otimes\bigotimes_{\gamma\in G\setminus\{i\}}\bar{u}_\gamma\Big) \quad \forall(\bar{v}_i)_{i\in G}\in\bigoplus_{i\in G}\bar{\mathcal{V}}_i.$$

Using the definitions of $\bar{B}$ and $\bar{L}$ one obtains

$$\sum_{m_u,m_v=1}^{M}\sum_{m_B\in M_B}\prod_{\beta\in B}B_\beta^{m_B}(u_\beta^{m_u},u_\beta^{m_v})\cdot\left[\sum_{i\in G}B_i^{m_B}(\bar{u}_i^{m_u},v_i^{m_v})\cdot\prod_{\gamma\in G\setminus\{i\}}B_\gamma^{m_B}(\bar{u}_\gamma^{m_u},\bar{u}_\gamma^{m_v})\right]=$$
$$\sum_{m_v=1}^{M}\sum_{m_L\in M_L}\prod_{\beta\in B}L_\beta^{m_L}(u_\beta^{m_v})\cdot\left[\sum_{i\in G}L_i^{m_L}(v_i^{m_v})\cdot\prod_{\gamma\in G\setminus\{i\}}L_\gamma^{m_L}(\bar{u}_\gamma^{m_u})\right]$$
$$\forall((v_i^{m_v})_{i\in G})_{m_v\in\{1,\dots,M\}}\in\Big(\bigoplus_{i\in G}\mathcal{V}_i\Big)^M.$$

Using the separability properties of B and L some terms can be grouped:

$$\sum_{m_u,m_v=1}^{M}B\Big(\bigotimes_{\beta\in B}u_\beta^{m_u}\otimes\bigotimes_{\gamma\in G}\bar{u}_\gamma^{m_u},\sum_{i\in G}v_i^{m_v}\otimes\bigotimes_{\beta\in B}u_\beta^{m_v}\otimes\bigotimes_{\gamma\in G\setminus\{i\}}\bar{u}_\gamma^{m_v}\Big)=$$
$$\sum_{m_v=1}^{M}L\Big(\sum_{i\in G}v_i^{m_v}\otimes\bigotimes_{\beta\in B}u_\beta^{m_v}\otimes\bigotimes_{\gamma\in G\setminus\{i\}}\bar{u}_\gamma^{m_v}\Big)\quad\forall((v_i^{m_v})_{i\in G})_{m_v\in\{1,\dots,M\}}\in\Big(\bigoplus_{i\in G}\mathcal{V}_i\Big)^M.$$

We conclude the proof by noting that this last equation is equivalent to (3.8).

Summarising, to perform the update we define a new system represented as a Galerkin PGD. If M modes have been computed, to compute the updated functions $(\bar{u}_\gamma^m)$, one defines a new separated linear map $\bar{\boldsymbol{K}}$. This enlarged separated map is a block matrix $M\times M$ for each dimension and mode. It has M^2 modes for every mode of the original PGD bilinear form. Every of such modes consists in one matrix for each dimension to be updated where one of the components of the block matrix (the same for every dimension) is the original PGD matrix and the rest are 0. Let denote the row and column of that component by i and j. Every mode is multiplied by a constant that consists in the product over all the dimensions not to be updated of the quadratic form $(\boldsymbol{u}_\beta^i)^T\boldsymbol{K}_\beta^{m_B}\boldsymbol{u}_\beta^j$.

To define the separated vector $\bar{\boldsymbol{f}}$ the process is similar. It consists of block column vectors with size M for each dimension and mode. It has M modes for every mode of the original PGD vector. Every mode consists again in a vector for each dimension where one component (say i) is the original PGD vector and the rest are 0. Every mode is scaled by $(\boldsymbol{u}_\beta^i)^T\boldsymbol{f}_\beta^{m_L}$.

The solution is computed as a Galerkin PGD solution of the system $\bar{\boldsymbol{K}}\bar{\boldsymbol{u}}=\bar{\boldsymbol{f}}$. Finally, every function u_γ^m is updated to the m component of $\bar{\boldsymbol{u}}_\gamma$.

To conclude we note two points concerning two extreme cases. On the one hand, if the update is to be performed only in one dimension, i.e. $|G|=1$, the resulting system is a PGD system involving only one dimension. That is, a conventional linear system of equations. On the other hand, if the update is to be performed in all dimensions, i.e. $G=A$, the method can be understood as a kind of a priori Proper Orthogonal Decomposition (POD). In this case, it makes no sense computing the solution by a sequential addition of modes as in the update step all functions are discarded and the update of all of them is computed.

3.3.2 Advection-diffusion formulation

Here we present the formulation of the problem of updating the time functions of the advection-diffusion problem. We reformulate the problem as a Galerkin one.

Suppose that after each computation of a new mode, we want to recompute all temporal functions of the solution while keeping the spatial ones. That is, we want to improve the accuracy of the solution enabling the modes to be coupled between them but without solving any problem involving the spatial dimension as it is more expensive. We use the following Galerkin formulation: find $(\lambda^m)_{m\in\{1,\dots,M\}} \in (\mathcal{V}_T)^M$ such that

$$B\left(\sum_{m=1}^{M}\omega^m\lambda^m, \sum_{m=1}^{M}\omega^m\lambda^{m*}\right) = L\left(\sum_{m=1}^{M}\omega^m\lambda^{m*}\right) \quad \forall(\lambda^{m*})_{m\in\{1,\dots,M\}} \in (\mathcal{V}_T)^M. \tag{3.19}$$

As the update is performed only in one dimension the resulting system is not of PGD type but a linear conventional Galerkin formulation. Here, for simplicity, we shall restrict ourselves to the case where $M = 2$ (see section 3.3.1 for the general case). That is, given $(\omega^1, \omega^2) \in (\mathcal{V}_S)^2$ find $(\lambda^1, \lambda^2) \in (\mathcal{V}_T)^2$ such that

$$B\big(\omega^1\lambda^1 + \omega^2\lambda^2, \omega^1\lambda^{1*} + \omega^2\lambda^{2*}\big) = L\big(\omega^1\lambda^{1*} + \omega^2\lambda^{2*}\big) \quad \forall(\lambda^{1*}, \lambda^{2*}) \in (\mathcal{V}_T)^2.$$

We expand the forms as follows:

$$\begin{aligned} B\big(\omega^1\lambda^1, \omega^1\lambda^{1*}\big) + B\big(\omega^2\lambda^2, \omega^1\lambda^{1*}\big) + B\big(\omega^1\lambda^1, \omega^2\lambda^{2*}\big) + B\big(\omega^2\lambda^2, \omega^2\lambda^{2*}\big) =& \\ L\big(\omega^1\lambda^{1*}\big) + L\big(\omega^2\lambda^{2*}\big) \quad \forall(\lambda^{1*}, \lambda^{2*}) \in (\mathcal{V}_T)^2.& \end{aligned}$$

We reformulate the previous equation as a coupled system of two equations:

$$\begin{aligned} B\big(\omega^1\lambda^1, \omega^1\lambda^{1*}\big) + B\big(\omega^2\lambda^2, \omega^1\lambda^{1*}\big) &= L\big(\omega^1\lambda^{1*}\big) \quad \forall\lambda^{1*} \in \mathcal{V}_T, \\ B\big(\omega^1\lambda^1, \omega^2\lambda^{2*}\big) + B\big(\omega^2\lambda^2, \omega^2\lambda^{2*}\big) &= L\big(\omega^2\lambda^{2*}\big) \quad \forall\lambda^{2*} \in \mathcal{V}_T. \end{aligned}$$

Using the separability properties:

$$\begin{aligned} \sum_{m\in\{e,d,a,0\}} \big(B_S^m(\omega^1,\omega^1)B_T^m(\lambda^1,\lambda^{1*}) + B_S^m(\omega^2,\omega^1)B_T^m(\lambda^2,\lambda^{1*})\big) &= \sum_{m\in\{f,0\}} L_S^m(\omega^1)L_T^m(\lambda^{1*}) \quad \forall\lambda^{1*} \in \mathcal{V}_T, \\ \sum_{m\in\{e,d,a,0\}} \big(B_S^m(\omega^1,\omega^2)B_T^m(\lambda^1,\lambda^{2*}) + B_S^m(\omega^2,\omega^2)B_T^m(\lambda^2,\lambda^{2*})\big) &= \sum_{m\in\{f,0\}} L_S^m(\omega^2)L_T^m(\lambda^{2*}) \quad \forall\lambda^{2*} \in \mathcal{V}_T. \end{aligned}$$

As ω^m are given, one can compute a priori the forms associated to them. That is, $\sigma_{S,m}^{i,j} = B_S^m(\omega^j, \omega^i),\ i,j \in \{1,2\}, m \in \{e,d,a,0\}$ and $\sigma_{S,m}^{i} = L_S^m(\omega_i)\ i \in \{1,2\}, m \in \{f,0\}$. This leads to

$$\begin{aligned} \sum_{m\in\{e,d,a,0\}} \big(\sigma_{S,m}^{1,1}B_T^m(\lambda^1,\lambda^{1*}) + \sigma_{S,m}^{1,2}B_T^m(\lambda^2,\lambda^{1*})\big) &= \sum_{m\in\{f,0\}} \sigma_{S,m}^{1}L_T^m(\lambda^{1*}) \quad \forall\lambda^{1*} \in \mathcal{V}_T, \\ \sum_{m\in\{e,d,a,0\}} \big(\sigma_{S,m}^{2,1}B_T^m(\lambda^1,\lambda^{2*}) + \sigma_{S,m}^{2,2}B_T^m(\lambda^2,\lambda^{2*})\big) &= \sum_{m\in\{f,0\}} \sigma_{S,m}^{2}L_T^m(\lambda^{2*}) \quad \forall\lambda^{2*} \in \mathcal{V}_T. \end{aligned}$$

To solve this system we first define the enlarged vector of unknowns

$$\lambda^1 = \sum_{i=1}^{n_{DoF_T}} N_i^T \boldsymbol{\lambda^1}_i, \quad \lambda^2 = \sum_{i=1}^{n_{DoF_T}} N_i^T \boldsymbol{\lambda^2}_i, \quad \bar{\boldsymbol{\lambda}} = \begin{pmatrix}\boldsymbol{\lambda^1}\\ \boldsymbol{\lambda^2}\end{pmatrix}.$$

The projection operators are defined as

$$\begin{aligned} \bar{P}_T^i : \mathcal{V}_T \oplus \mathcal{V}_T &\to \mathcal{V}_T, \quad i \in \{1,2\} \\ (\lambda^1, \lambda^2) &\mapsto \lambda^i. \end{aligned}$$

To solve it as an enlarged problem, we define the enlarged forms as

$$\begin{aligned}
\bar{B}(\bar{\lambda}, \bar{\lambda}^*) &= \sum_{\substack{m\in\{e,d,a,0\}\\ i,j\in\{1,2\}}} \sigma_{i,j}^{S,m} \bar{B}_{T,m}^{i,j}(\bar{\lambda}, \bar{\lambda}^*),\\
\bar{L}(\bar{\lambda}^*) &= \sum_{\substack{m\in\{f,0\}\\ i\in\{1,2\}}} \sigma_{i}^{S,m} \bar{L}_{T,m}^{i}(\bar{\lambda}^*).
\end{aligned}$$

Where

$$\begin{aligned}
\bar{B}_{T,m}^{i,j}(\bar{\lambda}, \bar{\lambda}^*) &= B_T^m(\bar{P}_T^j \bar{\lambda}, \bar{P}_T^i \bar{\lambda}^*),\\
\bar{L}_{T,m}^{i}(\bar{\lambda}^*) &= L_T^m(\bar{P}_T^i \bar{\lambda}^*).
\end{aligned}$$

The associated matrices and vectors are:

$$\begin{aligned}
\left[\bar{\boldsymbol{K}}_m^{i,j}\right]_{ab} &= B_T^m(\bar{P}_T^j \bar{N}_b^T, \bar{P}_T^i \bar{N}_a^T), \quad m \in \{e,d,a,0\},\ i,j \in \{1,2\},\\
\bar{\boldsymbol{K}}_m^{1,1} &= \begin{pmatrix} \boldsymbol{K}_T^m & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{pmatrix}, \qquad \bar{\boldsymbol{K}}_m^{1,2} = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{K}_T^m \\ \boldsymbol{0} & \boldsymbol{0} \end{pmatrix},\\
\bar{\boldsymbol{K}}_m^{2,1} &= \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{K}_T^m & \boldsymbol{0} \end{pmatrix}, \qquad \bar{\boldsymbol{K}}_m^{2,2} = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{K}_T^m \end{pmatrix},\\
\left[\bar{\boldsymbol{L}}_m^{i}\right]_{a} &= L_T^m(\bar{P}_T^i \bar{N}_a), \quad m \in \{f,0\},\ i \in \{1,2\},\\
\bar{\boldsymbol{L}}_m^{1} &= \begin{pmatrix} \boldsymbol{f}_T^m \\ 0 \end{pmatrix}, \qquad \bar{\boldsymbol{L}}_m^{2} = \begin{pmatrix} 0 \\ \boldsymbol{f}_T^m \end{pmatrix}.
\end{aligned}$$

The weights σ of each mode are:

$$\begin{aligned}
\sigma_m^{i,j} &= B_S^m(\omega^j, \omega^i), \quad m \in \{e,d,a,0\},\ i,j \in \{1,2\},\\
\sigma_m^{1,1} &= (\boldsymbol{\omega}^1)^T \boldsymbol{K}_S^m \boldsymbol{\omega}^1, \qquad \sigma_m^{1,2} = (\boldsymbol{\omega}^1)^T \boldsymbol{K}_S^m \boldsymbol{\omega}^2,\\
\sigma_m^{2,1} &= (\boldsymbol{\omega}^2)^T \boldsymbol{K}_S^m \boldsymbol{\omega}^1, \qquad \sigma_m^{2,2} = (\boldsymbol{\omega}^2)^T \boldsymbol{K}_S^m \boldsymbol{\omega}^2,\\
\sigma_m^{i} &= L_S^m(\omega^i), \quad m \in \{f,0\},\ i,j \in \{1,2\},\\
\sigma_m^{1} &= (\boldsymbol{\omega}^1)^T \boldsymbol{f}_S^m, \qquad \sigma_m^{2} = (\boldsymbol{\omega}^2)^T \boldsymbol{f}_S^m.
\end{aligned}$$

3.4 Numerical example: First approach and update

In the first chapters we restrict the study of two examples based on the same problem: one of pure advection type and the other with the presence of diffusion. On later chapters we introduce additional examples.

Such first examples are similar to one presented by Nouy in his paper. The spatial discretisation consists of a regular mesh of 40×40 triangular linear elements while the temporal discretisation consists of a regular mesh of 100 piecewise linear elements. This discretisation corresponds to $n_{DoF_S} = 1521$ and $n_{DoF_T} = 101$.

The force term is $f = 0$, the velocity field is $c(x, y, t) = \pi(-y + \frac{1}{2}, x - \frac{1}{2})$ and the initial condition is

$$u_0(x, y) = \exp\left(- \frac{(x - \frac{2}{3})^2 + (y - \frac{1}{2})^2}{0.07^2} \right).$$

For the diffusivity constant two different values have been chosen: $\nu_A = 0$ for the pure advection problem and $\nu_{AD} = 10^{-3}$ for the advection-diffusion problem.

Due to the chosen parameters, the solution does not present boundary layers. For this reason, the weak form of the spatial dimension has not been stabilised. In general, for advection dominated problems, some spatial stabilisation —e.g. SUPG, GLS, SGS— is needed as in classical advection-diffusion solvers [1].

In contrast, temporal stabilisation is not needed as the PGD methods inherit the stability property of the continuous Galerkin discretisation of the full problem. For a discussion on the stabilisation of the weak formulation, see chapter 5.

3.4.1 Reference solution

To obtain the reference solution we have used the procedure explained in section 2.3. The reference solution of both problems is plotted below.

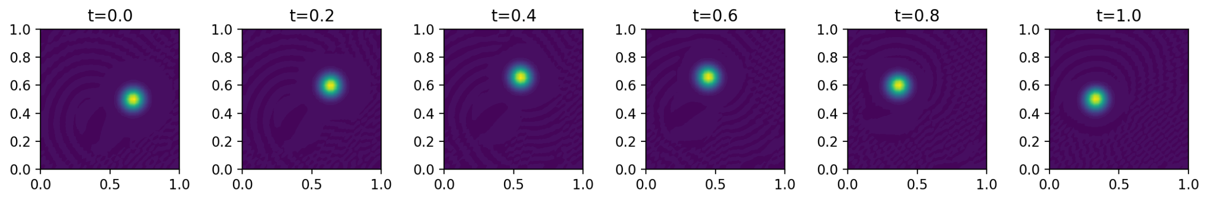

Figure 3.1 Reference solution for pure advection

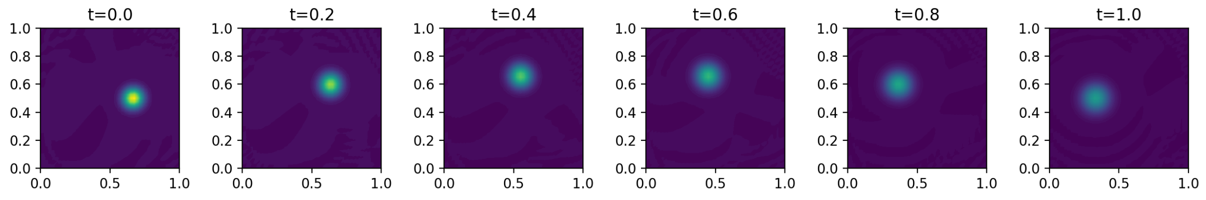

Figure 3.2 Reference solution for advection-diffusion

3.4.2 Numerical results

Now we present the results for the Galerkin PGD algorithm approximating both reference solutions. In the figures we plot, in that order, the first ten spatial modes, the first ten temporal modes and the reconstruction of the solution adding those ten modes.

Galerkin without update

First, we present the results for the plain Galerkin algorithm. That is, without performing any update.

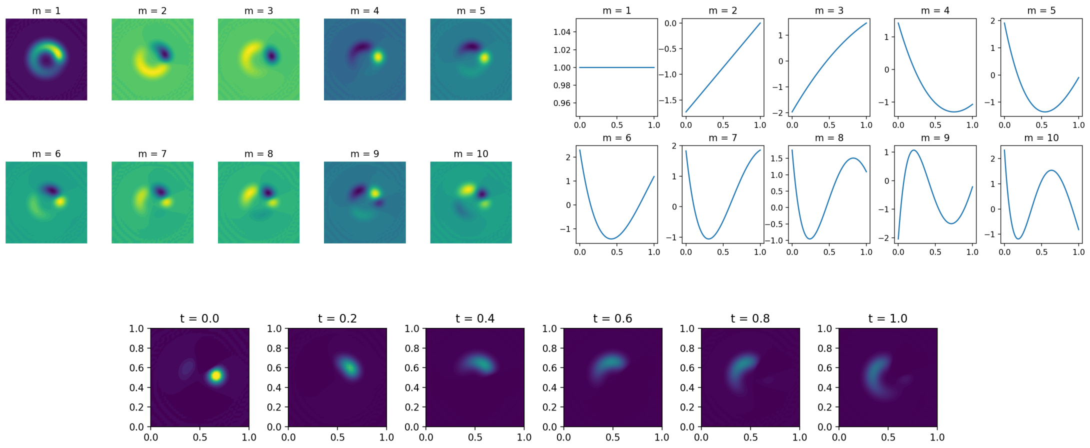

Figure 3.3 Galerkin PGD approximation for the pure advection problem

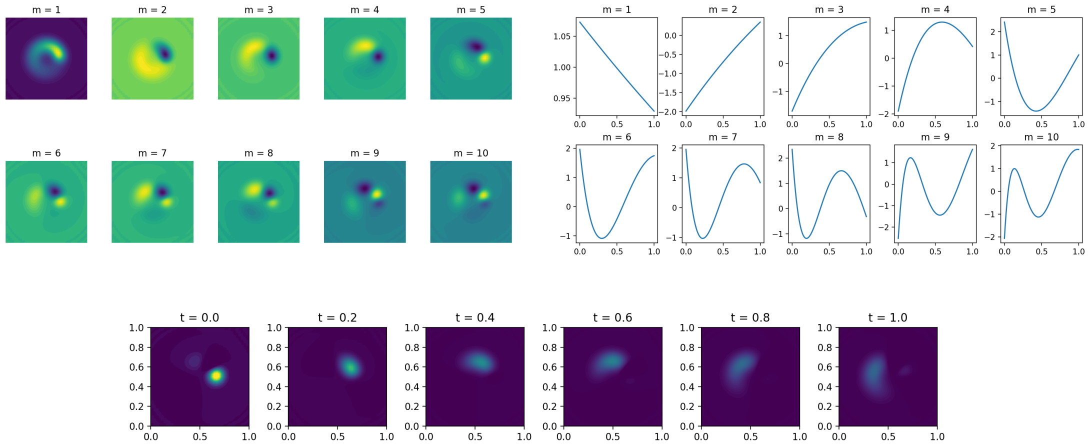

Figure 3.4 Galerkin PGD approximation for the advection-diffusion problem

It is seen by simple inspection that the approximation of the solution is relatively accurate for the first instants of time, but then it presents larger errors. It can also be seen the obtained modes are far from being orthogonal. In fact, most of consecutive modes present similar spatial and temporal solutions (note that any mode can be multiplied by factor of -1 in both spatial and temporal dimension resulting in an equivalent normalised mode). This reveals that the solution is not optimal as it is known that the modes of the optimal decomposition of a second order tensor are orthogonal.

Galerkin with time update

Now we present the results with the update algorithm in the temporal dimension.

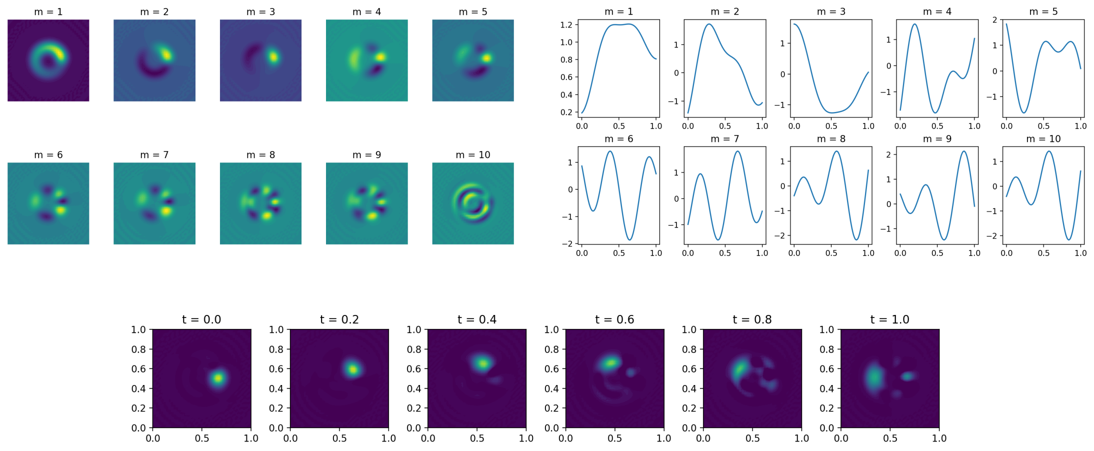

Figure 3.5 Galerkin PGD approximation with update in time for the pure advection problem

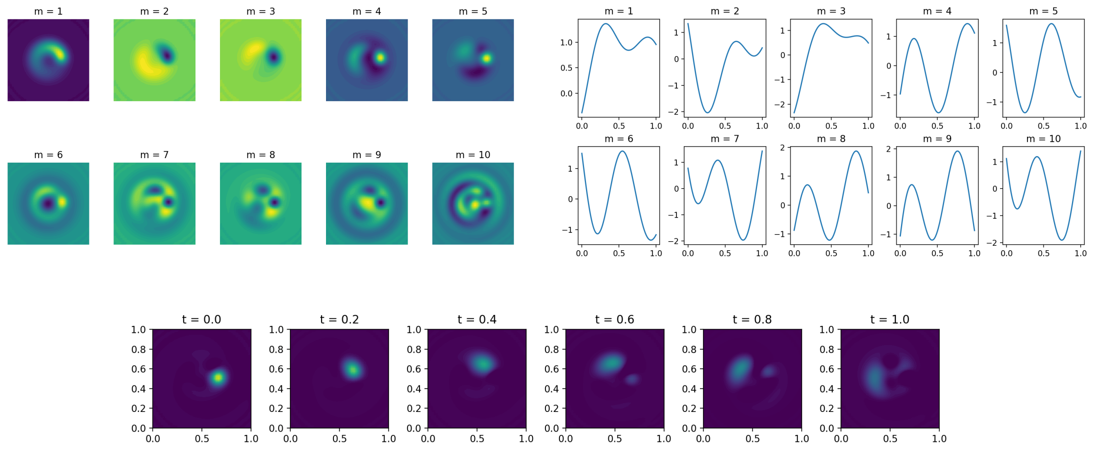

Figure 3.6 Galerkin PGD approximation with update in time for the advection-diffusion problem

We see, again, that the modes are not optimal. The error presents the same property than in the standard Galerkin algorithm: it increases as time advances.

Galerkin with space-time update

For the sake of completeness, we have computed the solution of the Galerkin PGD problem with space-time update. However, this method is highly unefficient because the computational power nedeed is very high.

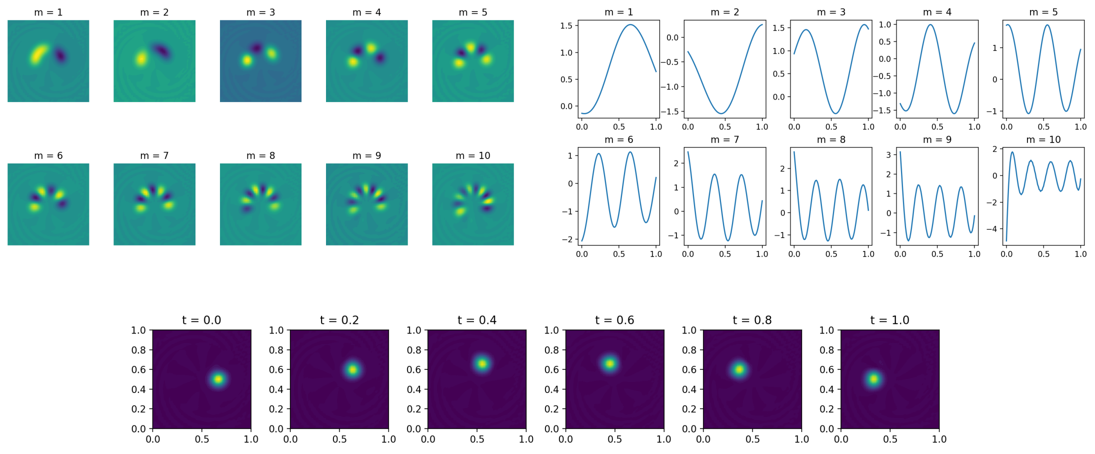

Figure 3.7 Galerkin PGD approximation with update in space and time for the pure advection problem

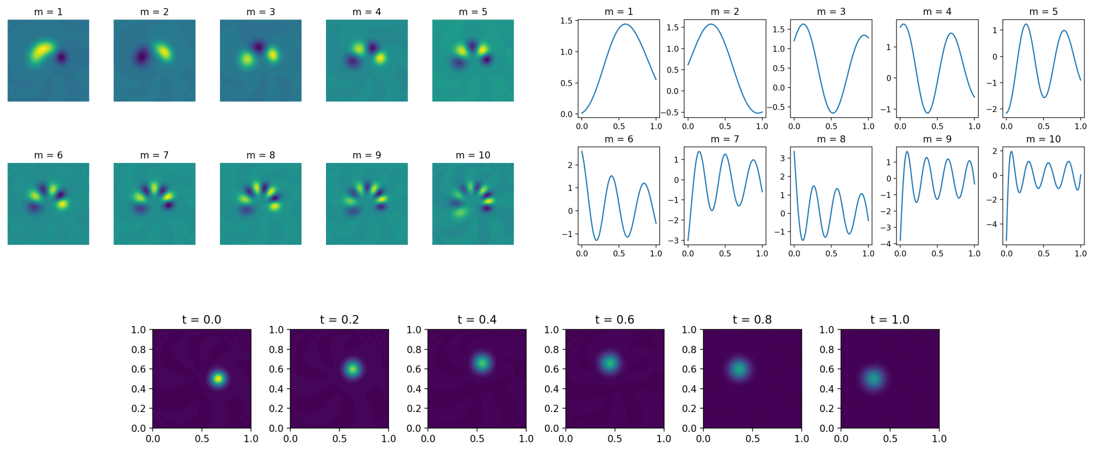

Figure 3.8 Galerkin PGD approximation with update in space and time for the advection-diffusion problem

The solutions obtained with this method are identical, by simple inspection, to the reference solutions. The modes seem to be orthogonal, as in the optimal decomposition.

3.4.3 Convergence

Now we present the convergence properties of the different methods to make quantitative assertions about the previous qualitative statements. For that, it has been computed the approximation of the solution of both problems using the three methods already presented. For the plain Galerkin, the solution has been computed up to 640 modes; for the Galerkin with time update, up to 60 modes and for the Galerkin with space and time update, up to 20 due to the high computational cost of this method. In figure 3.9, the convergence of the error of each method is shown. All errors are normalised by the norm of the reference solution such that the plots show the relative error.

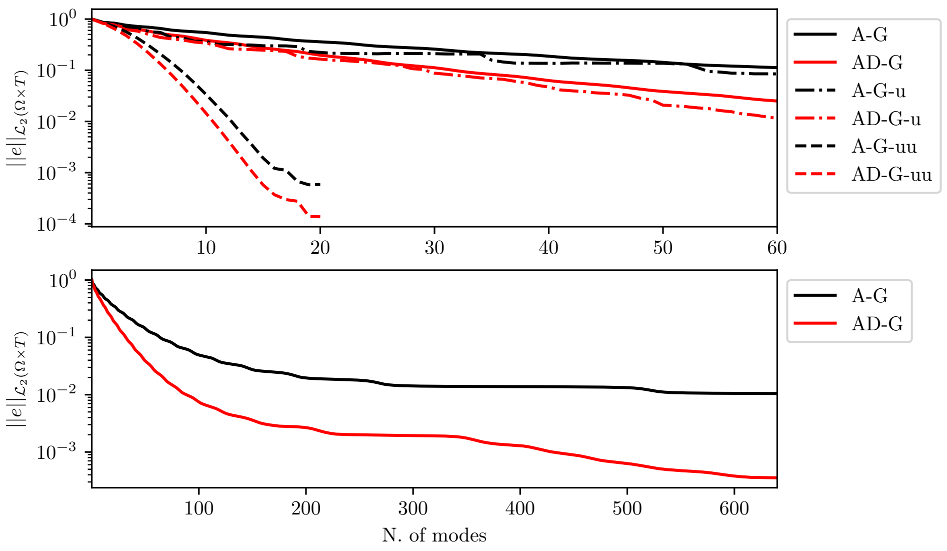

Figure 3.9 Error convergence

Here A stands for advection problem and AD for advection-diffusion problem, G for Galerkin method, u for time update and uu for space-time update.

The error of the plain Galerkin is monotonically decreasing but the order at which decreases is reduced as the number of modes increases, this is typical of the Galerkin PGD methods. Note that the error of the plain Galerkin and Galerkin with time update methods are similar; the update in time does not produce any significant improvement in the solution. We see, as expected, that the solution of the problem using the space-time update is much more accurate compared with the previous ones.

One can also note that the approximation to the advection-diffusion problem is more accurate than the approximation to the pure advection problem. The reason of this is that as the solution diffuses, the solution can be represented with less modes to the same degree of accuracy.

We focus now on the observed fact that the error of the solution increases as the time evolve. We show the evolution of the error $\|e(t)\|_{\mathcal{L}(\Omega)}$ for different methods and number of modes. We show only the solution for the pure advection problem. For the advection-diffusion problem the results are similar.

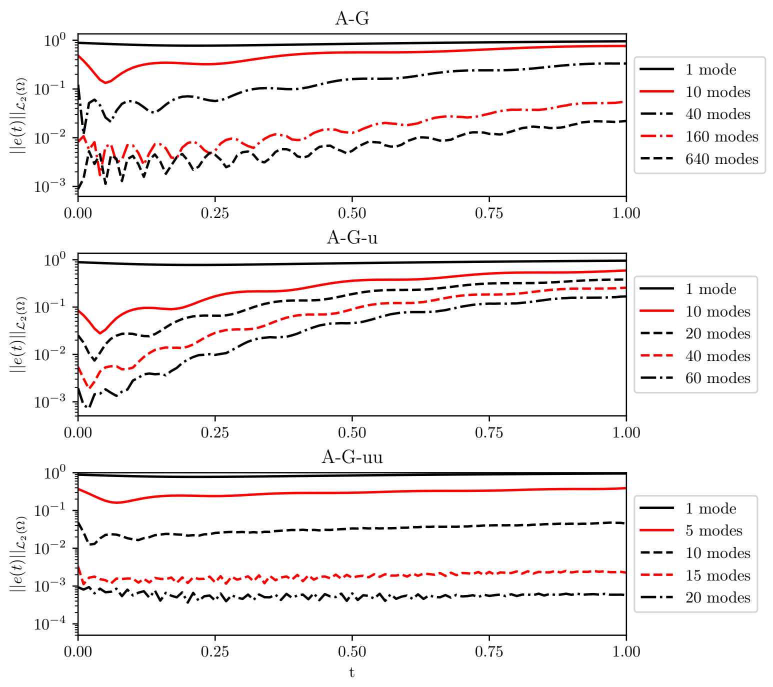

Figure 3.10 Temporal evolution of the error

We see that effectively for the plain Galerkin and time update methods the solution is more accurate at $t = 0$ and it increases roughly exponentially as time advances. One would expect that the time update would made the error more uniform in time. Surprisingly, it is seen that the opposite occurs. Take for example the solution with 40 modes. We have just seen that the error in space time for both methods is similar. The error at the instant $t = 0$ is an order of magnitude lower for the time update method compared to the plain Galerkin while at $t = 1$ is of the same order of magnitude. Conversely, for the space-time update method the error in time is constant.

In section 3.2 we have shown the procedure of sequential addition of modes to the approximation to reduce the error. The typical error estimator for the plain PGD algorithm is the norm of the last computed mode. For the algorithms that use the update procedure, as they recompute all previous modes, we have defined the error estimator as the minimum norm over all modes. Note that if we use the same error estimator for the plain method, the stopping criterion is unchanged. Now we show this error estimator for the two problems and three methods. We also show how precise is such estimator. That is, we show the ratio between the actual error and the error estimator.

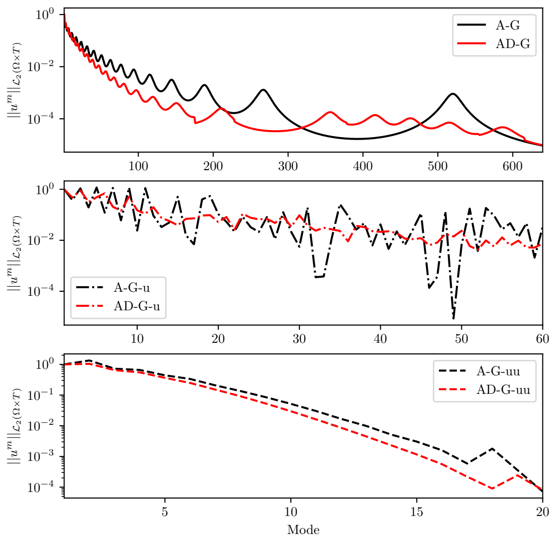

Figure 3.11 Modal amplitude

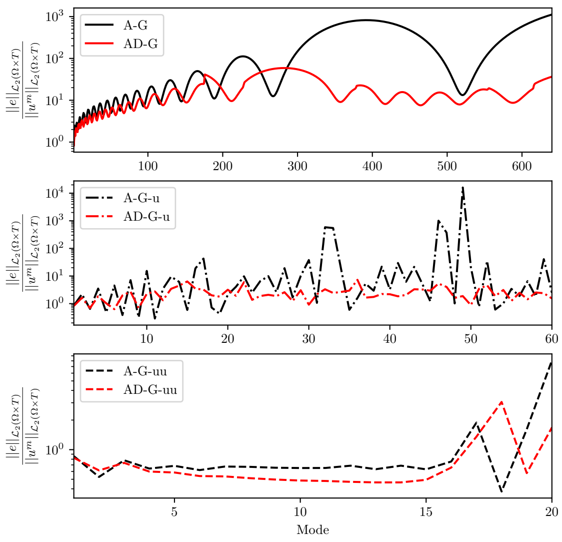

Figure 3.12 Error estimator accuracy

We see that for the plain Galerkin method and the method with time update the error estimator is not precise. The ratio is of order $10^3 - 10^4$. For the Galerkin with time-update the error is of order 1.

Chapter 4

Petrov-Galerkin PGD

In this chapter we introduce the formulation of the Petrov-Galerkin PGD method introduced by Nouy. This method consists in replacing the test functions of the Galerkin PGD problem by some different functions defined in the same space $\mathcal{V}$ in order to try to minimise the error with respect to an a priory defined inner product. In his paper, the method is introduced as a dual variation problem. We shall interpret the problem in a different way. This different derivation of the method is not only more intuitive but is also used for introducing a SUPG stabilisation in chapter 5. In the whole chapter we assume for simplicity that the solution of the proposed method is unique. In future examples we show that this might not be the case (cf. chapter 6).

4.1 Motivation and rank one approximation

In the case that the bilinear form B defines an inner product on $\mathcal{V}$, i.e. it is coercive and symmetric, it can be shown that the solution of the Galerkin rank-one problem is the one that minimises the error respect to this inner product. That is

$$u_{R1} = \underset{u_{R1}=\bigotimes_{\alpha\in A} u_\alpha}{\arg\min} \; B(u-u_{R1}, u-u_{R1}). \tag{4.1}$$

If B is non-symmetric this property does not hold anymore and the Galerkin method presents a slow convergence or, in some cases, it does not converge.

To overcome this problem, the Petrov-Galerkin approach tries to minimise the error respect to an a priori defined inner product. We require the inner product to have the following separability condition:

$$\left\langle \bigotimes_{\alpha\in A} u_\alpha, \bigotimes_{\alpha\in A} v_\alpha \right\rangle = \prod_{\alpha\in A} \langle u_\alpha, v_\alpha\rangle_\alpha.$$

Where $\langle\cdot,\cdot\rangle_\alpha$ is some valid inner product in $\mathcal{V}_\alpha$. Note that the natural inner product of the tensor product space $\mathcal{V}$ satisfies the previous property.

Now it is defined a dual element $\tilde{u}$ such that

$$B(v,\tilde{u}) = \langle u, v\rangle \quad \forall v \in \mathcal{V}. \tag{4.2}$$

The previous problem is assumed to be well-posed. We want now to find u_{R1} such that

$$u_{R1} = \underset{u_{R1}=\bigotimes_{\alpha\in A} u_\alpha}{\arg\min} \; \frac{1}{2}\langle u-u_{R1}, u-u_{R1}\rangle = \underset{u_{R1}=\bigotimes_{\alpha\in A} u_\alpha}{\arg\min} \; \frac{1}{2}\langle u_{R1}, u_{R1}\rangle - \langle u, u_{R1}\rangle. \tag{4.3}$$

Using the definition of the dual problem (4.2) (substituting v by u_{R1}), the last term of the right-hand side can be replaced and the problem now is

$$u_{R1} = \underset{u_{R1}=\bigotimes_{\alpha\in A} u_\alpha}{\arg\min} \; \frac{1}{2}\langle u_{R1}, u_{R1}\rangle - B(u_{R1},\tilde{u}). \tag{4.4}$$

Imposing the stationarity of the functional, the equation to solve is

$$\langle u_{R1}, v\rangle = B(v, \tilde{u}) \quad \forall v \in \mathcal{V}^{test} = \Big\{ v \in \mathcal{V} : v = \sum_{\alpha\in A} v_\alpha \bigotimes_{\beta\in A\backslash\{\alpha\}} u_\beta,\ v_\alpha \in \mathcal{V}_\alpha \Big\}. \tag{4.5}$$

Here it is assumed that $\tilde{u}$ is known so $v \mapsto B(v, \tilde{u})$ is a continuous linear functional on $\mathcal{V}$.

As the problem implies computing $\tilde{u}$ (and so u), we adopt an alternative approach. That is, instead of computing $\tilde{u}$ we compute $(\tilde{u}_\alpha)_{\alpha\in A} \in \bigoplus_{\alpha\in A} \mathcal{V}_\alpha$ and we approximate $\tilde{u} \approx \tilde{u}_{R1} = \bigotimes_{\alpha\in A} \tilde{u}_\alpha$. To do that, a similar approach is followed. First, an additional inner product is defined as

$$\langle u, v\rangle_\sim = \langle b^* u, b^* v\rangle = \langle bb^* u, v\rangle.$$

Where $b : \mathcal{V} \to \mathcal{V}$ is the operator associated with B by Riesz representation. Note that this is a valid inner product as, by the assumption of well posedness, the nullspace of b is $\{0\}$.

Now a bi-dual element $\tilde{\tilde{u}}$ is defined similarly to $\tilde{u}$:

$$B(\tilde{\tilde{u}}, v) = \langle \tilde{u}, v\rangle_\sim \quad \forall v \in \mathcal{V}. \tag{4.6}$$

We show that, in fact $\tilde{\tilde{u}} = u$. For that, let the element $l \in \mathcal{V}$ be the one associated to L by Riesz representation (using the $\langle \cdot, \cdot\rangle$ inner product). By (2.1) and (4.2) we obtain $bu = l$ and $b^*\tilde{u} = u$. The bi-dual problem is

$$B(\tilde{\tilde{u}}, v) = \langle \tilde{u}, v\rangle_\sim = \langle bb^*\tilde{u}, v\rangle = \langle bu, v\rangle = \langle l, v\rangle = L(v) \quad \forall v \in \mathcal{V}.$$

And by the assumption of uniqueness of the solution $\tilde{\tilde{u}} = u$.

Similarly on what we did for the primal variable, now we define the approximation to the dual solution as the one that minimises the error respect to the additionally defined inner product. That is

$$\tilde{u}_{R1} = \underset{\tilde{u}_{R1} = \bigotimes_{\alpha\in A} \tilde{u}_\alpha}{\arg\min} \frac{1}{2} \langle \tilde{u} - \tilde{u}_{R1}, \tilde{u} - \tilde{u}_{R1}\rangle_\sim. \tag{4.7}$$

The stationarity of the functional leads to

$$\langle \tilde{u} - \tilde{u}_{R1}, v\rangle_\sim = 0 \quad \forall v \in \tilde{\mathcal{V}}^{test} = \Big\{ v \in \mathcal{V} : v = \sum_{\alpha\in A} v_\alpha \bigotimes_{\beta\in A\backslash\{\alpha\}} \tilde{u}_\beta,\ v_\alpha \in \mathcal{V}_\alpha \Big\}. \tag{4.8}$$

On the other hand, expanding the inner product we get

$$\tilde{u}_{R1} = \underset{\tilde{u}_{R1} = \bigotimes_{\alpha\in A} \tilde{u}_\alpha}{\arg\min} \frac{1}{2} \langle \tilde{u} - \tilde{u}_{R1}, \tilde{u} - \tilde{u}_{R1}\rangle_\sim = \underset{\tilde{u}_{R1} = \bigotimes_{\alpha\in A} \tilde{u}_\alpha}{\arg\min} \frac{1}{2} \langle \tilde{u}_{R1}, \tilde{u}_{R1}\rangle_\sim - B(\tilde{\tilde{u}}, \tilde{u}_{R1}).$$

The stationarity condition is now:

$$B(u, v) = \langle \tilde{u}_{R1}, v\rangle_\sim = \langle \tilde{u}, v\rangle_\sim - \langle \tilde{u} - \tilde{u}_{R1}, v\rangle_\sim = L(v) \quad \forall v \in \tilde{\mathcal{V}}^{test}. \tag{4.9}$$

Note that the term $\langle \tilde{u} - \tilde{u}_{R1}, v\rangle_\sim$ is null by (4.8). Now we have equations (4.5) and (4.9) that involve the exact solution. If we substitute the exact solution on this equations by the rank-one approximation, we obtain the Petrov-Galerkin formulation:

$$B(u_{R1}, v) = L(v) \quad \forall v \in \tilde{\mathcal{V}}^{test}, \tag{4.10}$$

$$B(v, \tilde{u}_{R1}) = \langle u_{R1}, v\rangle \quad \forall v \in \mathcal{V}^{test}. \tag{4.11}$$

Where $\mathcal{V}^{test}$ and $\tilde{\mathcal{V}}^{test}$ are as defined in (4.5) and (4.8).

Remark 4.1: We interpret the previous equations as a Petrov-Galerkin formulation noting that (4.10) is identical to the standard Galerkin PGD formulation with the exception that in the definition of the space of test functions the functions u_β are substituted by $\tilde{u}_\beta$. With that interpretation, the function of (4.11) is to determine which are the dual functions. But note that we have shown that actually equation (4.11) is the one that minimises the error of the rank-one representation of the primal solution and (4.10) minimises the error of the dual function.

4.1.1 Interpretation as a dual variation problem

In his original paper, Nouy introduces equations (4.10) and (4.11) as the stationary conditions of a dual variation problem. We present it in the original form, that is, restricted to the case of separation of two dimensions (space and time).

$$(\omega\lambda,\tilde{\omega}\tilde{\lambda}) = \arg\max_{\substack{\tilde{\omega}\in\mathcal{V}_S\\ \tilde{\lambda}\in\mathcal{V}_T}}\min_{\substack{\omega\in\mathcal{V}_S\\ \lambda\in\mathcal{V}_T}} \mathcal{L}(\omega\lambda,\tilde{\omega}\tilde{\lambda}) = \arg\max_{\substack{\tilde{\omega}\in\mathcal{V}_S\\ \tilde{\lambda}\in\mathcal{V}_T}}\min_{\substack{\omega\in\mathcal{V}_S\\ \lambda\in\mathcal{V}_T}} \frac{1}{2}\langle\omega\lambda,\omega\lambda\rangle - B(u_M+\omega\lambda,\tilde{\omega}\tilde{\lambda}) + L(\tilde{\omega}\tilde{\lambda}). \tag{4.12}$$

Where $u_M = \sum_{m=1}^{M}\omega^m\lambda^m$ stands for the previously computed modes. For simplicity, we take $u_M = 0$. See section 4.3 for the general case. One can check easily that the equations that make the functional stationary are (4.10) and (4.11).

It is straightforward to see that the functions ω and λ that are obtained by imposing the corresponding stationarity conditions of $\mathcal{L}$ are in fact at a relative minimum. Noting that $\mathcal{L}\to\infty$ as $\|\omega\|\to\infty$ or $\|\lambda\|\to\infty$ it follows that the solution of equation (4.11) is the argument of the minimum of $\mathcal{L}$ provided that the solution is unique. In fact, equations (4.4) and (4.12) are quite similar.

However, we claim that the functions $\tilde{\omega}$ and $\tilde{\lambda}$ that make the functional stationary are not the arguments of the maximum. First, note that by equation (4.10)

$$B(\omega\lambda,\tilde{\omega}\tilde{\lambda}) - L(\tilde{\omega}\tilde{\lambda}) = 0.$$

Now pick arbitrary $\tilde{\omega}^*\in\mathcal{V}_S,\ \tilde{\lambda}^*\in\mathcal{V}_T$. The function defined as $(\tilde{\omega}+\tilde{\omega}^*)(\tilde{\lambda}+\tilde{\lambda}^*)$ is a rank one function. If $(\tilde{\omega},\tilde{\lambda})$ defines a maximum on $\mathcal{L}$, then

$$B(\omega\lambda,(\tilde{\omega}+\tilde{\omega}^*)(\tilde{\lambda}+\tilde{\lambda}^*)) - L((\tilde{\omega}+\tilde{\omega}^*)(\tilde{\lambda}+\tilde{\lambda}^*)) \geqslant 0 \quad \forall\tilde{\omega}^*\in\mathcal{V}_S,\ \tilde{\lambda}^*\in\mathcal{V}_T.$$

Expanding the forms we get

$$\begin{aligned}B(\omega\lambda,\tilde{\omega}\tilde{\lambda}) - L(\tilde{\omega}\tilde{\lambda}) + B(\omega\lambda,\tilde{\omega}^*\tilde{\lambda}) - L(\tilde{\omega}^*\tilde{\lambda}) + B(\omega\lambda,\tilde{\omega}\tilde{\lambda}^*) - L(\tilde{\omega}\tilde{\lambda}^*) +&\\ B(\omega\lambda,\tilde{\omega}^*\tilde{\lambda}^*) - L(\tilde{\omega}^*\tilde{\lambda}^*) \geqslant 0 \quad \forall\tilde{\omega}^*\in\mathcal{V}_S,\tilde{\lambda}^*\in\mathcal{V}_T.&\end{aligned}$$

By equation (4.10) the six first terms cancel leading to

$$B(\omega\lambda,\tilde{\omega}^*\tilde{\lambda}^*) - L(\tilde{\omega}^*\tilde{\lambda}^*) \geqslant 0 \quad \forall\tilde{\omega}^*\in\mathcal{V}_S,\tilde{\lambda}^*\in\mathcal{V}_T.$$

Noting that for fixed $\omega\lambda$ the previous expression defines a linear form, we get that the condition for $(\tilde{\omega},\tilde{\lambda})$ being a maximum is

$$B(\omega\lambda,\tilde{\omega}^*\tilde{\lambda}^*) - L(\tilde{\omega}^*\tilde{\lambda}^*) = 0 \quad \forall\tilde{\omega}^*\in\mathcal{V}_S,\tilde{\lambda}^*\in\mathcal{V}_T.$$

The previous equation is equivalent to state that $\omega\lambda$ is a solution of the full order problem. In this particular case, the functional $\mathcal{L}$ is independent of the dual functions and makes no sense to talk about maximum respect to the dual functions.

With that we have shown that the functions $(\tilde{\omega},\tilde{\lambda})$ in general do not define a maximum on $\mathcal{L}$. Instead of talking about a *minimax* problem it is more accurate to talk about making the functional $\mathcal{L}$ stationary. This subtile change makes no difference on the method as the stationary conditions are equivalent.

4.2 Petrov-Galerkin as an enlarged Galerkin problem

In this section, we present a method for obtaining a problem equivalent to the Petrov-Galerkin problem stated in the previous section with the form of a standard Galerkin PGD problem. With this one can use the same rank-one solver for the Galerkin and Petrov-Galerkin methods leading to a non-intrusive implementation. This allows us to treat the non-linear Petrov-Galerkin method as a linear Galerkin one at the cost of doubling the discretisation space as we shall see.

4.2.1 Abstract formulation

In the Petrov-Galerkin method, what we search for is a pair of vectors $(u_{R1}, \tilde{u}_{R1})$. In this section we focus on how to turn this problem in finding a single vector representing the previous one in a larger space. We define the following Hilbert space $\hat{\mathcal{V}}$:

$$(u_{R1}, \tilde{u}_{R1}) \in \mathcal{V} \oplus \mathcal{V} = \big(\bigotimes_{\alpha \in A} \mathcal{V}_\alpha\big) \oplus \big(\bigotimes_{\alpha \in A} \mathcal{V}_\alpha\big) \hookrightarrow \bigotimes_{\alpha \in A} \big(\mathcal{V}_\alpha \oplus \mathcal{V}_\alpha\big) =: \hat{\mathcal{V}}.$$

Where one possible definition of the imbedding is

$$\Big(\sum_{m \in M_u} \bigotimes_{\alpha \in A} u_\alpha^m, \sum_{m \in M_{\tilde{u}}} \bigotimes_{\alpha \in A} \tilde{u}_\alpha^m\Big) \mapsto \sum_{m \in M_u} \bigotimes_{\alpha \in A} (u_\alpha^m, 0) + \sum_{m \in M_{\tilde{u}}} \bigotimes_{\alpha \in A} (0, \tilde{u}_\alpha).$$

We extend the definition of the imbedding using a density argument.

In the Petrov-Galerkin method we solve for two tuples: $(u_\alpha)_{\alpha \in A} \in \bigoplus_{\alpha \in A} \mathcal{V}_\alpha$ and $(\tilde{u}_\alpha)_{\alpha \in A} \in \bigoplus_{\alpha \in A} \mathcal{V}_\alpha$. With this approach we solve for a single tuple $(\hat{u}_\alpha)_{\alpha \in A} \in \bigoplus_{\alpha \in A} \mathcal{V}_\alpha^2$. Now we have to redefine the bilinear form B and functional L such that we obtain an equivalent problem in the enlarged space $\hat{\mathcal{V}}$. That is, we want to compute $\hat{u}_{R1} = \bigotimes_{\alpha \in A} \hat{u}_\alpha$ using the Galerkin method with bilinear form $\hat{B}$ and functional $\hat{L}$ on $\hat{\mathcal{V}}$ such that the obtained solution is equivalent to the two solutions of the Petrov-Galerkin problem, i.e. $\hat{u}_\alpha = (u_\alpha, \tilde{u}_\alpha)\ \forall \alpha \in A$, where vectors u_α and $\tilde{u}_\alpha$ are the same vectors obtained using the Petrov-Galerkin method. For that, first we define the projection operators

$$\begin{aligned} P_\alpha : \mathcal{V}_\alpha \oplus \mathcal{V}_\alpha &\to \mathcal{V}_\alpha \\ (u_\alpha, \tilde{u}_\alpha) &\mapsto u_\alpha, \end{aligned}$$

$$\begin{aligned} P : \hat{\mathcal{V}} &\to \mathcal{V} \\ \bigotimes_{\alpha \in A} \hat{u}_\alpha^i &\mapsto \bigotimes_{\alpha \in A} P_\alpha \hat{u}_\alpha^i, \end{aligned}$$

$$\begin{aligned} \tilde{P}_\alpha : \mathcal{V}_\alpha \oplus \mathcal{V}_\alpha &\to \mathcal{V}_\alpha \\ (u_\alpha, \tilde{u}_\alpha) &\mapsto \tilde{u}_\alpha, \end{aligned}$$

$$\begin{aligned} \tilde{P} : \hat{\mathcal{V}} &\to \mathcal{V} \\ \bigotimes_{\alpha \in A} \hat{u}_\alpha^i &\mapsto \bigotimes_{\alpha \in A} \tilde{P}_\alpha \hat{u}_\alpha^i. \end{aligned}$$

The definition of P and $\tilde{P}$ is extended to their whole domain using linearity and density arguments,.

The bilinear and linear forms that ensure that the solution is equivalent are defined as

$$\hat{B}(\hat{u}, \hat{v}) = B(P\hat{u}, \tilde{P}\hat{v}) + B(P\hat{v}, \tilde{P}\hat{u}) - \langle P\hat{u}, P\hat{v} \rangle, \tag{4.13}$$

$$\hat{L}(\hat{v}) = L(\tilde{P}\hat{v}). \tag{4.14}$$

Alternatively, using the projection operators and the separation properties of the forms, the forms can be defined equivalently on the subset of rank-one elements as

$$\hat{B}\Big(\bigotimes_{\alpha\in A}\hat{u}_\alpha, \bigotimes_{\alpha\in A}\hat{v}_\alpha\Big) = \sum_{m\in M_B}\left(\prod_{\alpha\in A}\hat{B}^{m,primal}_\alpha(\hat{u}_\alpha,\hat{v}_\alpha) + \prod_{\alpha\in A}\hat{B}^{m,dual}_\alpha(\hat{u}_\alpha,\hat{v}_\alpha)\right) - \Big\langle\bigotimes_{\alpha\in A}P_\alpha\hat{u}_\alpha, \bigotimes_{\alpha\in A}P_\alpha\hat{v}_\alpha\Big\rangle, \tag{4.15}$$

$$\hat{L}\Big(\bigotimes_{\alpha\in A}\hat{v}_\alpha\Big) = \sum_{m\in M_L}\prod_{\alpha\in A}\hat{L}^m_\alpha(\hat{v}_\alpha). \tag{4.16}$$

Where

$$\hat{B}^{m,primal}_\alpha(\hat{u}_\alpha,\hat{v}_\alpha) = B^m_\alpha(P_\alpha\hat{u}_\alpha, \tilde{P}_\alpha\hat{v}_\alpha), \tag{4.17}$$

$$\hat{B}^{m,dual}_\alpha(\hat{u}_\alpha,\hat{v}_\alpha) = B^m_\alpha(P_\alpha\hat{v}_\alpha, \tilde{P}_\alpha\hat{u}_\alpha), \tag{4.18}$$

$$\hat{L}^m_\alpha(\hat{v}_\alpha) = L^m_\alpha(\tilde{P}_\alpha\hat{v}_\alpha). \tag{4.19}$$

The enlarged problem to solve is of Galerkin PGD type:

$$\hat{B}(\hat{u}_{R1},\hat{v}) = \hat{L}(\hat{v}) \quad \forall\hat{v}\in\hat{\mathcal{V}}^{test} = \Big\{\hat{v}\in\hat{\mathcal{V}} : \hat{v} = \sum_{\alpha\in A}\hat{v}_\alpha\bigotimes_{\beta\in A\backslash\{\alpha\}}\hat{u}_\beta,\ \hat{v}_\alpha\in\mathcal{V}^2_\alpha\Big\}. \tag{4.20}$$

Now, we prove that solving a Galerkin problem for the previous forms is equivalent to the Petrov-Galerkin problem. That is $u_{R1} = P\hat{u}_{R1}$ and $\tilde{u}_{R1} = \tilde{P}\hat{u}_{R1}$. First, we express the space of test vectors as

$$\hat{\mathcal{V}}^{test} = \mathcal{V}^{test}_0 + \tilde{\mathcal{V}}^{test}_0.$$

Where

$$\mathcal{V}^{test}_0 = \Big\{\hat{v}\in\hat{\mathcal{V}} : \hat{v} = \sum_{\alpha\in A}\hat{v}_\alpha\bigotimes_{\beta\in A\backslash\{\alpha\}}\hat{u}_\beta,\ \hat{v}_\alpha = (v_\alpha, 0),\ v_\alpha\in\mathcal{V}_\alpha\Big\},$$

$$\tilde{\mathcal{V}}^{test}_0 = \Big\{\hat{v}\in\hat{\mathcal{V}} : \hat{v} = \sum_{\alpha\in A}\hat{v}_\alpha\bigotimes_{\beta\in A\backslash\{\alpha\}}\hat{u}_\beta,\ \hat{v}_\alpha = (0, \tilde{v}_\alpha),\ \tilde{v}_\alpha\in\mathcal{V}_\alpha\Big\}.$$

With this decomposition and by the linearity of the forms on $\hat{v}$, equation (4.20) can be interpreted as a system of two equations:

$$\hat{B}(\hat{u}_{R1},\hat{v}) = \hat{L}(\hat{v}) \quad \forall\hat{v}\in\mathcal{V}^{test}_0, \tag{4.21}$$

$$\hat{B}(\hat{u}_{R1},\hat{v}) = \hat{L}(\hat{v}) \quad \forall\hat{v}\in\tilde{\mathcal{V}}^{test}_0. \tag{4.22}$$

Equation (4.21) can be reformulated as a system of n_D equations. The equation for the i-th dimension is, given $\{\hat{u}_\alpha\}_{\alpha\in A\backslash\{i\}}$, find $\hat{u}_i\in\mathcal{V}_i\oplus\mathcal{V}_i$ such that

$$\hat{B}(\hat{u}_{R1},\hat{v}) = \hat{L}(\hat{v}) \quad \forall\hat{v}\in\mathcal{V}^{test}_{0,i} = \Big\{\hat{v}\in\hat{\mathcal{V}} : \hat{v} = \hat{v}_i\bigotimes_{\alpha\in A\backslash\{i\}}\hat{u}_\alpha,\ \hat{v}_i = (v_i, 0),\ v_i\in\mathcal{V}_i\Big\}.$$

Expanding the forms with respect to their definitions one gets:

$$\sum_{m\in M_B}\Big(\prod_{\alpha\in A}\hat{B}^{m,primal}_\alpha(\hat{u}_\alpha,\hat{v}_\alpha) + \prod_{\alpha\in A}\hat{B}^{m,dual}_\alpha(\hat{u}_\alpha,\hat{v}_\alpha)\Big) - \prod_{\alpha\in A}\langle P_\alpha\hat{u}_\alpha, P_\alpha\hat{v}_\alpha\rangle_\alpha =$$
$$\sum_{m\in M_L}\prod_{\alpha\in A}\hat{L}^m_\alpha(\hat{v}_\alpha) \quad \forall\hat{v}_i\in\{v\in\hat{\mathcal{V}}_i : v = (v_i, 0),\ v_i\in\mathcal{V}_i\}.$$

Where $\hat{v}_\alpha = \hat{u}_\alpha\ \forall\alpha\in A\backslash\{i\}$. Note that $\tilde{P}_i\hat{v}_i = 0$, this simplifies the equation as some terms are null:

$$\sum_{m\in M_B}\prod_{\alpha\in A}\hat{B}^{m,dual}_\alpha(\hat{u}_\alpha,\hat{v}_\alpha) = \prod_{\alpha\in A}\langle P_\alpha\hat{u}_\alpha, P_\alpha\hat{v}_\alpha\rangle \quad \forall\hat{v}_i\in\{v\in\hat{\mathcal{V}}_i : v = (v_i, 0),\ v_i\in\mathcal{V}_i\}.$$

Finally, the terms are regrouped using the separability property of the bilinear forms and the inner product. The system of equations is reformulated as a single equation imposing the multi-orthogonality of the residual:

$$B\big(P\hat{v}, \tilde{P}\hat{u}_{R1}\big) = \langle P\hat{u}_{R1}, P\hat{v}\rangle \quad \forall \hat{v} \in \mathcal{V}_0^{test}.$$

this equation is equivalent to (4.11).

Now we apply the same steps to equation (4.22). The associated equation for the i-th dimension is

$$\hat{B}(\hat{u}_{R1}, \hat{v}) = \hat{L}(\hat{v}) \quad \forall \hat{v} \in \tilde{\mathcal{V}}_i^{test} = \Big\{\hat{v} \in \hat{\mathcal{V}} : \hat{v} = \hat{v}_i \bigotimes_{\alpha \in A\setminus\{i\}} \hat{u}_\alpha,\ \hat{v}_i = (0, \tilde{v}_i),\ \tilde{v}_i \in \mathcal{V}_i\Big\}.$$

Expanding:

$$\sum_{m\in M_B}\Big(\prod_{\alpha\in A}\hat{B}_\alpha^{m,primal}(\hat{u}_\alpha,\hat{v}_\alpha) + \prod_{\alpha\in A}\hat{B}_\alpha^{m,dual}(\hat{u}_\alpha,\hat{v}_\alpha)\Big) - \prod_{\alpha\in A}\langle P_\alpha\hat{u}_\alpha, P_\alpha\hat{v}_\alpha\rangle_\alpha =$$
$$\sum_{m\in M_L}\prod_{\alpha\in A}\hat{L}_\alpha^m(\hat{v}_\alpha) \quad \forall \hat{v}_i \in \{v \in \hat{\mathcal{V}}_i : v = (0,\tilde{v}_i),\ \tilde{v}_i \in \mathcal{V}_i\}.$$

Noting that $P_i\hat{v}_i = 0$:

$$\sum_{m\in M_B}\prod_{\alpha\in A}\hat{B}_\alpha^{m,primal}(\hat{u}_\alpha,\hat{v}_\alpha) = \sum_{m\in M_L}\prod_{\alpha\in A}\hat{L}_\alpha^m(\hat{v}_\alpha) \quad \forall \hat{v}_i \in \{v \in \hat{\mathcal{V}}_i : v = (0,\tilde{v}_i),\ \tilde{v}_i \in \mathcal{V}_i\}.$$

This is reformulated as

$$B\big(P\hat{u}_{R1}, \tilde{P}\hat{v}\big) = L(\tilde{P}\hat{v}) \quad \forall \hat{v} \in \tilde{\mathcal{V}}_0^{test},$$

which is equivalent to (4.10).

4.2.2 Advection-diffusion formulation

The Petrov-Galerkin approximation of the problem is to find $(\omega, \tilde{\omega}, \lambda, \tilde{\lambda}) \in \mathcal{V}_S^2 \oplus \mathcal{V}_T^2$ such that

$$B(\omega\lambda, \tilde{\omega}^*\tilde{\lambda} + \tilde{\omega}\tilde{\lambda}^*) = L(\tilde{\omega}^*\tilde{\lambda} + \tilde{\omega}\tilde{\lambda}^*) \quad \forall(\tilde{\omega}^*, \tilde{\lambda}^*) \in \mathcal{V}_S \oplus \mathcal{V}_T, \tag{4.23}$$
$$B(\omega^*\lambda + \omega\lambda^*, \tilde{\omega}\tilde{\lambda}) = \langle \omega\lambda, \omega^*\lambda + \omega\lambda^*\rangle \quad \forall(\omega^*, \lambda^*) \in \mathcal{V}_S \oplus \mathcal{V}_T. \tag{4.24}$$

Where B and L are the same forms defined for the Galerkin PGD algorithm.

The system can be rewritten as a system of four coupled equations:

$$\begin{aligned}
B(\omega\lambda, \tilde{\omega}^*\tilde{\lambda}) &= L(\tilde{\omega}^*\tilde{\lambda}) \quad \forall \tilde{\omega}^* \in \mathcal{V}_S,\\
B(\omega\lambda, \tilde{\omega}\tilde{\lambda}^*) &= L(\tilde{\omega}\tilde{\lambda}^*) \quad \forall \tilde{\lambda}^* \in \mathcal{V}_T,\\
B(\omega^*\lambda, \tilde{\omega}\tilde{\lambda}) &= \langle \omega\lambda, \omega^*\lambda\rangle \quad \forall \omega^* \in \mathcal{V}_S,\\
B(\omega\lambda^*, \tilde{\omega}\tilde{\lambda}) &= \langle \omega\lambda, \omega\lambda^*\rangle \quad \forall \lambda^* \in \mathcal{V}_T.
\end{aligned}$$

To solve the problem as a Galerkin enlarged system, define the enlarged vectors of unknowns as

$$\omega = \sum_{i=1}^{n_{DoF_S}} N_i^S \boldsymbol{\omega}_i, \qquad \tilde{\omega} = \sum_{i=1}^{n_{DoF_S}} N_i^S \tilde{\boldsymbol{\omega}}_i, \qquad \hat{\boldsymbol{\omega}} = (\boldsymbol{\omega}, \tilde{\boldsymbol{\omega}}), \qquad \hat{\omega} = \sum_{i=1}^{2n_{DoF_S}} \hat{N}_i^S \hat{\boldsymbol{\omega}}_i \in \mathcal{V}_S^2 := \hat{\mathcal{V}}_S,$$
$$\lambda = \sum_{i=1}^{n_{DoF_T}} N_i^T \boldsymbol{\lambda}_i, \qquad \tilde{\lambda} = \sum_{i=1}^{n_{DoF_T}} N_i^T \tilde{\boldsymbol{\lambda}}_i, \qquad \hat{\boldsymbol{\lambda}} = (\boldsymbol{\lambda}, \tilde{\boldsymbol{\lambda}}), \qquad \hat{\lambda} = \sum_{i=1}^{2n_{DoF_T}} \hat{N}_i^T \hat{\boldsymbol{\lambda}}_i \in \mathcal{V}_S^T := \hat{\mathcal{V}}_T.$$

Note that the tensor product of two elements $\hat{\omega}$ and $\hat{\lambda}$ is

$$\hat{\omega}\hat{\lambda} = (\omega, \tilde{\omega}) \otimes (\lambda, \tilde{\lambda}) = \begin{pmatrix} \omega\lambda & \omega\tilde{\lambda} \\ \tilde{\omega}\lambda & \tilde{\omega}\tilde{\lambda} \end{pmatrix}.$$

Where the terms $\omega\tilde{\lambda}$ and $\tilde{\omega}\lambda$ are meaningless. That is why we define the embedding $\mathcal{V}\oplus\mathcal{V}\hookrightarrow\hat{\mathcal{V}}$ as

$$\left(\omega\otimes\lambda,\tilde{\omega}\otimes\tilde{\lambda}\right)\mapsto(\omega,0)\otimes(\lambda,0)+(0,\tilde{\omega})\otimes(0,\tilde{\lambda}).$$

Again, we extend the definition of the embedding using a linearity and density argument.

Now we define the enlarged bilinear form as

$$\hat{B}(\omega\lambda,\omega^*\lambda^*)=\sum_{m\in\{e,d,a,0\}}\hat{B}_S^{m,primal}(\omega,\omega^*)\hat{B}_T^{m,primal}(\lambda,\lambda^*)+\hat{B}_S^{m,dual}(\omega,\omega^*)\hat{B}_T^{m,dual}(\lambda,\lambda^*) \\ -\langle\omega,\omega^*\rangle_S\langle\lambda,\lambda^*\rangle_T. \tag{4.25}$$

The matrices associated to the different modes are:

$$\left[\hat{\boldsymbol{K}}_\alpha^{m,primal}\right]_{ij}=\hat{B}_\alpha^{m,primal}(\hat{N}_j^\alpha,\hat{N}_i^\alpha),\quad\alpha\in\{S,T\},m\in\{e,d,a,0\},$$
$$\hat{\boldsymbol{K}}_\alpha^{m,primal}=\begin{pmatrix}\mathbf{0}&\mathbf{0}\\\boldsymbol{K}_\alpha^m&\mathbf{0}\end{pmatrix},\tag{4.26}$$
$$\left[\hat{\boldsymbol{K}}_\alpha^{m,dual}\right]_{ij}=\hat{B}_\alpha^{m,dual}(\hat{N}_j^\alpha,\hat{N}_i^\alpha),\quad\alpha\in\{S,T\},m\in\{e,d,a,0\},$$
$$\hat{\boldsymbol{K}}_\alpha^{m,dual}=\begin{pmatrix}\mathbf{0}&(\hat{\boldsymbol{K}}_\alpha^m)^T\\\mathbf{0}&\mathbf{0}\end{pmatrix},\tag{4.27}$$
$$\left[\hat{\boldsymbol{K}}_S^0\right]_{ij}=-\langle R_S\hat{N}_i^S,R_S\hat{N}_j^S\rangle_S,$$
$$\hat{\boldsymbol{K}}_S^0=\begin{pmatrix}-\boldsymbol{M}_S&\mathbf{0}\\\mathbf{0}&\mathbf{0}\end{pmatrix},\tag{4.28}$$
$$\left[\hat{\boldsymbol{K}}_T^0\right]_{ij}=\langle R_T\hat{N}_i^T,R_T\hat{N}_j^T\rangle_T,$$
$$\hat{\boldsymbol{K}}_T^0=\begin{pmatrix}\boldsymbol{M}_T&\mathbf{0}\\\mathbf{0}&\mathbf{0}\end{pmatrix}.\tag{4.29}$$

The enlarged linear functional is

$$\hat{L}(\omega\lambda)=\sum_{m\in\{f,0\}}\hat{L}_S^m(\omega)\hat{L}_T^m(\lambda).\tag{4.30}$$

Their associated vectors are

$$\left[\hat{\boldsymbol{f}}_\alpha^m\right]_i=\hat{L}_\alpha^m(\hat{N}_i^\alpha),\quad\alpha\in\{S,T\},m\in\{f,0\},$$
$$\hat{\boldsymbol{f}}_\alpha^m=\begin{pmatrix}\mathbf{0}\\\boldsymbol{f}_\alpha^m\end{pmatrix}.\tag{4.31}$$

The Petrov-Galerkin problem can be solved using a Galerkin approach with the enlarged forms:

$$\hat{B}(\hat{\omega}\hat{\lambda},\hat{\omega}^*\hat{\lambda}+\hat{\omega}\hat{\lambda}^*)=\hat{L}(\hat{\omega}^*\hat{\lambda}+\hat{\omega}\hat{\lambda}^*)\quad\forall(\hat{\omega}^*,\hat{\lambda}^*)\in\hat{\mathcal{V}}_S\oplus\hat{\mathcal{V}}_T.\tag{4.32}$$

Where $\hat{\omega}=(\omega,\tilde{\omega})$ and $\hat{\lambda}=(\lambda,\tilde{\lambda})$.

4.3 Additional modes

In this section we discuss the problem of improving the accuracy of the rank-one approximation by sequentially adding rank-one functions to the approximation of the solution. In the case of the Petrov-Galerkin method, the bilinear form $\bar{B}$ is not as obvious as in the Galerkin problem. The reason why $\bar{B}\neq B$ is because in the formulation of the rank-one problem, the dual function must not take into account the contribution of the previously computed dual solutions while, when computing the primal solutions, of course we must subtract the contribution to the residual of the previously computed modes.

4.3.1 Abstract formulation

Assume that $\hat{u}_M = \sum_{m=1}^{M} \hat{u}_{R1}^m$ is already computed. We need to subtract the contribution of $P\hat{u}_M$ to the functional $\hat{L}$. That is achieved by defining

$$\tilde{\hat{B}}\Big(\bigotimes_{\alpha\in A} \hat{u}_\alpha, \bigotimes_{\alpha\in A} \hat{v}_\alpha\Big) = \sum_{m\in M_B} \left(\prod_{\alpha\in A} \hat{B}_\alpha^{m,primal}(\hat{u}_\alpha, \hat{v}_\alpha)\right) = B(P\hat{u}, \tilde{P}\hat{v}). \tag{4.33}$$

Note that $\tilde{\hat{B}}(\hat{u}, \hat{v})$ is independent of $\tilde{P}\hat{u}$ meaning that the previous modes of the dual solution has not effect on the bilinear form and it is independent of $P\hat{v}$ meaning that the previous mode contributions do not have any direct contribution to the dual function $\tilde{u}_{R1}$.

With this the problem to solve is

$$\hat{B}(\hat{u}_{R1}, \hat{v}) = \hat{L}(\hat{v}) - \tilde{\hat{B}}(\hat{u}_M, \hat{v}) \quad \forall \hat{v} \in \hat{\mathcal{V}}^{test}. \tag{4.34}$$

4.3.2 Advection-diffusion formulation

Using the previous definition we get the following bilinear form associated to the contribution of the previous modes approximation $\hat{u}_M$:

$$\tilde{\hat{B}}(\omega\lambda, \omega^*\lambda^*) = \sum_{m\in\{e,d,a,0\}} \hat{B}_S^{m,primal}(\omega, \omega^*) \hat{B}_T^{m,primal}(\lambda, \lambda^*). \tag{4.35}$$

The matrices associated to the previous modes contribution are

$$\tilde{\hat{\boldsymbol{K}}}_\alpha^m = \hat{\boldsymbol{K}}_\alpha^{m,primal}, \quad \alpha \in \{S, T\}, m \in \{e, d, a, 0\}.$$

The Galerkin enlarged formulation reads

$$\hat{B}(\hat{\omega}\hat{\lambda}, \hat{\omega}^*\hat{\lambda} + \hat{\omega}\hat{\lambda}^*) = \hat{L}(\hat{\omega}^*\hat{\lambda} + \hat{\omega}\hat{\lambda}^*) - \tilde{\hat{B}}(\hat{u}_M, \hat{\omega}^*\hat{\lambda} + \hat{\omega}\hat{\lambda}^*) \quad \forall(\hat{\omega}^*, \hat{\lambda}^*) \in \mathcal{V}_S^2 \oplus \mathcal{V}_T^2. \tag{4.36}$$

4.4 Update

In this section we discuss the same method of updating the functions of all modes for some subset of the separated dimensions as presented in the Galerkin PGD method (section 3.3.1). In fact, as presented by Nouy in his paper, the method of update in the Petrov-Galerkin is exactly the same than for the Galerkin PGD formulation. The reason of this is that, as noted in the previous section, the dual functions are completely decoupled from mode to mode. Therefore it makes no sense to couple the dual functions with the update method. In the Petrov-Galerkin context, the problem of updating some dimensions after the computation of each new mode is equivalent to the update problem for the Galerkin method with the difference that the functions not to update are different as they are computed using the Petrov-Galerkin method. That is, imagine the solution $\hat{u}_M = \sum_{m=1}^{m} \hat{u}_{R1}$ has been computed using the Petrov-Galerkin method. Then the update only modifies the primal functions of the PGD decomposition of the solution. The input data for the update is $P(\hat{u}_M)$ and with that, the problem is reduced to a Galerkin PGD update problem.

4.5 Numerical example: Petrov-Galerkin without and with time update. Comparison with Galerkin

4.5.1 Numerical results

We present the solution of the same examples solved using the Galerkin PGD algorithm and compare the results. We have computed the solution of the plain Petrov-Galerkin and the one with update in the time dimension. Note that the method of updating all dimensions produce identical results as the Galerkin method.

Petrov-Galerkin without update

First, we present the results for the plain Petrov-Galerkin algorithm. In the figures we plot, in that order, the first ten spatial and temporal modes of the primal solution, the ten spatial and temporal corresponding modes of the dual solution and the reconstruction of the solution adding those ten modes.

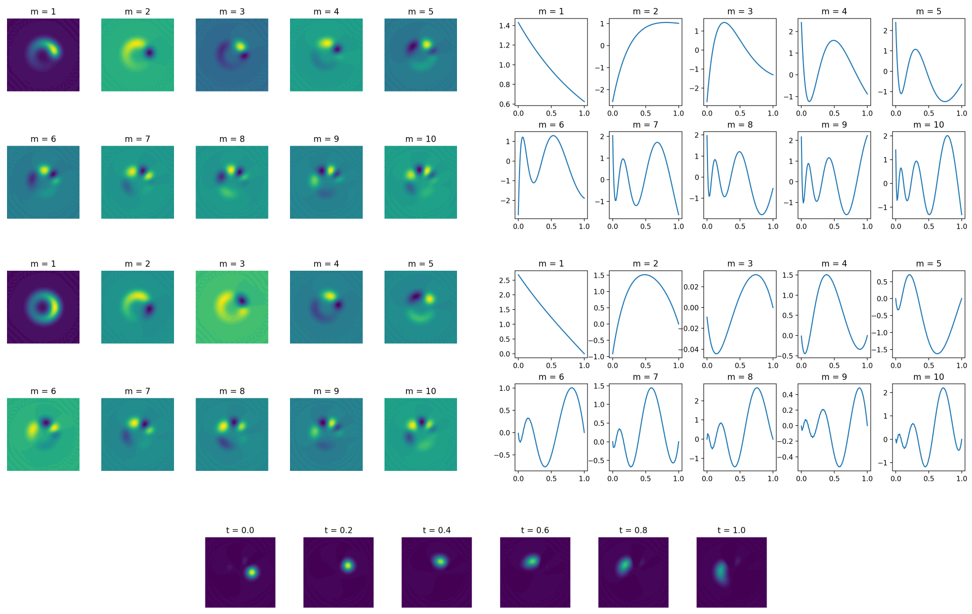

Figure 4.1 Petrov-Galerkin PGD approximation for the pure advection problem

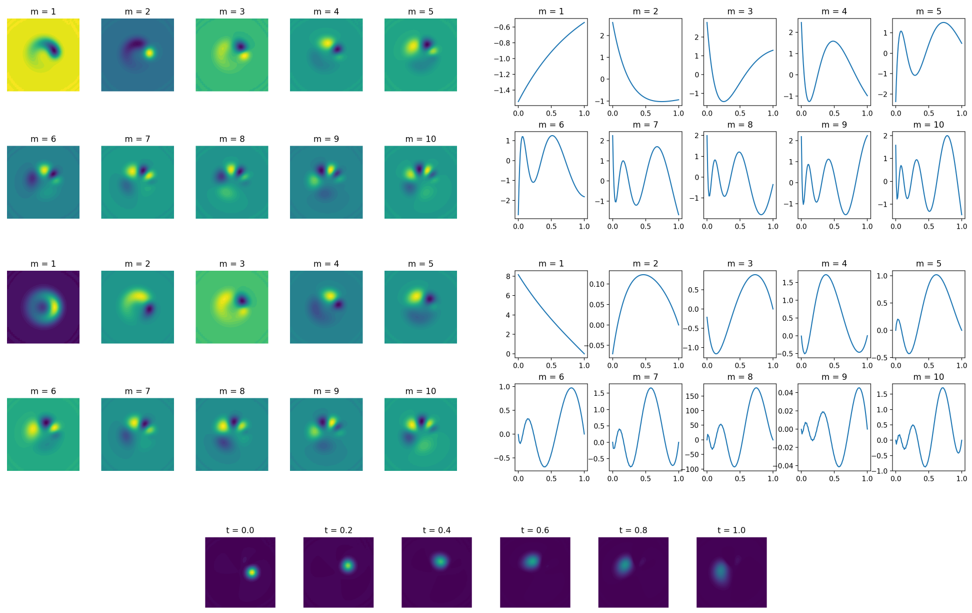

Figure 4.2 Petrov-Galerkin PGD approximation for the advection-diffusion problem

If we look at the reconstruction of the solution approximating it by only ten modes we see that there is a clear improvement in accuracy compared with the Galerkin method. We also see a faster scale reduction in both spatial and temporal functions revealing that this method is closer to being optimal than the Galerkin one. From the reconstruction of the solution we see that the error increases as time evolves as in the Galerkin method.

Petrov-Galerkin with time update

Now we present the results with the update algorithm in the temporal dimension. We do not present the results of the dual solutions. As they lack of meaning once the solution has been updated, the implementation used to implement the update algorithm, implies that the information of the dual solution is lost at the update step.

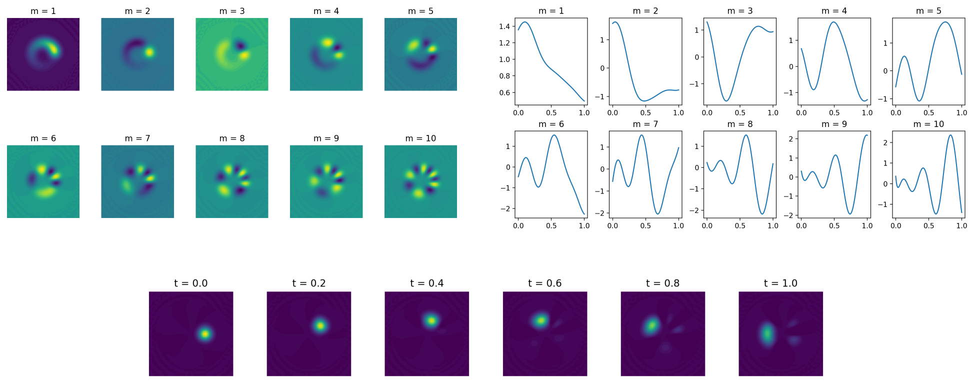

Figure 4.3 Petrov-Galerkin PGD approximation for the pure advection problem

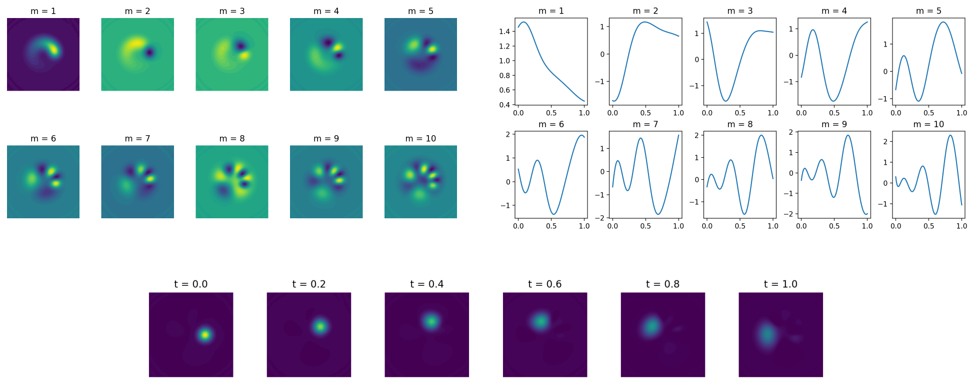

Figure 4.4 Petrov-Galerkin PGD approximation for the advection-diffusion problem

The results are similar to the plain Petrov-Galerkin and it is hard to extract any conclusion without additional information.

4.5.2 Convergence

Here we present the convergence properties of the methods and we compare them with the Galerkin PGD method. Using the plain Petrov-Galerkin method we have computed the PGD approximation up to 322 modes for the problem of pure advection and 144 for the problem of advection-diffusion. At that point, the PGD algorithm estimated that the solution is exact (up to numerical precision). For the Petrov-Galerkin method with time update we have computed the approximation up to 60 modes due to the increasing computational power as more modes are added. In the next plot, we show the convergence of the normalised error of the Petrov-Galerkin methods as well as of the plain Galerkin for comparison purposes.

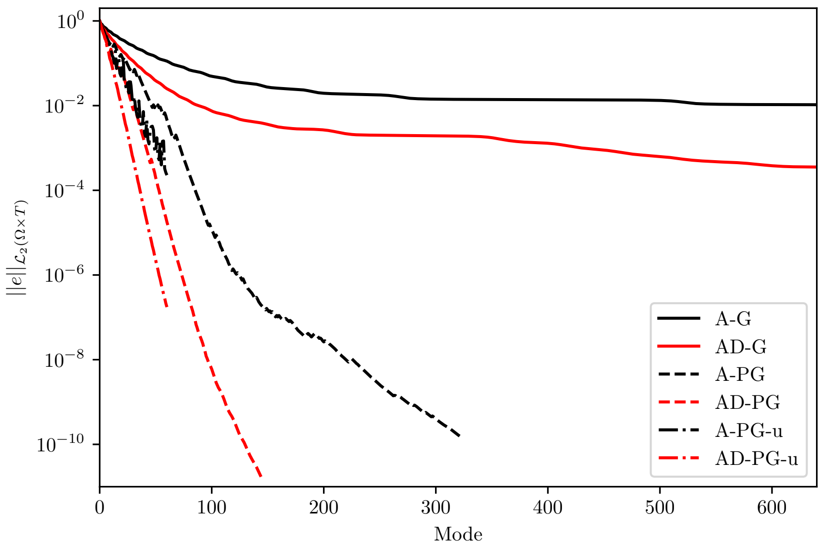

Figure 4.5 Error convergence

Here A stands for advection problem and AD for advection-diffusion problem, G for Galerkin method, PG for Petrov-Galerkin method and u for time update.

The comparison shows a great improvement in the convergence behaviour using the Petrov-Galerkin method. We see that for the plain Petrov-Galerkin, the order of convergence also degrades as the number of modes is increased. However, in this problem, the degradation is much lower than in the case of the plain Galerkin method. We also note that the update in the time dimension improves the accuracy of the method. Finally, as in the Galerkin method, the introduction of the diffusion term makes the problem to converge faster.

Now, we plot the error of the spatial dimension as a function of time for different number of modes.

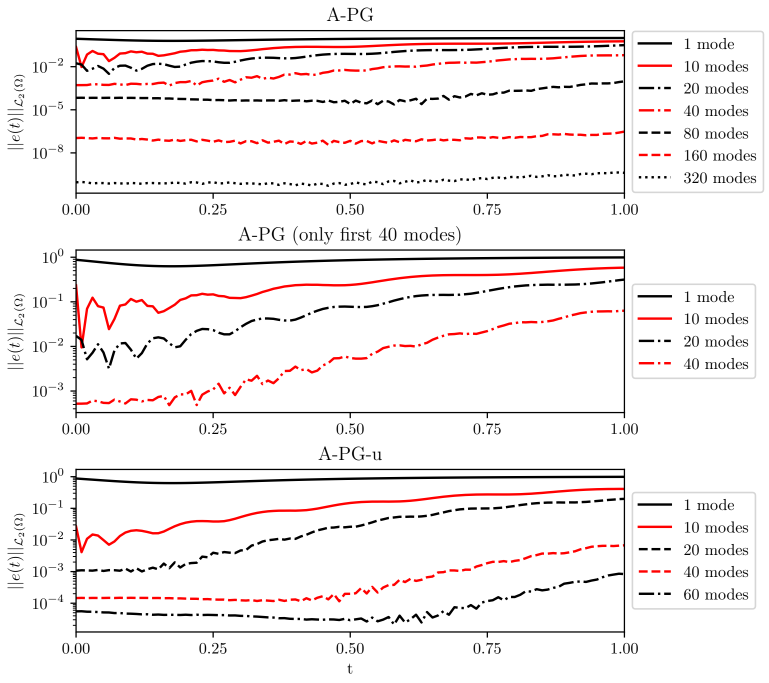

Figure 4.6 Temporal evolution of the error

In the case of Petrov-Galerkin the error increases exponentially with time as in the Galerkin case. However, we see that once the error is approximated up to a certain level of accuracy, then the error is constant until some instant of time and then it increases exponentially.

Finally, we see that in the case of Petrov-Galerkin (with and without update), the error estimator has an accuracy of order 1.

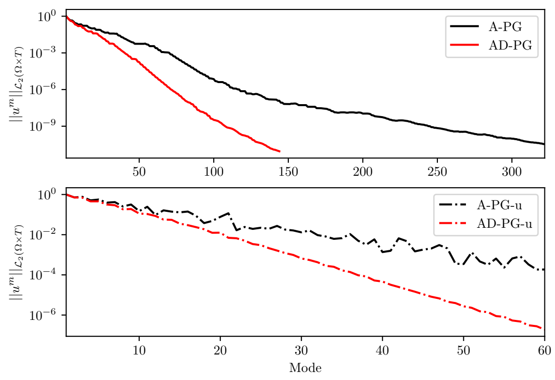

Figure 4.7 Modal amplitude

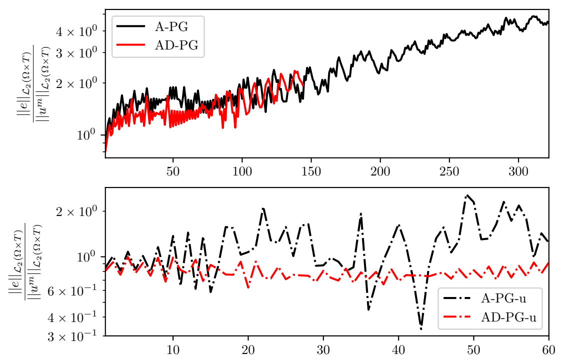

Figure 4.8 Error estimator accuracy

4.6 Concluding remarks

To conclude, we remark the clear improvement of the PGD method proposed by Nouy compared with the traditional Galerkin in non-self-adjoint problems. In case that the bilinear form defines an inner product the solution of both methods are similar. In fact, if one uses the bilinear form as the inner product to minimise the error, the Petrov-Galerkin and the Galerkin methods are equivalent.

About the update procedure, in the problems we solved we see that updating all temporal functions improve the solution but the increase of computational cost as more modes are added make this method inefficient for computing approximations with a high number of modes. In his paper, Nouy used the same example exposed in this thesis with an advection velocity ten times higher in magnitude. In that case, the plain methods present a slow convergence and updating the time functions after each step increases notably the accuracy of the method. We note as future work to study if updating only the last N modes (N being a relatively small number), we can obtain similar results for the convergence properties without increasing to much the computational cost.

Finally, we make an important remark about a possible modification to improve the computational efficiency of the Petrov-Galerkin algorithm. Note that at every fixed-point iteration, when computing the function of the α dimension, the resulting matrix is a linear combination of the matrices defined in (4.26 - 4.29) for fixed α. This is a 2×2 matrix with null coefficients at the lower-right entry. This means that the primal function can be computed by solving

$$\boldsymbol{K}_\alpha \boldsymbol{u}_\alpha = \boldsymbol{f}_\alpha.$$

Where $\boldsymbol{K}_\alpha$ is a linear combination of $\boldsymbol{K}_\alpha^m$ and $\boldsymbol{f}_\alpha$ the corresponding force vector. The dual function can be computed similarly. That is

$$\boldsymbol{K}_\alpha^T \tilde{\boldsymbol{u}}_\alpha = \boldsymbol{f}_\alpha.$$

Where $\boldsymbol{f}_\alpha$ is different from the previous case and the matrix $\boldsymbol{K}_\alpha^T$ is the transposed of the used in the previous problem. This means that if some matrix decomposition is performed to solve the linear system of the primal problem. The dual problem can be solved without having to recompute the decomposition. This makes the Petrov-Galerkin PGD almost as computationally cheap as the Galerkin PGD algorithm. However, this implementation is code intrusive.

This remark was made by Nouy in his paper for the case of a two dimensional PGD. We show that the result is valid also in the abstract formulation.

Fix a dimension α and consider all functions $\{u_\beta : \beta \in A \backslash \{\alpha\}\}$ as given. The bilinear form of equation (4.22) can be written as

$$\hat{B}\Big(\hat{u}_\alpha \otimes \bigotimes_{\beta \in A\backslash\{\alpha\}} \hat{u}_\beta, (0, \tilde{v}_\alpha) \otimes \bigotimes_{\beta \in A\backslash\{\alpha\}} \hat{u}_\beta\Big) = \sum_{m \in M_B} \prod_{\beta \in A\backslash\{\alpha\}} B_\beta^m(u_\beta, \tilde{u}_\beta) B_\alpha^m(u_\alpha, \tilde{v}_\alpha).$$

Were we have used the fact that $P_\alpha(0, \tilde{v}_\alpha) = 0$.

Conversely, the bilinear form of equation (4.21) can be written as

$$\begin{aligned} &\hat{B}\Big(\hat{u}_\alpha \otimes \bigotimes_{\beta \in A\backslash\{\alpha\}} \hat{u}_\beta, (v_\alpha, 0) \otimes \bigotimes_{\beta \in A\backslash\{\alpha\}} \hat{u}_\beta\Big) - \Big\langle u_\alpha \otimes \bigotimes_{\beta \in A\backslash\{\alpha\}} u_\beta, v_\alpha \otimes \bigotimes_{\beta \in A\backslash\{\alpha\}} u_\beta \Big\rangle = \\ &\qquad \sum_{m \in M_B} \prod_{\beta \in A\backslash\{\alpha\}} B_\beta^m(u_\beta, \tilde{u}_\beta) B_\alpha^m(v_\alpha, \tilde{u}_\alpha) - \Big\langle u_\alpha \otimes \bigotimes_{\beta \in A\backslash\{\alpha\}} u_\beta, v_\alpha \otimes \bigotimes_{\beta \in A\backslash\{\alpha\}} u_\beta \Big\rangle. \end{aligned}$$

Where we have used the fact that $\tilde{P}_\alpha(v_\alpha, 0) = 0$. The term of the inner product can be computed a priori as the primal solutions are known. Then, noting that the discretisation of $B_\alpha^m(u_\alpha, \tilde{v}_\alpha)$ is the transpose of the discretisation of $B_\alpha^m(v_\alpha, \tilde{u}_\alpha)$, it follows that the matrices $\boldsymbol{K}_\alpha$ and $\boldsymbol{K}_\alpha^T$ are in fact transposes of each other.

Chapter 5

Stabilisation of the weak form

It is well known that for advection dominated problems, if Dirichlet BCs are prescribed at the outflow, the FEM solution may present spurious oscillations node to node **??**. We use a SUPG strategy to stabilise the problem in the spatial dimensions. In the context of Petrov-Galerkin PGD, some modifications must be carried in order to prevent the dual problem to present spurious solutions in the time dimension.

In addition, if the problem presents a temporal dependency on the BC type it can be the case that boundary nodes that were initially free (natural BCs were enforced there) became prescribed (essential BCs) this induces spurious oscillations in the time dimension.

In this chapter we present the stabilisation of both phenomena using a SUPG strategy. However, the methodology can be generalised to any stabilisation method. For simplicity, we consider the formulation of homogeneous Dirichlet BCs on all the spatial boundary and the time interval as $T = (0,1)$.

Finally, note that the abstract formulation is not present in this chapter as it lacks of meaning. In any case, the methodology can be extended to the problem of parametric transient advection-diffusion problems.

5.1 Stabilisation of the spatial dimension

The solution of the spatial dimension presents spurious oscillations from node to node if the Péclet number is larger than 1 and Dirichlet BCs are enforced at the outflow generating a boundary layer. To stabilise the spatial dimension we use a SUPG formulation:

$$B^{SUPG_S}(u,v) = \sum_{e\in\mathcal{T}_h} \int_{\Omega_e\times T} \big(u_t + c\cdot\nabla u - \nabla\cdot(\nu\nabla u)\big)\tau_S(c\cdot\nabla v)\,d(\Omega\times T), \tag{5.1}$$

$$L^{SUPG_S}(v) = \sum_{e\in\mathcal{T}_h} \int_{\Omega_e\times T} f\tau_S(c\cdot\nabla v)\,d(\Omega\times T). \tag{5.2}$$

Where

$$\tau_S = \left(\frac{2c}{h} + \frac{4\nu}{h^2}\right)^{-1},$$

and h stands for the diameter of the element. The summation being over all the elements partitioning the domain.

Remark 5.1: The forms of the SUPG are integrated in time as we are using a global in time formulation of the advection-diffusion problem.

The weak (full order) stabilised problem is to find $u \in \mathcal{V}$ such that

$$(B + B^{SUPG_S})(u,v) = (L + L^{SUPG_S})(v) \quad \forall v \in \mathcal{V}. \tag{5.3}$$

Following the steps of chapter 4, the Petrov-Galerkin formulation is to define the dual problem as

$$(B + B^{SUPG_S})(v, \tilde{u}) = \langle u, v \rangle \quad \forall v \in \mathcal{V}. \tag{5.4}$$

However, with this approach the dual solution $\tilde{u}$ presents spurious oscillations in the temporal dimension. Alternatively, we define first both primal and dual problems and then stabilise separately each one. The stabilisation of the primal problem is identical as in the previous case. To stabilise the dual problem, we first recover the strong form of the problem.

$$B(v, \tilde{u}) = \langle u, v \rangle \quad \forall v \in \mathcal{V},$$
$$\int_{\Omega\times T} v_t\tilde{u} + (c\cdot\nabla v)\tilde{u} + \nu\nabla v\cdot\nabla\tilde{u}\, d(\Omega\times T) + \int_\Omega (\tilde{u}v)(0)\, d\Omega = \int_{\Omega\times T} uv\, d(\Omega\times T).$$

Applying the Gauss divergence theorem:

$$\int_{\Omega\times T} (\tilde{u}v)_t\, d(\Omega\times T) = \int_\Omega (\tilde{u}v)(1)\, d\Omega - \int_\Omega (\tilde{u}v)(0)\, d\Omega = \int_{\Omega\times T} \tilde{u}_t v + \tilde{u}v_t\, d(\Omega\times T) \Rightarrow$$
$$\int_{\Omega\times T} v_t\tilde{u}\, d(\Omega\times T) + \int_\Omega (\tilde{u}v)(0) d\Omega = \int_{\Omega\times T} -\tilde{u}_t v\, d(\Omega\times T) + \int_\Omega (\tilde{u}v)(1) d\Omega,$$

$$\int_{\Omega\times T} \nabla\cdot(c\,uv)\, d(\Omega\times T) = \int_{\partial\Omega\times T} c\cdot n\, uv\, d(\partial\Omega\times T) =$$
$$\int_{\Omega\times T} (\nabla\cdot c)vu + (c\cdot\nabla u)v + (c\cdot\nabla v)u\, d(\Omega\times T) \Rightarrow$$

$$\int_{\Omega\times T} (c\cdot\nabla v)u\, d(\Omega\times T) = \int_{\Omega\times T} (-c\cdot\nabla u)v\, d(\Omega\times T).$$

In the last step we have used the fact that $v = 0$ on $\partial\Omega$ and $\nabla\cdot c = 0$. Substituting in the dual weak problem:

$$\int_{\Omega\times T} -\tilde{u}_t v + (-c\cdot\nabla\tilde{u})v + \nu\nabla v\cdot\nabla\tilde{u}\, d(\Omega\times T) + \int_\Omega (\tilde{u}v)(1)\, d\Omega = \int_{\Omega\times T} uv\, d(\Omega\times T).$$

Remark 5.2: If the initial condition of the primal problem is imposed in weak sense, like in the present formulation, the initial condition of the dual problem is imposed identically (note that the initial condition is imposed at $t = 1$ as the problem is backward in time). Conversely, if one imposes the initial condition of the primal problem in strong sense, the bilinear form of the dual problem has the same term imposing weakly the initial condition at $t = 1$ but as the space $\mathcal{V}$ consist of functions such that $u(0) = 0$, the dual problem presents also an spurious final condition imposed strongly at $t = 0$. This can be interpreted as imposing an essential BC in the outflow of a pure advection problem (as is discussed in the next section). We guess that this would result in the presence of spurious oscillations in time as it is the case in the example showed in chapter 6.

The strong form of the problem is

$$\begin{aligned} -\tilde{u}_t + (-c\cdot\nabla\tilde{u}) - \nabla\cdot(\nu\nabla\tilde{u}) &= u \quad \text{in}\,\Omega\times T, \\ \tilde{u} &= 0 \quad \text{on}\,\partial\Omega\times T, \\ \tilde{u} &= 0 \quad \text{on}\,\Omega\times\{1\}. \end{aligned} \tag{5.5}$$

This problem is a backward in time advection-diffusion transient PDE. To apply a SUPG stabilisation, we first define a change of the time variable to recover a forward in time PDE.

$$\begin{aligned} \phi : T &\to \tilde{T} = (0,1), \\ t &\mapsto \tilde{t} = 1 - t. \end{aligned}$$

With this change of the time variable the PDE is

$$\begin{aligned} \tilde{u}_{\tilde{t}} + (-c \cdot \nabla \tilde{u}) - \nabla \cdot (\nabla \nu \tilde{u}) = u \quad & \text{in}\, \Omega \times \tilde{T}, \\ \tilde{u} = 0 \quad & \text{on}\, \partial\Omega \times \tilde{T}, \\ \tilde{u} = 0 \quad & \text{on}\, \Omega \times \{0\}. \end{aligned} \tag{5.6}$$

The SUPG stabilisation of this problem is:

$$\tilde{B}^{SUPG_S}(\tilde{u}, v) = \sum_{e \in \mathcal{T}_h} \int_{\Omega_e \times \tilde{T}} \big(\tilde{u}_{\tilde{t}} + (-c \cdot \nabla \tilde{u}) - \nabla \cdot (\nu \nabla \tilde{u})\big) \tau_S (-c \cdot \nabla v)\, d(\Omega \times \tilde{T}) =$$
$$\sum_{e \in \mathcal{T}_h} \int_{\Omega_e \times T} \big(\tilde{u}_t + (c \cdot \nabla \tilde{u}) + \nabla \cdot (\nu \nabla \tilde{u})\big) \tau_S (c \cdot \nabla v)\, d(\Omega \times T),$$

$$\tilde{L}^{SUPG_S}(v) = \sum_{e \in \mathcal{T}_h} -u \tau_S (c \cdot \nabla v)\, d(\Omega \times T).$$

The stabilised dual problem is:

$$(\tilde{B} + \tilde{B}^{SUPG_S})(\tilde{u}, v) = (\tilde{L} + \tilde{L}^{SUPG_S})(v) \quad \forall v \in \mathcal{V}. \tag{5.7}$$

Where $\tilde{B}(\tilde{u}, v) = B(v, \tilde{u})$ and $\tilde{L}(v) = \langle u, v \rangle$.

Remark 5.3: To implement this SUPG scheme, one shall add to the dual problem the same modes corresponding to the SUPG matrices of the primal problem without transposing them and changing the sign of the mode corresponding to the residual of the diffusion (if linear elements are used this mode is null). Note that if instead of using a SUPG stabilisation one uses a GLS stabilisation (and piecewise linear elements), the stabilisation matrices are symmetric and it makes no difference on first defining the dual problem and then stabilise or viceversa.

Remark 5.4: The term $\tilde{L}^{SUPG_S}$ under the Petrov-Galerkin PGD formulation becomes a coupled term between the primal and the dual problem.

5.2 Stabilisation of the temporal dimension

In the PGD context, we have used continuous elements to perform the time discretisation (see *Remark 2.3*). This is because at the fixed point algorithm of the rank-one problem, one must solve the global problem in the time dimension. This means that using discontinuous Galerkin or discretisations based on finite differences does not result in any save of computational power as in classical discretisations, where the time steps are computed sequentially.

Discretising the time using continuous shape functions makes the problem stable so we do not need to apply classical time stabilisation schemes. This is because of the global nature of the problem in time dimension. However, stability is lost if we let the system evolve and then impose some Dirichlet BC on some part of the domain. In this case, node to node oscillations appear in the temporal dimension in the time interval before the enforcement of the Dirichlet BC. This phenomenon can be understood in a simple way by considering the transient advection-diffusion equation as a kind of transport equation in the temporal dimension. In this section we develop this assertion and propose a stabilisation method to avoid the oscillations.

First, consider the PDE as an equation in the sense of distributions where the solution is in $\mathcal{H}_1(T; \mathcal{H}_1^D(\Omega)) \sim \mathcal{H}_1^D(\Omega) \otimes \mathcal{H}_1(T)$. That is, u and u_t are square integrable functions in time. For a.e. $t \in T$ such functions assign to t an element of $\mathcal{H}_1^D(\Omega)$.

The problem is to find a family of functions in $\mathcal{H}_1^D(\Omega)$ denoted by $\{u(t) : u(t) \in \mathcal{H}_1^D(\Omega),\, t \in T\}$ that fulfil the following PDE in the sense of distributions.

$$c \cdot \nabla u(t) - \nabla \cdot (\nu \nabla u(t)) + L(u(t)) = f \quad \text{on}\ \Omega \quad \forall t \in T.$$

Where $L(u(t)) = u_t(t)$ can be seen as a linear functional on $u(t)$ depending on functions of different times. This term does not affect the stability properties of the PDE. That is, L is similar to a reaction term. In fact, when solving the rank-one problem, in the spatial dimension, the equation to solve is a steady state advection-diffusion-reaction equation where the reaction term comes from the function L. At this point, one can stabilise the discretised form of the PDE using an standard SUPG scheme. The term L has no effect on the stabilisation more than its contribution to the residual of the equation. That is,

$$\int_{\Omega \times T} \mathcal{R}(u)\tau_S(c \cdot \nabla v)\, d(\Omega \times T).$$

This is the SUPG term added in the previous section.

Now we use the same argument but interchanging the spatial and temporal dimensions to stabilise the time oscillations. We consider u as a function of $H_1^D(\Omega; H_1(T)) \sim \mathcal{H}_1^D(\Omega) \otimes \mathcal{H}_1(T)$. The problem is now to find a family of functions in $\mathcal{H}_1(T)$, $\{u(x) : u(x) \in \mathcal{H}_1(T),\ x \in \Omega\}$. Such that they satisfy

$$u_t(x) + L(u(x)) = f \quad \text{on } T \quad \forall x \in \Omega.$$

Where the linear functional is $L(u(x)) = -\nabla \cdot (\nu \nabla u(x)) + c \cdot \nabla u(x)$. The problem is now a pure advection problem in the time dimension. The advection velocity is 1 in all the domain. For generality, suppose that the term u_t is replaced by $a \cdot u_t$; $a > 0$. With this, the advection velocity is a. Now it is clear why if one prescribes the value of $u(x)$ after some interval of time (i.e. the outflow) oscillations appear. These oscillations can be stabilised using a SUPG scheme. That is,

$$\int_{\Omega \times T} \mathcal{R}(u)\tau_T(a \cdot v_t)\, d(\Omega \times T).$$

Where $\tau_T = \frac{h}{2a}$.

Remark 5.5: With this stabilisation technique, we have modified the problem so now is second order in time. This means that the system evolves in the same manner than a damped harmonic oscillator. However, it is expected that the effect of the stabilisation parameter is so small than this results in a highly overdamped oscillator. Note also that the problem has become elliptic, i.e. the future affects the past. This is the case, for instance, when some Dirichlet values are prescribed in the future. The solution evolves to agree with the prescribed value. This evolution starts some instant of time before the actual values are prescribed. However, as the stabilisation is consistent we have ensured that both phenomena are negligible if the discretisation (in time) is fine enough.

Now we can proceed to stabilise the problem in the same way we did in the previous section. That is, we stabilise in a separate way the primal and the dual problem. For the primal problem, the stabilisation is:

$$\begin{aligned} B^{SUPG_T}(u, v) &= \sum_{e \in \mathcal{T}_h} \int_{\Omega_e \times T} (u_t + c \cdot \nabla u - \nabla \cdot (\nu \nabla u))\tau_T v_t\, d(\Omega \times T), \\ L^{SUPG_T}(v) &= \sum_{e \in \mathcal{T}_h} \int_{\Omega_e \times T} f \tau_T v_t\, d(\Omega \times T). \end{aligned}$$

The stabilisation of the dual problem can be deduced applying the same steps that we did in the previous section. The SUPG terms are:

$$\begin{aligned} \tilde{B}^{SUPG_T}(\tilde{u}, v) &= \sum_{e \in \mathcal{T}_h} \int_{\Omega_e \times T} (\tilde{u}_t + c \cdot \nabla \tilde{u} + \nabla \cdot (\nu \nabla \tilde{u}))\tau_T v_t\, d(\Omega \times T), \\ \tilde{L}^{SUPG_T}(v) &= \sum_{e \in \mathcal{T}_h} \int_{\Omega_e \times T} -u \tau_T v_t\, d(\Omega \times T). \end{aligned}$$

The structure of the added stabilising terms is exactly the same that the one obtained in the spatial case. The stabilised problems in space and time are

$$(B + B^{SUPG_S} + B^{SUPG_T})(u, v) = (L + L^{SUPG_S} + L^{SUPG_T})(v) \quad \forall v \in \mathcal{V}, \tag{5.8}$$

$$(\tilde{B} + \tilde{B}^{SUPG_S} + \tilde{B}^{SUPG_T})(\tilde{u}, v) = (\tilde{L} + \tilde{L}^{SUPG_S} + \tilde{L}^{SUPG_T})(v) \quad \forall v \in \mathcal{V}. \tag{5.9}$$

One can then substitute the original bilinear and linear form by their stabilised versions in the PGD algorithms.

5.3 Stabilisation of the rank-one problem

Although the stabilisation method defined is satisfactory for the full order problem, we make a point in the case of the separated form. Let us assume that all parameters of the PDE and the stabilisation parameter τ_S are time independent except the advection velocity which is a rank-one function $c = c_S \otimes c_T$. The stabilised (in space) separated bilinear form of the rank-one problem for the Galerkin PGD formulation is

$$B(\omega\otimes\lambda, \omega^*\otimes\lambda^*) = \int_\Omega \omega\omega^* \, d\Omega \int_T \lambda_t\lambda^* \, dT + \int_\Omega (c_S\cdot\nabla\omega)\omega^* \, d\Omega \int_T c_T\lambda\lambda^* \, dT + \int_\Omega \nu\nabla\omega\cdot\nabla\omega^* \, d\Omega \int_T \lambda\lambda^* \, dT +$$
$$\int_\Omega \omega\omega^* \, d\Omega \cdot \lambda(0)\lambda^*(0) + \sum_{e\in\mathcal{T}_h} \left(\int_{\Omega_e} \omega\tau_S(c_S\cdot\nabla\omega^*) \, d\Omega \int_T c_T\lambda_t\lambda^* \, dT + \right.$$
$$\left. \int_{\Omega_e} (c_S\cdot\nabla\omega)\tau_S(c_S\cdot\nabla\omega^*) \, d\Omega \int_T c_T^2\lambda\lambda^* \, dT + \int_{\Omega_e} \nabla\cdot(\nu\nabla\omega)\tau_S(c_S\cdot\nabla\omega^*) \, d\Omega \int_T c_T\lambda\lambda^* \, dT \right).$$

The term corresponding to the advection is

$$\int_\Omega (c_S\cdot\nabla\omega)\omega^* \, d\Omega \int_T c_T\lambda\lambda^* \, dT.$$

The stabilisation term of the SUPG is

$$\sum_{e\in\mathcal{T}_h} \left(\int_{\Omega_e} (c_S\cdot\nabla\omega)\tau_S(c_S\cdot\nabla\omega^*) \, d\Omega \int_T c_T^2\lambda\lambda^* \, dT \right).$$

Solving the rank-one problem, in principle, one could find a time function λ that is solution of the problem such that the ratio

$$\frac{\int_T c_T^2\lambda^2 \, dT}{\int_T c_T\lambda^2 \, dT} \leqslant \epsilon,$$

ϵ being an arbitrary small constant.

This argument shows that although the full order problem is stable, one can find spurious oscillations in the spatial dimension of the separated approximation due to lack of stabilisation. In case this phenomenon arises, we expect that, as the full order problem is stable, the modes would have the tendency to compensate the error induced by previous unstable modes.

An alternative possible solution is to stabilise the full order problem and use the stabilised form only to compute the contribution of the previous modes. Then, when solving the rank-one problem, at the step of solving the spatial functions one gets an advection-diffusion problem where every term is weighted by a factor depending on the time function λ and the term involving the time evolution becomes a reaction term. Then, one can stabilise this problem and avoid in that way the spatial oscillations. The stabilisation operators should not be computed every time but only be weighted by the corresponding factor and summed.

Using this method, the solution of the rank-one problem would not be consistent in the sense that the stabilisation applied to their spatial functions would not be the same as the one applied to the full order problem. However, the difference of the solution due to such inconsistency is expected to be small. The computation of the contribution of that mode to the residual would be computed consistently for future modes so that the errors due to that inconsistency should be continuously corrected. The main drawback of this method is that it is highly code intrusive.

The question that arises using this method is if the resulting advection-diffusion-reaction steady state PDE is well posed. For the Galerkin method the answer is positive, even assuming parameters of arbitrary rank. For if $\lambda \in \mathcal{V}_T$ is an arbitrary function, the problem in the spatial dimension is

$$\int_\Omega \omega\omega^* \, d\Omega \int_T \lambda_t \lambda \, dT + \sum_{m\in M_C} \int_\Omega (c_S^m \cdot \nabla\omega)\omega^* \, d\Omega \int_T c_T^m \lambda^2 \, dT + \sum_{m\in M_\nu} \int_\Omega \nu_S^m \nabla\omega \cdot \nabla\omega^* \, d\Omega \int_T \nu_T^m \lambda^2 \, dT = \int_\Omega f_S \omega^* \, d\Omega \int_T f_T \lambda \, dT \quad \forall \omega^* \mathcal{V}_S. \tag{5.10}$$

Where $c = \sum_{m\in M_C} c_S^m \otimes c_T^m, \ \nabla \cdot c_S^m = 0, \ \nu = \sum_{m\in M_\nu} \nu_S^m \otimes \nu_T^m, \ \nu_S^m, \nu_T^m \geqslant 0$ and the force term has no interest for us. This is the weak form of a steady state advection-diffusion-reaction PDE.

$$\begin{aligned} c^* \cdot \nabla\omega - \nabla \cdot (\nu^* \nabla\omega) + \sigma^*\omega &= f \text{ in } \Omega, \\ \omega &= 0 \text{ on } \partial\Omega. \end{aligned} \tag{5.11}$$

Where

$$\begin{aligned} \sigma^* &= \int_T \lambda_t \lambda dT = \int_T \frac{1}{2}(\lambda^2)_t dT = \frac{1}{2}\big(\lambda^2(1) - \lambda^2(0)\big) = \frac{1}{2}\lambda^2(1) \geqslant 0, \\ c^* &= \sum_{m\in M_C} c_S^m \int_T c_T^m \lambda^2 dT, \ \nabla \cdot c^* = 0, \\ \nu^* &= \sum_{m\in M_C} \nu_S^m \int_T \nu_T^m \lambda^2 dT \geqslant 0. \end{aligned}$$

As $\sigma^*, \nu^* \geqslant 0$, $\nabla \cdot c^* = 0$ the problem is well posed and a SUPG scheme can be used to stabilise the problem. This introduces a nonlinearity with respect to λ but without any increase in the computational cost more than computing the stabilisation matrix. However, in the Petrov-Galerkin scheme the positiveness of σ^* and ν^* cannot be assured by this argument meaning that the problem may be ill-posed. We note as future work to investigate the stabilisation of the Petrov-Galerkin PGD scheme.

For the stabilisation in the time dimension, a similar reasoning can be applied by interchanging the role of the spatial and temporal functions.

5.4 Numerical example

To illustrate the capabilities of the SUPG stabilisation method, we provide a numerical example. The problem to solve is the same advection-diffusion PDE than in previous examples

$$\begin{aligned} u_t + \nabla \cdot (cu - \nu \nabla u) &= 0 \quad \text{in } \Omega \times T, \\ u &= 0 \quad \text{on } \partial\Omega, \\ u &= u_0 \quad \text{on } \Omega \times \{0\}. \end{aligned} \tag{5.12}$$

We modify the parameters so that the problem becomes advection dominated and the solution presents a boundary layer. For that we have chosen $c = (cos(2\pi t), 0) = (1, 0) \otimes cos(2\pi t)$ and $u_0 = \frac{1}{16}x(x-1)y(y-1)$. For the discretisation we use a regular mesh of 40×40 triangular elements in space and a regular mesh of 100 elements in space. This gives a diameter $h = \sqrt{2}\ 40$. The Péclet number is then

$$Pe = \frac{|c|h}{2\nu} = \frac{1}{\sqrt{2}\ 80\nu}$$

Where the reference magnitude of the velocity has been taken as 1, the maximum over the time interval. In order that the problem becomes advection dominated the diffusivity must satisfy $\nu < (\sqrt{2}\ 80)^{-1}$. We chose the extreme case of $\nu = 0$. That is, the problem is of pure transport. The Dirichlet BC at the outflow is spurious but we maintain it in order to see the performance of the SUPG stabilisation.

The procedure is the same than for the previous examples. First we compute the reference solution to be able to compute the error of the PGD approximation. Then we compute the Petrov-Galerkin operators (cf. chapter 4). We then add the stabilisation terms. Here we have stabilised only the spatial dimension. See next chapter for an example with stabilisation in time. Finally, we sequentially constructed the approximation by solving the corresponding rank-one problems.

The standard fixed point lacks of convergence in this problem, so we used an alternative method. This method consists in relaxing the iterations of the rank-one problem and using a technique of vanishing diffusivity to obtain a sequence of approximations of the problem. Each of those approximations are obtained as the solution of the problem with the addition of fictitious diffusivity. The first of the solutions is obtained with a large amount of diffusivity and is used as initial point to compute the second solution (with a lower amount of fictitious diffusivity). We proceed in that way until reaching a zero diffusivity. In all rank-one problems with different amount of artificial diffusivity, the iterations are sub-relaxed. In that way the method converges. See chapter 7 for more details.

We have computed the PGD approximation using the Petrov-Galerkin formulation up to 500 modes. In figure 5.1 we present the ten first spatial and temporal modes for the primal and dual solution and the reconstruction of the solution by adding the 500 modes (which is identical to solution of the full order problem by simple inspection).

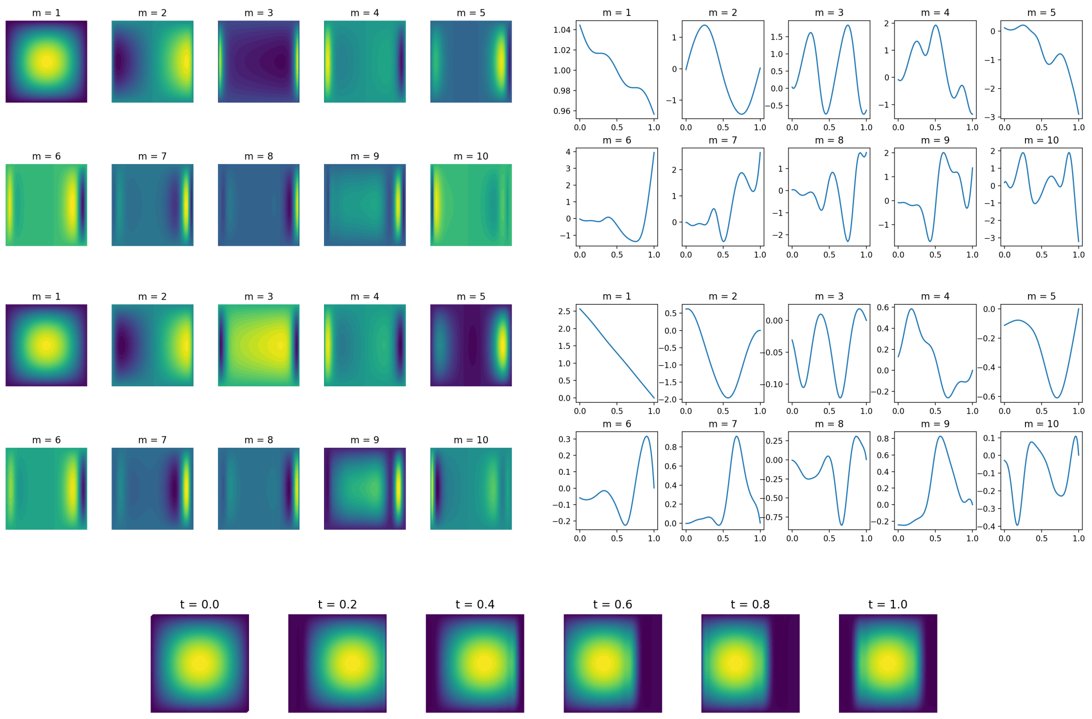

Figure 5.1 Petrov-Galerkin approximation for the stabilised pure advection problem

In the next figure we show the convergence of the PGD approximation.

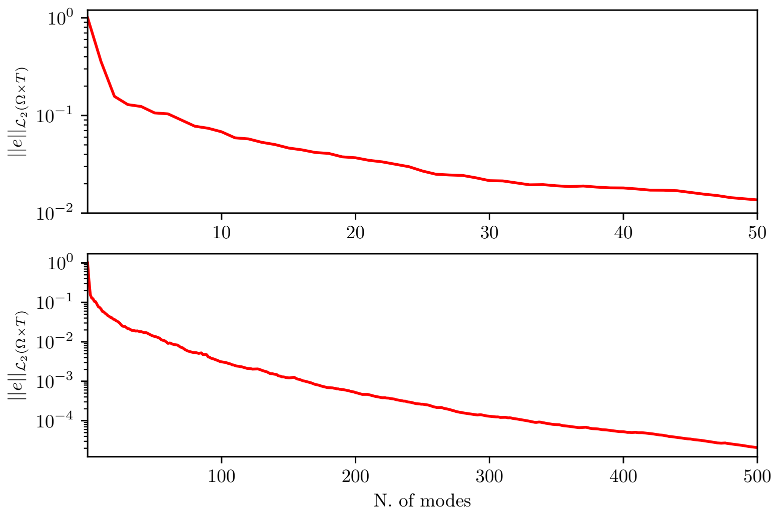

Figure 5.2 Error convergence

We see that with the two first modes the error is quickly reduced to approximately 10^{-1}. This is because the first two modes represent the main part of the solution in rough outlines (as can be seen in figure 5.1). The first mode is approximately constant in time and represents the central part of the solution, i.e. the initial paraboloid. The second mode adds a negative term to the left side of the domain and positive to the right side at about $t = 0.25$ as represents that the paraboloid has moved to the right and the opposite holds at $t = 0.75$, when the paraboloid has moved to the left. The rest of the terms continue improving the approximation of the solution but as the main threats of the solution are already computed, the convergence is slower. This explains the sudden drop of the error in the first two modes and then the convergence of the solution at a more or less constant order.

Finally, we show the modal amplitude of each computed mode and the ratio of the error with the modal amplitude.

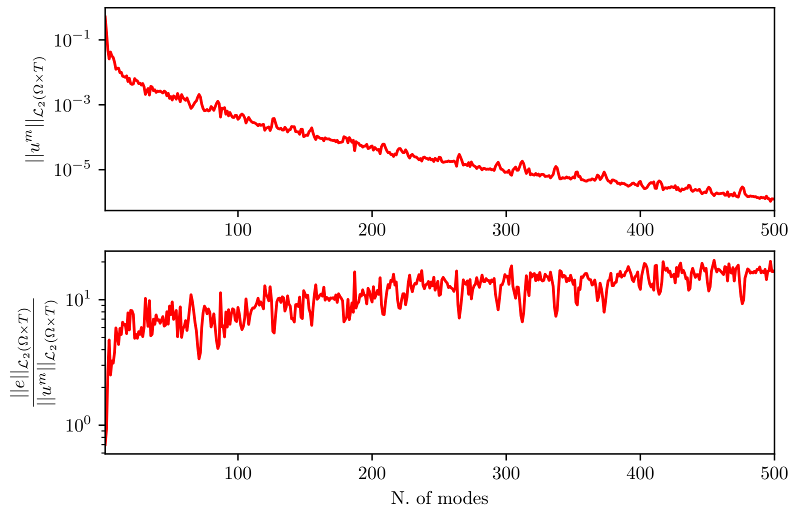

Figure 5.3 Modal amplitude and error estimator accuracy

As in previous Petrov-Galerkin examples, the modal amplitude stands as a good error estimator. The ratio of the error and the modal amplitude is roughly constant around 10.

5.4.1 A diverging example

In this subsection we show an example of problem such that the approximation obtained with the Petrov-Galerkin PGD method is diverging. The interesting phenomenon is that the divergence arises although every rank-one problem is computed correctly (of course, up to a small numerical tolerance). Such example is identical to the previous one with the magnitude of the advection velocity scaled by a factor of π. This is the smallest value of the velocity such that all points of the domain are displaced at some instant of time outside the domain. The solution at $t = 1$ should be $u(1) = 0$. However, we expect some diffusive effect arising from the SUPG stabilisation. The solution of the full order problem and the reconstruction of the PGD solution with different number of modes are plotted in figures 5.4 to 5.7. In figure 5.8 it is plotted the error and modal amplitude up to 50 modes of the PGD solution.

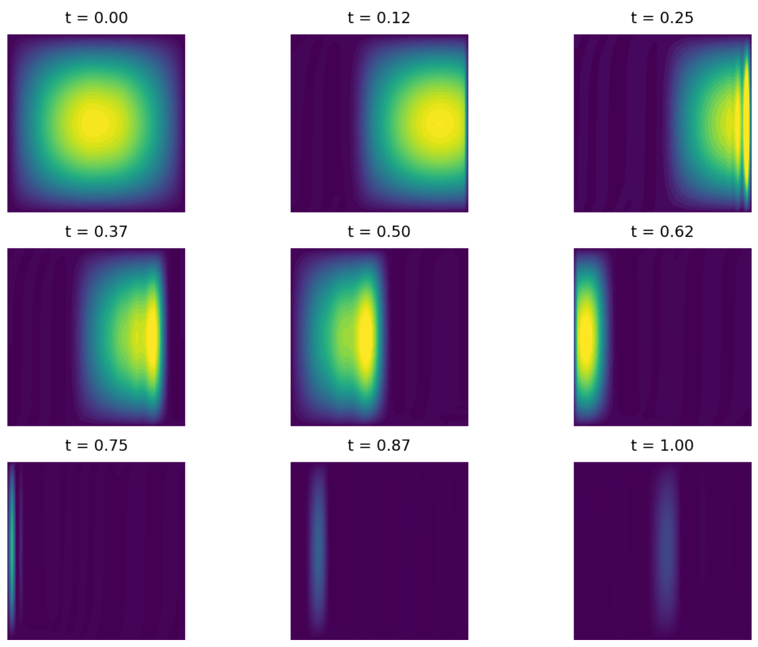

Figure 5.4 Full order solution for the diverging example

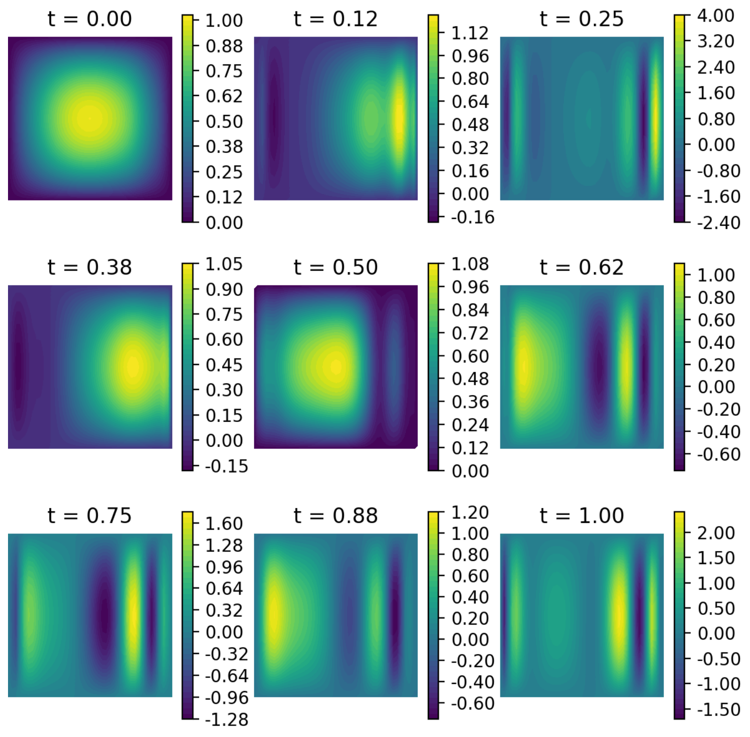

Figure 5.5 Reconstructed solution of the diverging example with 10 modes

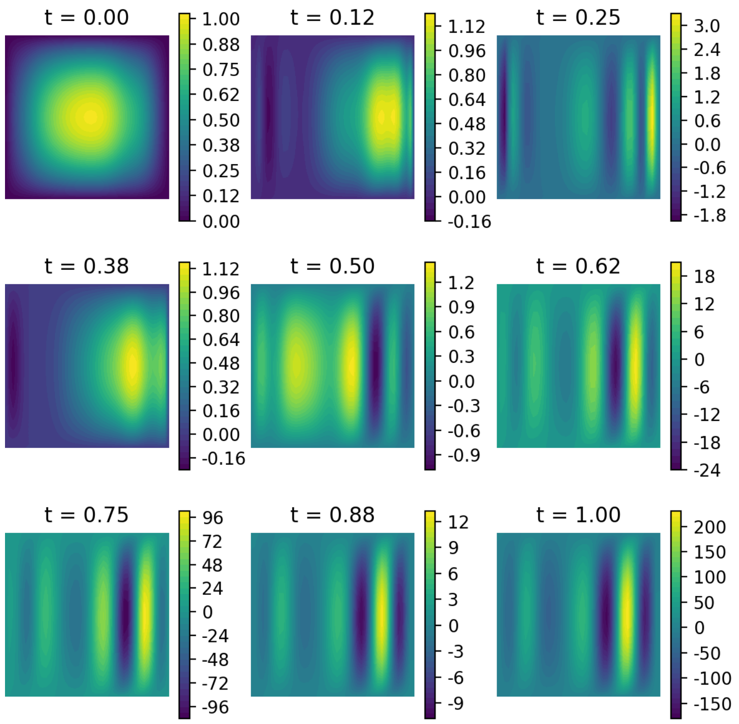

Figure 5.6 Reconstructed solution of the diverging example with 25 modes

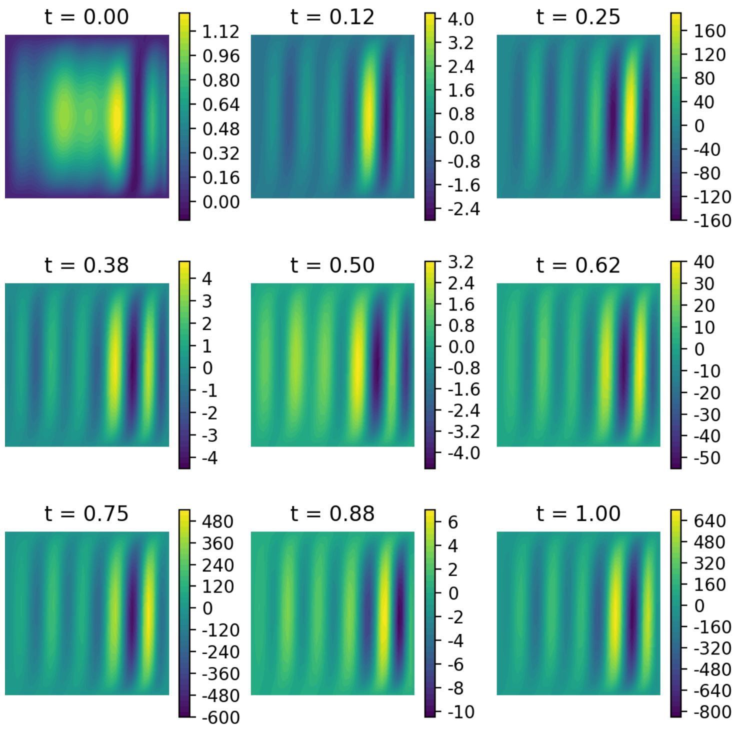

Figure 5.7 Reconstructed solution of the diverging example with 50 modes

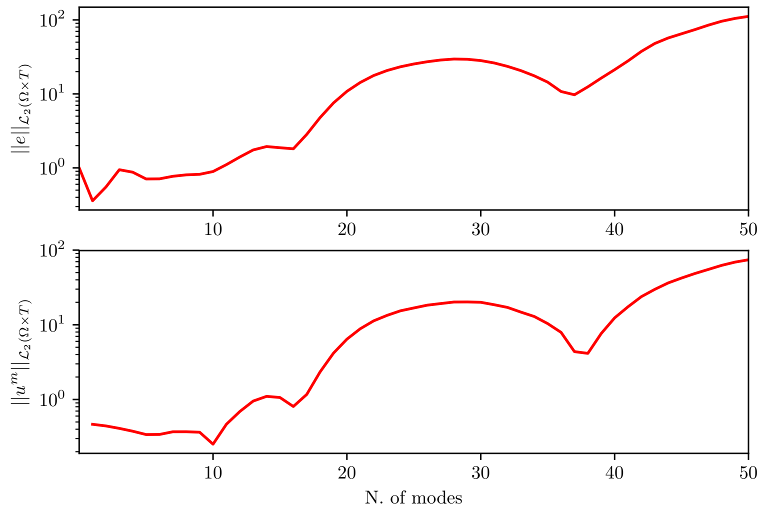

Figure 5.8 Error and modal amplitude for the diverging example

5.5 Concluding remarks

In this chapter we have shown successfully how to stabilise a transient advection-diffusion problem in space and time using a continuous Galerkin method (the full order problem). We have also shown how to stabilise the dual problem defined in the Petrov-Galerkin PGD method.

However, we have succeeded only partially in the PGD implementation of the stabilisation, as noted in section 5.3. The problem is that although, the PGD algorithm converges to the full order solution, the rank-one problems may not be stabilised correctly. We guess that this is the reason why the convergence properties of the rank-one problems are soo poor. Although in chapter 7 we explain several algorithms used successfully to obtain the solution of the rank-one problem, we note as future work to study the alternative method of stabilising each rank-one iteration individually in the case of the Galerkin PGD algorithm, and to investigate a possible extension to the Petrov-Galerkin formulation.

Finally, we have shown with the diverging example a case of a diverging PGD algorithm. In the example showed, one may chose an alternative approach to avoid the divergence. Divide the time interval in smaller intervals (in this case probably two are enough) and solve sequentially each of those, using as initial condition the final condition of the previous interval.

Chapter 6

Temporal/parametric dependence of the Boundary Conditions type

In the context of the advection-diffusion problem, in engineering applications is very common to prescribe Dirichlet BCs on the inflow part of the boundary while of Neumann type on the rest of the boundary. As the flow velocity may vary on time and/or on additional parameters of the PDE, the essential BCs cannot be prescribed in strong form. That is, one cannot decompose the space of functions as a tensor product of a spatial functions space that vanish on the Dirichlet Boundary part and spaces representing the time and/or parameters of the PDEs. Instead, we prescribe the essential BCs in a weak way, in particular, we use Nitsche's method. This results in some additional terms in the forms defining the problem.

$$B(u,v) = \int_{\Omega\times T} u_t v + (-cu+\nu\nabla u)\cdot\nabla v \, d(\Omega\times T) + \int_{\partial\Omega\times T} i_D(cu-\nu\nabla u)\cdot n\, v + i_D(cv-\nu\nabla v)\cdot n\, u \, d(\partial\Omega\times T) + \int_{\partial\Omega\times T} i_D\beta uv \, d(\partial\Omega\times T) + \int_{\Omega}(uv)(0)\, d\Omega, \quad (6.1)$$

$$L(v) = \int_{\Omega\times T} fv \, d(\Omega\times T) + \int_{\partial\Omega\times T} i_D(cu_D-\nu\nabla u_D)\cdot n\, v + i_D(cv-\nu\nabla v)\cdot n\, u_D \, d(\partial\Omega\times T) + \int_{\partial\Omega\times T} i_D\beta u_D v \, d(\partial\Omega\times T) + \int_{\partial\Omega\times T} i_N h v \, d(\partial\Omega\times T) + \int_{\Omega} u_0 v(0)\, d\Omega. \quad (6.2)$$

Where β is the parameter to enforce the essential BC and i_D and i_N are the indicator function of Dirichlet and Neumann BCs, i.e. the image of $(x,t) \in \partial\Omega\times T$ under i_D (or i_N) is 1 if at that point Dirichlet (or Neumann) BCs are enforced, and 0 otherwise. The methodology can be extended to any number of different BC types with the condition that the set of all indicator functions must form a partition of unity on $\partial\Omega\times T$. Note that in order to preserve the separability of the forms, the indicator functions must be separable. If a separated representation is not possible, they can be approximated using a tensor separation strategy. For the separable approximation of input parameters we refer, again, to [5]. The separation representation ideally should preserve two properties:

– The approximation should be a partition of unity.

– The representation should be a function with image in $\{0,1\}$.

The first property is easily achieved by defining $i_N = 1 - i_D$. The second one is more difficult to fulfil. We note as future work to study the effect of the approximation of the indicator functions in the solution of the full-order problem.

Remark 6.1: In the full order formulation one can avoid the use of the indicator functions by changing the domain of integration of the terms involving the BC by the set on which the corresponding BC are active.

The two alternatives are equivalent. When writing the separated formulation, is more convenient to use the indicator functions.

Remark 6.2: Although we have restricted the methodology to the particular of the advection-diffusion problem, it is straightforward to extend it to different problems and number of parameters.

6.1 Numerical examples

6.1.1 Example on the unit square

To illustrate the devised formulation, we use a simple advection-diffusion example in the unit square. For that, we chose a rank-one uniform in space time dependent advection velocity $c = (\cos(\pi t), 0) = (1, 0) \otimes \cos(\pi t)$. With this choice of the velocity, there is no advective flux across the upper and lower boundaries and Neumann BC are prescribed there. The boundary conditions prescribed on the sides of the square depend on time. That is, if the advective flow across them corresponds to the one of an inflow Dirichlet BCs are prescribed, and Neumann otherwise. The strong form of the problem is

$$\begin{aligned}
u_t + \nabla \cdot (cu - \nu \nabla u) &= 0 && \text{in } \Omega \times T, \\
(cu - \nu \nabla u) \cdot n &= 0 && \text{on } \Gamma_0 \times T, \\
u &= u_D^L && \text{on } \Gamma_L \times (0, 1/2), \\
(cu - \nu \nabla u) \cdot n &= 0 && \text{on } \Gamma_L \times [1/2, 1), \\
u &= u_D^R && \text{on } \Gamma_R \times (1/2, 1), \\
(cu - \nu \nabla u) \cdot n &= 0 && \text{on } \Gamma_R \times (0, 1/2], \\
u &= u_0 && \text{on } \Omega \times \{0\}.
\end{aligned} \tag{6.3}$$

where

$$\begin{aligned}
\Omega &= (0, 1)^2, \\
T &= (0, 1), \\
\Gamma_0 &= (0, 1) \times \{0, 1\}, \\
\Gamma_L &= \{0\} \times (0, 1), \\
\Gamma_R &= \{1\} \times (0, 1) \text{ and}
\end{aligned}$$

n stands for the outward normal unit vector. The chose parameters are

$$\begin{aligned}
c &= (\cos(\pi t), 0), \\
\nu &= 10^{-2}, \\
u_0 &= x, \\
u_D^L &= 2t, \\
u_D^R &= 2t - 1.
\end{aligned}$$

The weak formulation of the problem is

$$B(u, v) = L(v) \ \forall v \in \mathcal{V}, \tag{6.4}$$

where

$$\begin{aligned}B(u,v) = \int_{\Omega\times T} u_t v + (-cu+\nu\nabla u)\cdot\nabla v \; d(\Omega\times T) + \int_{\Omega} (uv)(0)\; d\Omega + \\ \sum_{e\in\mathcal{T}_h}\int_{\Omega_e\times T}\big(u_t + \nabla\cdot(cu-\nu\nabla u)\big)\tau_S(c\cdot\nabla v)d(\Omega\times T) + \int_{\Omega_e\times T}\big(u_t+\nabla\cdot(cu-\nu\nabla u)\big)\tau_T(v_t)d(\Omega\times T) + \\ \int_{\Gamma_L\times T} i_{\Gamma_L}\big((cu-\nu\nabla u)\cdot n\, v + (cv-\nu\nabla v)\cdot n\, u\big)\; d(\partial\Omega\times T) + \int_{\Gamma_R\times T} i_{\Gamma_R}\big((cu-\nu\nabla u)\cdot n\, v + (cv-\nu\nabla v)\cdot n\, u\big)\; d(\partial\Omega\times T) + \\ \int_{\Gamma_L\times T} i_{\Gamma_L}\beta uv\; d(\partial\Omega\times T) + \int_{\Gamma_R\times T} i_{\Gamma_R}\beta uv\; d(\partial\Omega\times T),\end{aligned} \tag{6.5}$$

$$\begin{aligned}L(v) = \int_{\Omega} u_0 v(0)\; d(\Omega) + \int_{\Gamma_L\times T} i_{\Gamma_L}(cv-\nu\nabla v)\cdot n\; u_D^L\; d(\partial\Omega\times T) + \int_{\Gamma_R\times T} i_{\Gamma_R}(cv-\nu\nabla v)\cdot n\; u_D^R\; d(\partial\Omega\times T) + \\ \int_{\Gamma_L\times T}\beta i_{\Gamma_L} u_D^L v\; d(\partial\Omega\times T) + \int_{\Gamma_R\times T}\beta i_{\Gamma_R} u_D^R v\; d(\partial\Omega\times T).\end{aligned} \tag{6.6}$$

Note that we have included the terms of the SUPG stabilisation in space and time. The stabilisation parameters τ_S and τ_T are defined in chapter 5. The indicator functions are defined as $i_{\Gamma_L}(x,t) = \chi_{(0,1/2)}(t)$, $i_{\Gamma_R}(x,t) = \chi_{(1/2,1)}(t)$. Where χ_S denotes the indicator function of an arbitrary set S. The parameter to impose the Dirichlet BCs is defined as $\beta = 10^4/h$ where h represents the diameter of each element. Noting that all parameters appearing in the weak formulation are rank-one functions so the separation of the forms is straightforward.

The domain is discretised with the same discretisation used in previous examples: a regular triangular mesh of 40×40 and 100 regular elements in time. In figure 6.1, it is plotted the solution of the full order problem (6.4). In figure 6.2, it is plotted the solution at point $(1, 1/2) \in \Gamma_R$ as a function of time. It is noted that, as stated in *Remark 5.5*, the Dirichlet BC actually affects the solution some instant of time before $t = 1/2$ where it is enforced.

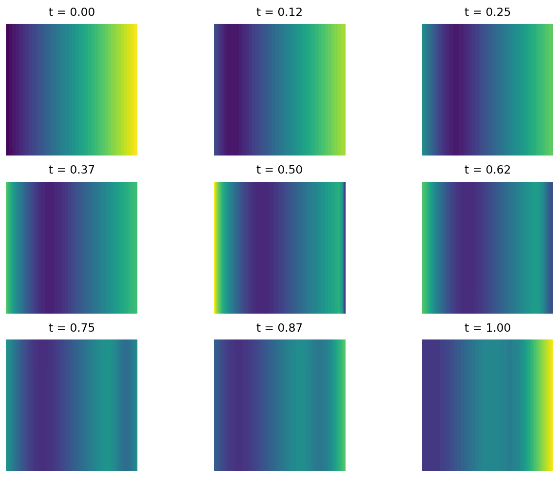

Figure 6.1 Full order solution of the problem

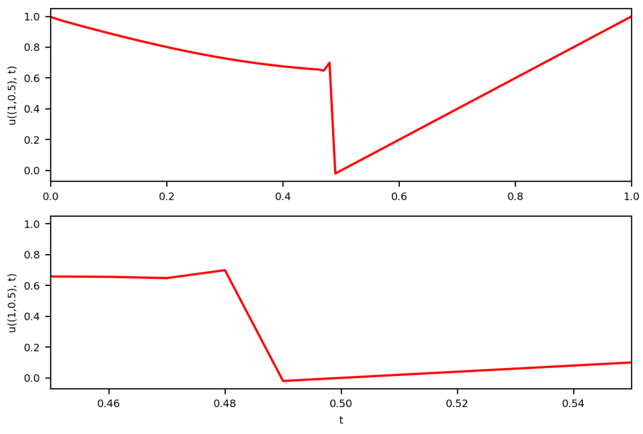

Figure 6.2 Solution at the right boundary of the domain as function of time

To solve the rank-one problem, the fixed-point classical algorithm lacks of convergence. For this reason, we have adopted two different strategies that lead to different results. Both algorithms are described in detail in chapter 7.

6.1.2 Vanishing diffusivity strategy

The first method used, is the vanishing diffusivity method. It is the same used in solving the examples of chapter 5. In particular, it also uses the sub-relaxation. The algorithm solved correctly every rank-one problem.

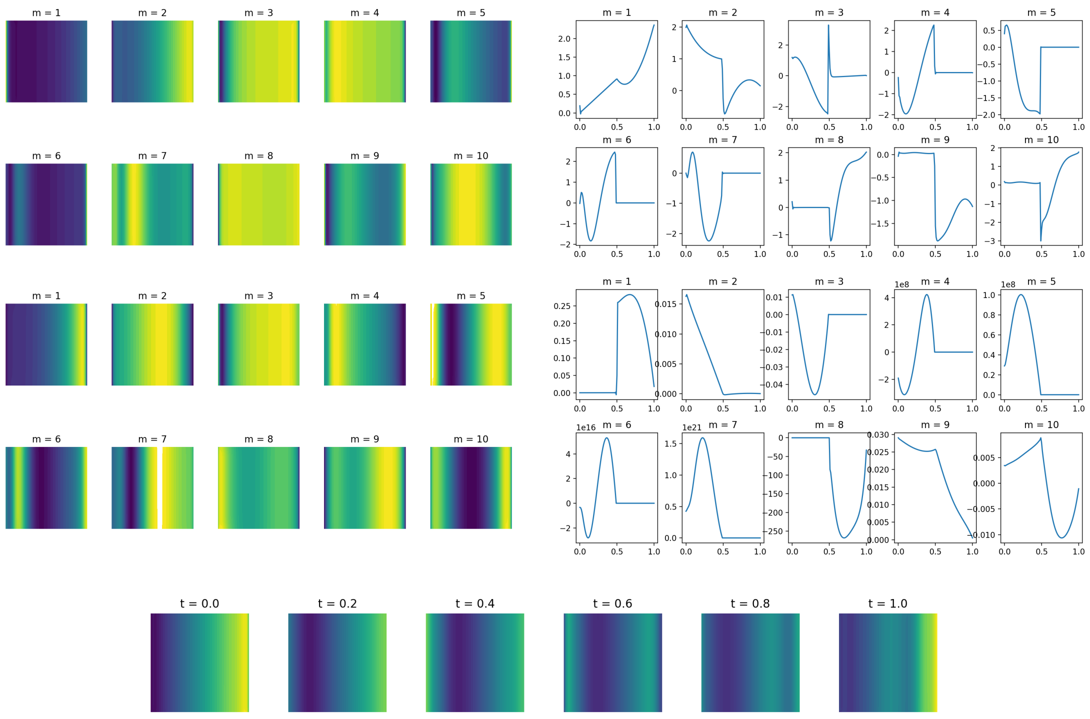

Figure 6.3 PGD approximation obtained by vanishing diffusion

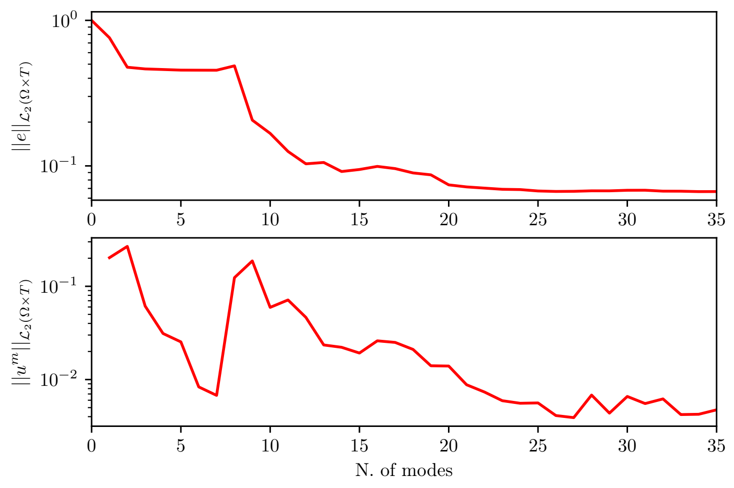

Figure 6.4 Error convergence and modal amplitude obtained by vanishing diffusion

As can be seen in figure 6.4, the error of the solution stagnates around 10^{-1}. However, using the sub-relaxed fixed point algorithm, the solution obtained does not present this behaviour. This phenomenon arises because, at least for some modes, the solution of the rank-one problem is not unique. If the solution obtained at each mode is the corresponding to the vanishing diffusivity method, the PGD approximation does not converge to the full-order solution but to a different fixed-point. However, in some cases, if the sequence of the values of the artificial diffusivity is to coarse, the rank-one function used as initial point for following lower artificial diffusivity does not converge to the solution that corresponds to the vanishing diffusivity scheme. In this case, the error is reduced below the value of stagnation. The solution then stagnates until the same phenomenon occurs.

This phenomenon occurs by chance and is hard to reproduce deliberately. For this reason, we show in figure 6.5 the same plot of convergence obtained for a coarser discretisation and different parameters. This solution was obtained after the debugging step before using the actual parameters. The problem is the same with a coarser discretisation and non-homogeneous Neumann BC at the outflow of the sides of the boundary. The convergence properties are essentially the same.

In the figure it is clearly seen that the solution stagnates until mode 53 where the vanishing diffusivity strategy failed to converge to the corresponding rank-one solution and so the error is decreased suddenly to stagnate for the following modes. Note that at mode 95 the same phenomenon appears.

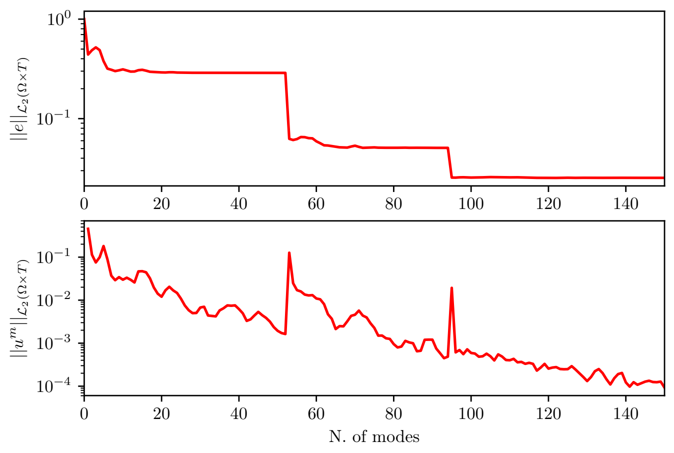

Figure 6.5 Error convergence and modal amplitude obtained by vanishing diffusion (coarser discretisation)

6.1.3 Sub-relaxation strategy

This method solves the problem of the convergence of the PGD solution towards a different point from the reference solution. It consists in the standard fixed-point algorithm with a sub-relaxation step after computing each sectional function. The main drawback of this method is that the stability of the system is very susceptible to the relaxation parameter. The parameter has been adjusted manually.

The results and convergence are shown in the following plots. In particular, note that the modes obtained are different than the ones obtained with the vanishing diffusivity strategy. Also note that, although the error stagnates at some point, it eventually breaks the tendency and then, it continues converging.

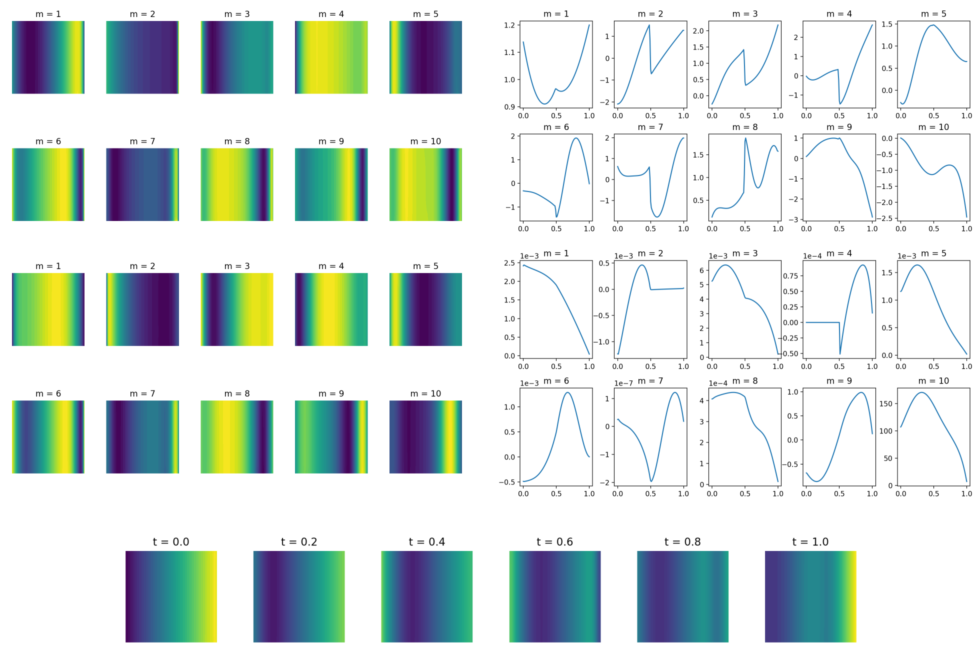

Figure 6.6 PGD approximation obtained by fixed point sub-relaxation

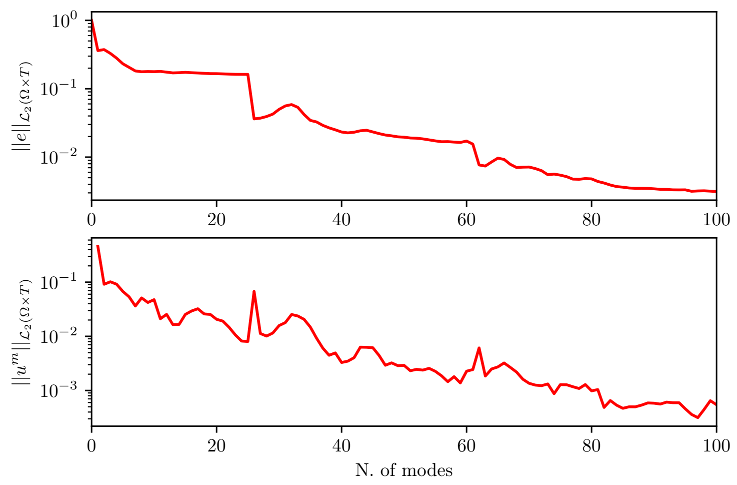

Figure 6.7 Error convergence and modal amplitude obtained by fixed point sub-relaxation

6.1.4 Example on the unit circle

In the last example, one could have divided the time interval as $T = (0, 1/2] \cup (1/2, 1)$ and solved each interval individually with constant BCs type. In other words, the indicator function i_D is a finite (2) rank function. In some cases this may be not possible. In this example we solve a the problem in the unit circle where a uniform advection velocity is rotating with constant angular velocity as time evolves. In this case, the indicator function is not of finite rank. The problem statement is

$$\begin{aligned} u_t + \nabla \cdot (cu - \nu \nabla u) &= f \quad \text{in } \Omega \times T, \\ u &= 0 \quad \text{on } \Gamma_D, \\ (cu - \nu \nabla u) \cdot n &= 0 \quad \text{on } \Gamma_N, \\ u &= 0 \quad \text{on } \Omega \times \{0\}, \end{aligned} \tag{6.7}$$

where

$$\begin{aligned} \Omega &= \{(x, y) \in \mathbb{R}^2 : x^2 + y^2 < 1\}, \\ T &= (0, 1), \\ \Gamma_D &= \{(x, t) \in \partial\Omega \times T : c \cdot n < 0\}, \\ \Gamma_L &= (\partial\Omega \times T) \backslash \Gamma_D, \text{ and} \end{aligned}$$

n stands for the outward normal unit vector. The chosen parameters are

$$\begin{aligned} c &= \big(\cos(\pi t), \sin(\pi t)\big), \\ \nu &= 10^{-2}, \\ f &= 1. \end{aligned}$$

With the chosen advection velocity, the explicit boundary splitting is

$$\begin{aligned} \Gamma_D &= \{\big((x, y), t\big) \in \partial\Omega \times T : (x, y) \cdot \big(\cos(\pi t), \sin(\pi t)\big) < 0\}, \\ \Gamma_N &= \{\big((x, y), t\big) \in \partial\Omega \times T : (x, y) \cdot \big(\cos(\pi t), \sin(\pi t)\big) \geqslant 0\}. \end{aligned}$$

The bilinear and linear forms of the weak problem are

$$B(u,v) = \int_{\Omega\times T} u_t v + (-cu + \nu\nabla u)\cdot\nabla v \; d(\Omega\times T) + \int_{\Omega} (uv)(0)\; d\Omega + \\ \int_{\partial\Omega\times T} i_D\big((cu-\nu\nabla u)\cdot n\; v + (cv - \nu\nabla v)\cdot n\; u\big)\; d(\partial\Omega\times T) + \int_{\partial\Omega\times T} i_D \beta uv\; d(\partial\Omega\times T), \quad (6.8)$$

$$L(v) = \int_{\Omega\times T} fv\; d(\Omega\times T). \quad (6.9)$$

Note that the terms involving the stabilisation have been omitted for the sake of clarity.

In order to separate the bilinear form as a finite sum of rank-one operators we have to approximate the non-separable function i_D by a finite rank representation, for a complete treatment of the approximation step we refer to [5]. By interpolating the function in the discrete setting, it is always possible to obtain a separated representation of the function. As it is a finite dimensional space one can always express the function as a finite sum of base vectors. That is

$$i_D^{full} = \sum_{\substack{1\leqslant i_S\leqslant n_{DoF_S}\\ 1\leqslant i_T\leqslant n_{DoF_T}}} c_{(i_S,i_T)} N^S_{i_S}\otimes N^T_{i_T}, \quad (6.10)$$

where $c_{(i_S,i_T)} = 1$ if the node corresponding to (i_S,i_T) is in Γ_D and $c_{(i_S,i_T)} = 0$ otherwise. In general, this representation has a high number of modes. For computational reasons, an algebraic PGD separation scheme is used to approximate the full order representation.

In the current example, we have compared the results obtained by using the full order representation and the separated approximation. In order to reduce the computational power needed for computing the solution with the full order a coarse time discretisation of 20 elements has been used. With this, the full order function can be represented with at most 21 modes (in fact, 20 are enough). The results obtained with the full order representation are plotted in figures 6.8-6.10. We have used the fixed point strategy for the computation of the solution.

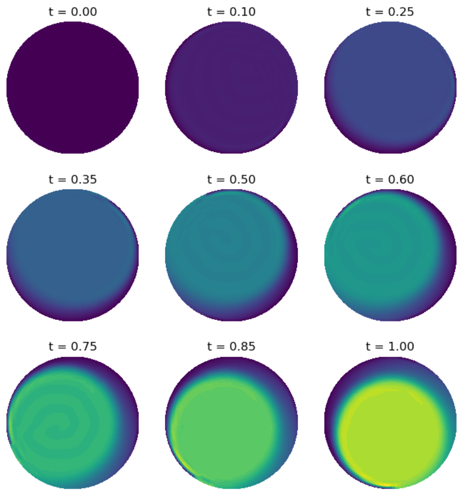

Figure 6.8 Full order solution

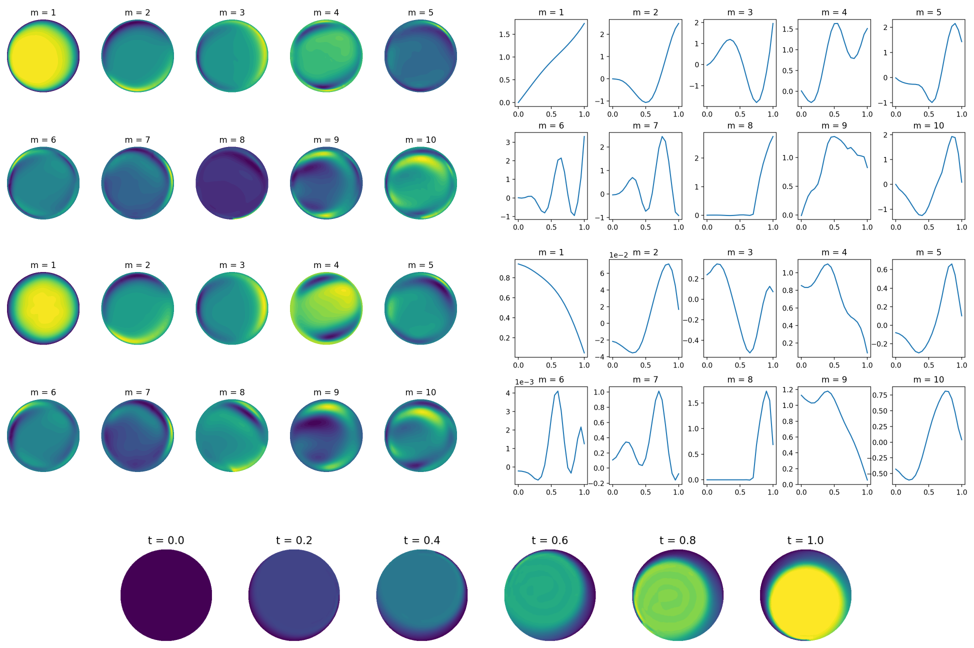

Figure 6.9 PGD approximation for the full order representation of i_D

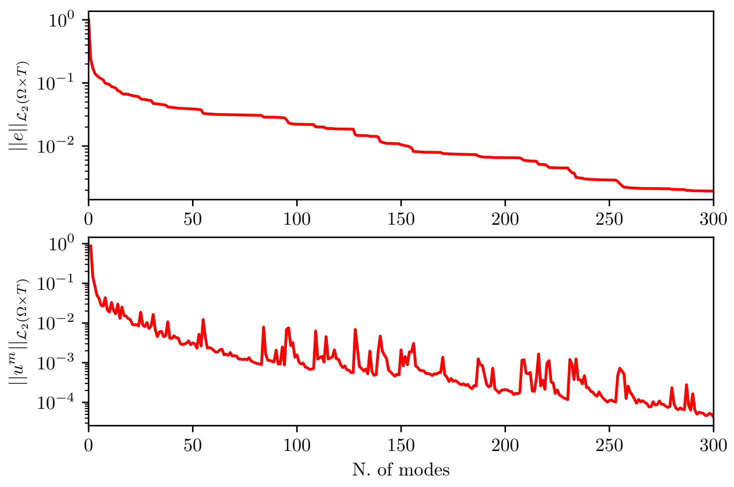

Figure 6.10 Convergence and modal amplitude for the full order representation of i_D

We have also computed the PGD solution using an approximation of i_D with only 10 modes (corresponding with a tolerance of 10^{-1} in the separation tensor algorithm). The results are plotted in figures 6.11 and 6.12

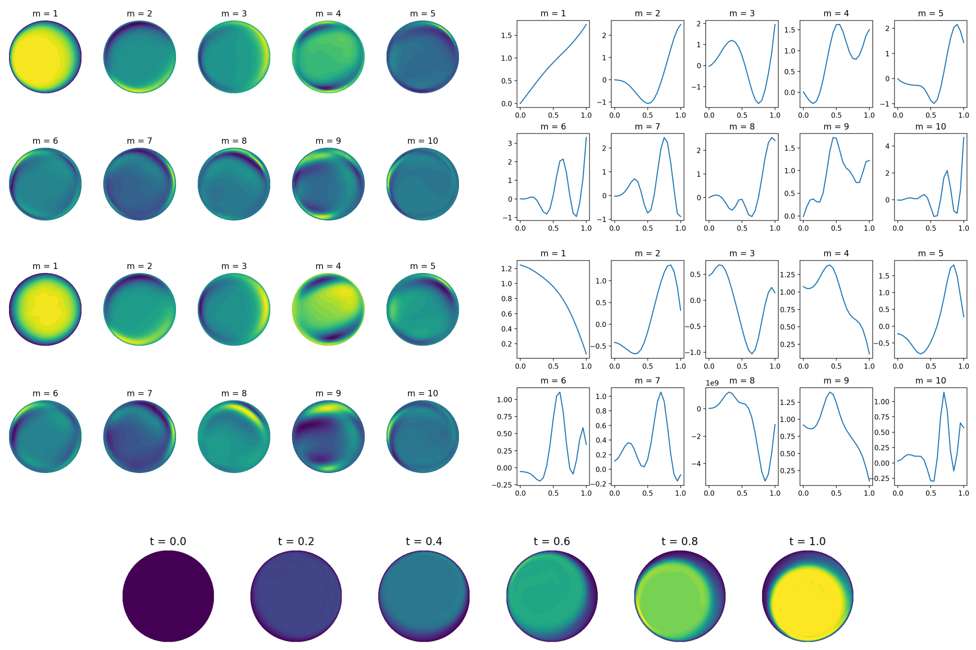

Figure 6.11 PGD approximation for the separated representation of i_D

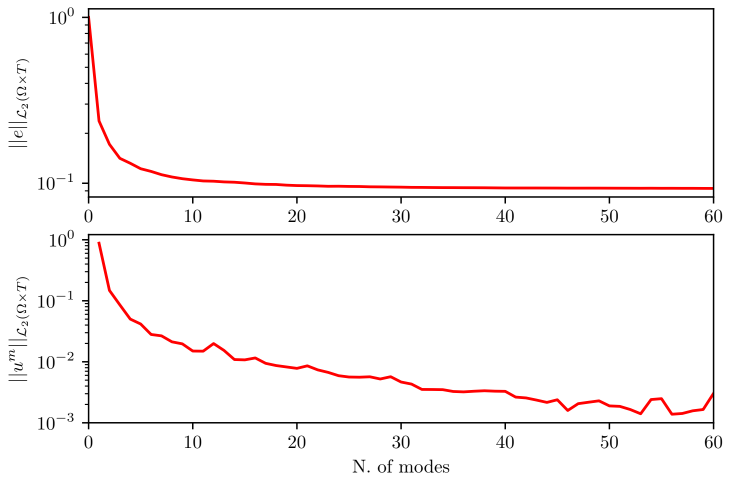

Figure 6.12 Convergence and modal amplitude for the separated representation of i_D

From the results provided by this example we see that for the full order tensor i_D, the PGD solution seems to converge towards the full order solution and, as expected, if one approximates i_D using a lower number of modes, the error stagnates at a value of the same order of the error of the approximation of i_D. Furthermore, to guarantee stability, the Nitsche's parameter β needs to be increased from 10^2 to 10^4 in the last example.

6.2 Concluding Remarks

In this chapter we have shown two examples of problems with boundary conditions such that its type depends on time (or any parameter in general). We have solved them using a PGD strategy and although the results are in general satisfactory, we note some points related to some difficulties arising when using this methodology.

- In case of changing BCs, the well-posedness of the continuous problem is not trivial to state. We have not discussed it as it is out of the scope of the thesis.
- The Dirichlet part of the BCs are imposed in weak form, this implies that the parameter to impose the coercivity has to be tuned.
- The rank-one problem lacks of convergence when using the standard fixed-point algorithm and its solution is not unique. We guess that this phenomenon may be due to a lack of stabilisation as noted in chapter 5.

Chapter 7

Algorithms for solving the rank-one approximation

In section 3.1.1 we defined the solution of the Galerkin rank-one approximation as the tuple $u_{DS} = (u_\alpha)_{\alpha\in A} \in \bigoplus_{\alpha\in A} \mathcal{V}_\alpha$ (DS stands for *Direct Sum*) that is solution of the non-linear equation (3.1). In this chapter, we discuss several methods for solving the non-linear problem. The first one is the traditional fixed-point problem. However, in some cases it does not converge. For this reason, we introduce a method of relaxation of the solution as well as several alternative methods. All but the vanishing viscosity method introduced are code intrusive.

Let us first remark that the convergence of all rank-one modes is not necessary in the PGD context. That is, if one adds a mode to the separated approximation that is not the solution of the corresponding rank-one problem, further modes added after those tend to mitigate the error of that *erroneous* mode. The problem arises when most of the rank-one problems do not converge and, in average, the error induced by the *erroneous* modes cannot be compensated by the converging modes.

We use the alternative formulation of the problem as a coupled system of equations (3.2). We assume that the rank-one solution exists and depends continuously on L. Note that we do not assume that the solution of the rank-one problem is unique. However, we may assume that any variation in u_{DS} that induces a variation in u_{R1} induces a variation on the residual of the rank-one problem (see section 7.2.1 for the explicit form of such a residual). This last assumptions implies that the set of solutions consists of isolated points in the tensor product space; but not in the direct sum space as it is evident from the following remark.

Remark 7.1: The representation of u_{R1} as the tensor product $\bigotimes_{\alpha\in A} u_\alpha$ is not unique. This is because of the following equivalence relation on the tensor product of two spaces

$$c(v\otimes w) \sim (cv)\otimes w \sim v\otimes(cw).$$

7.1 Fixed-point

This first algorithm is widely used in the PGD community to solve the rank-one problem. It consists of an iterative scheme over all dimensions in A. For each dimension one computes the solution of the corresponding function space considering the solutions of all other dimensions as known. After all dimensions have been computed, an error indicator is used to determine if the solution has converged.

The equation for the i-th dimension is: given $(u_\alpha)_{\alpha\in A\setminus\{i\}}$, find $u_i \in \mathcal{V}_i$ such that

$$B(u_{R1}, v) = L(v) \quad \forall v \in \mathcal{V}_i^{test}. \tag{7.1}$$

Using the separability properties of B and L the equation is rewritten as

$$\sum_{m\in M_B}\left[\prod_{\alpha\in A\setminus\{i\}} B_\alpha^m(u_\alpha,u_\alpha)\cdot B_i^m(u_i,v_i)\right] = \sum_{m\in M_L}\left[\prod_{\alpha\in A\setminus\{i\}} L_\alpha^m(u_\alpha)\cdot L_i^m(v_i)\right] \quad \forall v_i\in\mathcal{V}_i. \tag{7.2}$$

For the discrete case, suppose that all $\mathcal{V}_\alpha$ spaces are finite dimensional:

$$\mathcal{V}_\alpha = LS\Big(\big\{N_i^\alpha, i \in \{1, ..., n_{DoF_\alpha}\}\big\}\Big) \cong \mathbb{R}^{n_{DoF_\alpha}}.$$

Now, define the matrices $\boldsymbol{K}_\alpha^m$ and vectors $\boldsymbol{f}_\alpha^m$ in terms of the bilinear and linear forms:

$$\begin{aligned}\big[\boldsymbol{K}_\alpha^m\big]_{ij} &:= B_\alpha^m(N_j^\alpha, N_i^\alpha), \quad \alpha \in A, m \in M_B,\\ \big[\boldsymbol{f}_\alpha^m\big]_i &:= L_\alpha^m(N_i^\alpha), \quad \alpha \in A, m \in M_L.\end{aligned}$$

Equation (7.2) rewritten in matrix form is

$$\left(\sum_{m\in M_B} c^m \boldsymbol{K}_i^m\right)\boldsymbol{u}_i = \sum_{m\in M_L} \hat{c}^m \boldsymbol{f}_i^m. \tag{7.3}$$

Where

$$\begin{aligned}c^m &= \prod_{\alpha\in A\setminus\{i\}} \boldsymbol{u}_\alpha^T \boldsymbol{K}_\alpha^m \boldsymbol{u}_\alpha,\\ \hat{c}^m &= \prod_{\alpha\in A\setminus\{i\}} (\boldsymbol{f}_\alpha^m)^T \boldsymbol{u}_\alpha.\end{aligned}$$

Remark 7.2: The common implementation of the algorithm implies updating, at every iteration, the previously computed u_α and use it as guess for computing the rest of dimensions in the current iteration. That is, it uses an scheme similar to the Gauss-Seidel solver instead to the Jacobi one.

Remark 7.3: With the fixed-point approach, at every iteration, $|A|$ systems of equations are solved each of one of the size of its corresponding space $\mathcal{V}_\alpha$. Also note that the sparsity of the resulting system for each dimension is, in general, the same as the sparsity of the operator resulting of the sum of all modes for that dimension, i.e. $\sum_{m\in M_B} B_\alpha^m$.

7.1.1 Additional Modes

To reduce the error of the approximation, several modes can be added to the PGD form. The problem for computing these new modes are defined in (3.6) where we define an additional contribution to the linear functional L of the form $-\tilde{B}(u_M, v)$. With this modification, equation (7.2) reads

$$\begin{aligned}\sum_{m\in M_B}\left[\prod_{\alpha\in A\setminus\{i\}} B_\alpha^m(u_\alpha, u_\alpha)\cdot B_i^m(u_i, v_i)\right] &= \sum_{m\in M_L}\left[\prod_{\alpha\in A\setminus\{i\}} L_\alpha^m(u_\alpha)\cdot L_i^m(v_i)\right]\\ &\quad - \sum_{m_P=1}^{M}\sum_{m\in M_{\tilde{B}}}\left[\prod_{\alpha\in A} \tilde{B}_\alpha^m(u_\alpha^{m_P}, v_\alpha)\right] \quad \forall v_i \in \mathcal{V}_i. \end{aligned}\tag{7.4}$$

For the discrete case, define the matrix $\tilde{\boldsymbol{K}}_\alpha^m$ as follows:

$$\big[\tilde{\boldsymbol{K}}_\alpha^m\big]_{ij} := \tilde{B}_\alpha^m(N_j^\alpha, N_i^\alpha), \quad \alpha \in A, m \in M_{\tilde{B}}.$$

The matrix equation now has an additional term:

$$\left(\sum_{m\in M_B} c^m \boldsymbol{K}_i^m\right)\boldsymbol{u}_i = \sum_{m\in M_L} \hat{c}^m \boldsymbol{f}_i^m - \sum_{m_P=1}^{M}\left(\sum_{m\in M_{\tilde{B}}} \tilde{c}^{m,m_P}\tilde{\boldsymbol{K}}_i^{m_P}\right)\boldsymbol{u}_i^{m_P}. \tag{7.5}$$

Where

$$\tilde{c}^{m,m_P} = \prod_{\alpha\in A\setminus\{i\}} \boldsymbol{u}_\alpha^T \tilde{\boldsymbol{K}}_\alpha^m \boldsymbol{u}_\alpha^{m_P}.$$

7.1.2 Stopping criteria and normalisation

As noted in *Remark 7.1*, the solution of the system is not unique. This can lead to the situation where the approximation of the rank-one solution changes after every iteration without changing its tensor product. That is,

$$\left\| \bigotimes_{\alpha\in A} u_\alpha^i - \bigotimes_{\alpha\in A} u_\alpha^{i-1} \right\| = 0 \quad \text{in} \bigotimes_{\alpha\in A} \mathcal{V}_\alpha,$$
$$\left\| (u_\alpha^i)_{\alpha\in A} - (u_\alpha^{i-1})_{\alpha\in A} \right\| \neq 0 \quad \text{in} \bigoplus_{\alpha\in A} \mathcal{V}_\alpha.$$

Where $(u_\alpha^i)_{\alpha\in A}$ denotes the result of the i-th iteration.

To solve this ambiguity it is a common practice to normalise the solution. That is, to express the solution as

$$u_{R1} = \sigma \bigotimes_{\alpha\in A} u_\alpha, \quad \sigma \in \mathbb{R}^+,\ \|u_\alpha\|_\alpha = 1. \tag{7.6}$$

Using this representation, the solution is unique up to the sign of u_α. When a new iteration $(u_\alpha^*)_{\alpha\in A}$ is computed (u_α^* are not normalised), to check the convergence one normalises every vector and changes the sign of u_α^* if necessary. That is,

$$u_\alpha^i = \pm \frac{u_\alpha^*}{\|u_\alpha^*\|_\alpha}, \quad \alpha \in A.$$

Where the sign is chosen such that it minimises $\left\|u_\alpha^i - u_\alpha^{i-1}\right\|$.

Note that one should check that the number of signs changed is even. Define the modal amplitude as

$$\sigma^i = \prod_{\alpha\in A} \|u_\alpha^*\|.$$

The iteration scheme stops if two stopping criteria are fulfilled. First, the relative increment of the amplitude σ between two consecutive iterations must be smaller than a user-defined small tolerance $\epsilon_{amplitude} > 0$. Second, the norm in $\bigoplus_{\alpha\in A} \mathcal{V}_\alpha$ of the difference between two consecutive iterations must be smaller than $\epsilon_{modal} > 0$. That is,

$$\frac{\sigma^i - \sigma^{i-1}}{\sigma^i} < \epsilon_{amplitude},$$
$$\sum_{\alpha\in A} \left\|u_\alpha^i - u_\alpha^{i-1}\right\| < \epsilon_{modal}.$$

Note that norm in the last criterion can be substituted by any equivalent norm in $\bigoplus_{\alpha\in A} \mathcal{V}_\alpha$.

7.1.3 Relaxation

The fixed-point algorithm does not converge under all conditions. In some cases it is seen to not converge for all initial guesses of the solution and in others, even providing the exact solution u_{R1} as initial guess, any small perturbation grows exponentially. The stability of the fixed-point is seen to depend not only on the bilinear form B but also on L. In particular, for some problems the algorithm may be stable for some modes and not for others. To overcome this problem, after computing every new guess the solution is sub-relaxed. That is, after computing u_α^* using (7.5), the new guess is

$$u_\alpha^i = u_\alpha^* + (1-r)(u_\alpha^{i-1} - u_\alpha^*), \quad r \in (0,1].$$

With this procedure the solution is not normalised after every iteration, only for the sake of checking convergence.

Remark 7.4: The relaxation step can be applied after solving every single dimension or at the end of every iteration. In the former case, after computing each dimension the solution is relaxed and this new guess is used for the further dimensions in the same iteration. For the latter case, for the computation of the further iterations, the solution is not relaxed. In this case, is after the computation of all dimensions that the solution is relaxed. This makes a difference because as noted in *Remark 7.2* the new value of the solution is used for the following ones. In the examples used, the best convergence properties has been obtained, relaxating at the end of every iteration.

With the relaxation algorithm we obtained satisfactory results of the convergence of the fixed point algorithm. But we note a complex behaviour observed. In most iterative methods that use relaxation, the algorithm converges for some small enough value of the relaxation parameter r. In this method, the algorithm converge only if r is between some interval. This interval depends not only on the bilinear form B but also on the linear functional L. If r is too low the method does not converge as for the case that r is too large. This makes difficult the tuning of r.

7.2 Newton-Raphson

For the sake of improving the convergence of the fixed-point algorithm, we propose two methods based on the Newton-Raphson iterative method for the solution of the non-linear Galerkin PGD equation. In particular, we study the possibility of using the Newton-Raphson method to the system of equations without normalising and normalising according to section 7.1.2.

7.2.1 Newton-Raphson without normalisation

Again, the problem is to find $u_{DS} = (u_\alpha)_{\alpha\in A} \in \bigoplus_{\alpha\in A} \mathcal{V}_\alpha$ to represent the rank-one solution $u_{R1} = \bigotimes_{\alpha\in A} u_\alpha$. This can be stated as finding the zero of a non-linear operator: the residual. To define the residual, we use the system of equations (3.2). The i-th equation reads

$$B\big(u_{R1}, v \otimes \bigotimes_{\alpha\in A\setminus\{i\}} u_\alpha\big) - L\big(v \otimes \bigotimes_{\alpha\in A\setminus\{i\}} u_\alpha\big) = 0, \quad \forall v \in \mathcal{V}_i.$$

Where L includes the contributions from the previous modes. Using the Riesz representation theorem we obtain

$$\big\langle B(u_{R1}) - L, v \otimes \bigotimes_{\alpha\in A\setminus\{i\}} u_\alpha \big\rangle = 0, \quad \forall v \in \mathcal{V}_i.$$

Here we have used the same symbols B and L to denote the Riesz representation of the linear forms. Using the separability properties of B and L, as well as the separability of the inner product we get

$$\sum_{m\in M_B} \Big(\prod_{\alpha\in A\setminus\{i\}} \langle B^m_\alpha u_\alpha, u_\alpha\rangle_\alpha \cdot \langle B^m_i u_i, v\rangle_i \Big) - \sum_{m\in M_L} \Big(\prod_{\alpha\in A\setminus\{i\}} \langle L^m_\alpha, u_\alpha\rangle_\alpha \cdot \langle L^m_i, v\rangle_i \Big) = 0, \quad \forall v \in \mathcal{V}_i.$$

Define the residual as

$$\begin{aligned} r : \bigoplus_{\alpha\in A} \mathcal{V}_\alpha &\to \bigoplus_{\alpha\in A} \mathcal{V}_\alpha, \\ u_{DS} &\mapsto \left(\sum_{m\in M_B} \Big(\prod_{i\in A\setminus\{\alpha\}} \langle B^m_i u_i, u_i\rangle_i \cdot B^m_\alpha u_\alpha \Big) - \sum_{m\in M_L} \Big(\prod_{i\in A\setminus\{\alpha\}} \langle L^m_i, u_i\rangle_i \cdot L^m_\alpha \Big) \right)_{\alpha\in A}. \end{aligned} \tag{7.7}$$

With this definition of the residual, its variation is:

$$\begin{aligned} \delta r(u_{DS}) : \bigoplus_{\alpha\in A} \mathcal{V}_\alpha &\to \bigoplus_{\alpha\in A} \mathcal{V}_\alpha, \\ \delta u_{DS} = (\delta u_\alpha)_{\alpha\in A} &\mapsto \sum_{\alpha\in A} \delta r_\alpha(\delta u_\alpha). \end{aligned} \tag{7.8}$$

where δr_α is defined component-wise as:

$$\big(\delta r_\alpha(\delta u_\alpha)\big)_\beta = \begin{cases} \sum_{m\in M_B}\Big(\prod_{i\in A\setminus\{\alpha\}}\langle B_i^m u_i, u_i\rangle_i \cdot B_\alpha^m \delta u_\alpha\Big) \text{ if } \alpha=\beta, \\ \sum_{m\in M_B}\Big(\prod_{i\in A\setminus\{\alpha,\beta\}}\langle B_i^m u_i, u_i\rangle_i \cdot \langle (B_\alpha^m + (B_\alpha^m)^*)u_\alpha, \delta u_\alpha\rangle_\alpha \cdot B_\beta^m u_\beta\Big) - \\ -\sum_{m\in M_L}\Big(\prod_{i\in A\setminus\{\alpha,\beta\}}\langle L_i^m, u_i\rangle_i \cdot \langle L_\alpha^m, \delta u_\alpha\rangle_\alpha \cdot L_i\Big) \text{ otherwise.}\end{cases}$$

It is immediate to check that if all B_α^m are self-adjoint operators in $\mathcal{V}_\alpha$, then δr_α is self-adjoint for all in the direct sum space $\alpha \in A$ and so $\delta r(u_{DS})$.

Now we define a function to iterate the successive guesses of the approximation to u_{R1} by using the Newton-Raphson iterative scheme.

$$u^{i+1} = u^i - \delta r^{-1}(u^i)(r(u^i)). \tag{7.9}$$

Remark 7.5: At each iteration only one linear system has to be solved. The size of the system is the sum of every single dimension. The sparsity of the systems in the diagonal terms, i.e. $(\delta r_\alpha(\delta u_\alpha))_\beta$ for $\alpha=\beta$, conserve the sparsity of $\sum_{m\in M_B} B_\alpha^m$ whereas the off diagonal terms are full.

Remark 7.6: If the off-diagonal terms of the variation operator $(\delta r_\alpha(\delta u_\alpha))_\beta$ for $\alpha \neq \beta$ are substituted by 0 we recover the Jacobi version of the fixed-point algorithm.

There is a problem with this approach. We noted in *Remark 7.1* that the solution is not unique. For this reason, the residual is not an invertible linear function when the guess is the actual solution of the system. The variation map is a bijection when the guess is not the solution of the system. However, the function $u \mapsto \delta r(u)$ is continuous meaning that the system becomes ill-conditioned when the guess is close to the solution.

By performing numerical examples it is seen that this method presents a faster convergence than the fixed-point algorithm in terms of number of iterations. However, when the solution gets close to the solution of the system (error indicator of order 10^{-3}), at the step of solving the linear system associated to (7.9) numerical errors amplify in a fatal way obtaining a new guess with error of order 10^{10}.

Stabilisation of Newton-Raphson algorithm

To get over the problem of the singularity of δr when the guess is close to the solution of the system, an alternative formulation of (7.9) is proposed. From now on suppose that u is a solution of the rank-one problem. If we knew the nullspace $N(\delta r)$ and the annihilator of the range $R(\delta r)^\perp$, we could solve for the new guess. The natural way is to restrict the operator as $\delta r : \bigoplus_{\alpha\in A}\mathcal{V}_\alpha/N(\delta r) \to \bigoplus_{\alpha\in A}\mathcal{V}_\alpha/R^\perp(\delta r)$. To do that, one has to define a new basis for both spaces and to compute the restriction of the operator with such basis. The computational cost is too high so we adopt an alternative approach. In (7.9), replace $\delta r^{-1}(u)(r(u))$ by

$$\Big(\delta r(u) + \sum_{i=1}^{N}\alpha_i k_i b_i\Big)^{-1}\Big(r(u) - \sum_{i=1}^{N}\langle b_i, r(u)\rangle\Big).$$

where k_i and b_i form a basis on $N(\delta r)$ and $R(\delta r)^\perp$ ($\dim N(\delta r) = \dim R(\delta r) = N$) and the scalars α_i are chosen in order to preserve the condition number of the operator.

Now, we reason which is the space $N(\delta r)$. For that, we note the fact that if $r(u)=0$, then

$$r(u^*) = 0 \quad \forall u^* = \big(cu_\beta, c^{-1}u_\gamma, (u_\alpha)_{\alpha\in A\setminus\{\beta,\gamma\}}\big),\ \beta,\gamma\in A.$$

That is, multiplying the function of on dimension by an scalar c and the function of another dimension by c^{-1} makes no change in the residual (as it induces no change in u_{R1}).

If we take the variation of u^* with respect to c and evaluate at u, i.e. $c=1$, we obtain

$$\delta u^*(1)(\delta c) = (u_\beta, -u_\gamma, (0)_{\alpha\in A\setminus\{\beta,\gamma\}}).$$

With this we conclude that all variations of the previous form do not induce any variation in the residual (which is null), so they are in the nullspace of δr. Furthermore, by the assumption that if δu induces a variation in u_{R1}, then it induces a variation in r, it follows that the nullspace of δr is the linear span of vectors of the previous form. That is,

$$N(\delta r) = LS\Big(\Big\{u \in \bigoplus_{\alpha\in A} \mathcal{V}_\alpha : u = (u_\beta, -u_\gamma, (0)_{\alpha\in A\setminus\{\beta,\gamma\}}),\ u_\beta \in \mathcal{V}_\beta, u_\gamma \in \mathcal{V}_\gamma,\ \beta,\gamma \in A\Big\}\Big). \tag{7.10}$$

From the fundamental theorem of linear algebra we know that for the self-adjoint case, $N(\delta r) = R(\delta r)^\perp$. For the non-self adjoint case we use another reasoning.

We make the usual assumption that the solution u_{R1} depends continuously on L. Now, think of the residual map r as a function of $(L_\alpha^m)_{\alpha\in A, m\in M_L}$. We denote this map by r_L. In r_L we fix u to be one of the solutions of the system $r = 0$. Now we claim that $R(\delta r) = R(\delta r_L)$.

On the one hand, by the continuity assumption we have $R(\delta r) \supseteq R(\delta r_L)$. Suppose by contradiction that exists δL such that $\delta r_L(L)(\delta L) \notin R(\delta r)$. Then, we could define a new problem by doing an arbitrarily small change in L in the direction δL. By definition of r_L, the residual of this new problem evaluated in the solution of the original problem is $r^* u = \delta(r_L)(L)(\delta L)$. However, as $\delta(r_L)(L)(\delta L) \notin R(\delta r)$ there is not an arbitrarily small change in u such that $r^* = 0$ this contradicts the assumption of continuity of the solutions with respect to L.

On the other hand, we have $R(\delta r) \subseteq R(\delta r_L)$. To prove that, pick any $\delta u \in \bigoplus_{\alpha\in A} \mathcal{V}_\alpha$. Define δL such that

$$\sum_{m\in M_L} \bigotimes_{\alpha\in A} \delta L_\alpha^m = \lim_{\epsilon\to 0} \epsilon^{-1}\Big(B\big(\bigotimes_{\alpha\in A}(u_\alpha + \epsilon\delta u_\alpha)\big) - B\big(\bigotimes_{\alpha\in A}(u_\alpha)_{\alpha\in A}\big)\Big).$$

Here B denotes the Riesz representation of the original bilinear form. Using the separability property of B, we ensure that the representation of δL is possible for a finite number of modes. If $|M_L|$ is not large enough such that the previous representation is possible, add additional modes to L such that they are all 0. With that we have

$$\lim_{\epsilon\to 0}\Big(\epsilon^{-1}\big(r(u + \epsilon\delta u_\alpha; L - \epsilon\delta L) - r(u; L)\big)\Big) = 0.$$

Where

$$r(u; L) = B\Big(\bigotimes_{\alpha\in A} u_\alpha\Big) - \sum_{m\in M_L} \bigotimes_{\alpha\in A} L_\alpha^m.$$

We conclude that $\delta r(u)(\delta u) - \delta r_L(L)(\delta L) = 0$. That is, for every element in $R(\delta r)$ we can chose δL so that $\delta r_L(L)(\delta L) = \delta r(u)(\delta u)$ completing the proof.

At this point we know that $R(\delta r) = R(\delta r_L)$ and that for the case where the bilinear forms are self-adjoint, $R(\delta r)^\perp = N(\delta r)$. A quick inspection reveals that $R(\delta r_L)$ does not depend on B. With that we conclude that $R(\delta r)^\perp = N(\delta r)$ also for the non self-adjoint cases.

7.2.2 Newton-Raphson with normalisation

An alternative approach to overcome the problem of the non-uniqueness of the solution, is to enforce the normalisation defined in (7.6). With this approach, the solution in the direct sum space is not unique but consists of isolated points (assuming that the solution is different from 0). This is due to the possibility of multiplying by -1 an even number of functions. The definition of this alternative residual is

$$r : \mathbb{R} \oplus \bigoplus_{\alpha\in A} \mathcal{V}_\alpha \to \mathbb{R}^{|A|} \oplus \bigoplus_{\alpha\in A} \mathcal{V}_\alpha,$$

$$(\sigma, u_{DS}) \mapsto \Bigg(\big(\langle u_\alpha, u_\alpha\rangle_\alpha - 1\big)_{\alpha\in A},$$
$$\Big(\sigma\Big(\sum_{m\in M_B}\big(\prod_{i\in A\setminus\{\alpha\}}\langle B_i^m u_i, u_i\rangle_i\big)B_\alpha^m u_\alpha\Big) - \sum_{m\in M_L}\big(\prod_{i\in A\setminus\{\alpha\}}\langle L_i^m, u_i\rangle_i\big)L_\alpha\Big)_{\alpha\in A}\Bigg). \tag{7.11}$$

The variation of such residual is

$$\begin{aligned}\delta r\big(\sigma,(u_\alpha)_{\alpha\in A}\big) : \mathbb{R}\oplus\bigoplus_{\alpha\in A}\mathcal{V}_\alpha &\to \mathbb{R}^{|A|}\oplus\bigoplus_{\alpha\in A}\mathcal{V}_\alpha\\ (\delta\sigma,0) &\mapsto \Bigg(0,\Big(\delta\sigma\Big(\sum_{m\in M_B}\big(\prod_{i\in A\setminus\{\alpha\}}\langle B_i^m u_i,u_i\rangle_i\big)B_\alpha^m u_\alpha\Big)\Big)_{\alpha\in A}\Bigg)\end{aligned}$$

$$\begin{aligned}\Big(0,\big(\delta u_\alpha,(0)_{i\in A\setminus\{\alpha\}}\big)\Big) \mapsto \Bigg(\big(2\langle u_\alpha,\delta u_\alpha\rangle_\alpha,(0)_{i\in A\setminus\{\alpha\}}\big),\Big(\Big(\sigma\big(\sum_{m\in M_B}\big(\prod_{i\in A\setminus\{\alpha\}}\langle B_i^m u_i,u_i\rangle_i\big)B_\alpha^m\delta u_\alpha\big)\Big),\\ \Big(\sigma\big(\sum_{m\in M_B}\big(\prod_{j\in A\setminus\{\alpha,i\}}\langle B_j^m u_j,u_j\rangle_j\big)\cdot\langle(B_\alpha^m+B_\alpha^{m*})u_\alpha,\delta u_\alpha\rangle_\alpha\cdot B_i^m u_i\big)-\\ \sum_{m\in M_L}\big(\prod_{j\in A\setminus\{\alpha,i\}}\langle L_i^m,u_i\rangle_i\big)\cdot\langle L_\alpha^m,\delta u_\alpha\rangle_\alpha\cdot L_i^m\Big)_{i\in A\setminus\{\alpha\}}\Big)\Bigg)\end{aligned}$$

We extend the definition of the residual operator by a linearity argument. Note that this non-linear system is over-determined resulting in an over-determined residual operator. To solve the linearised problem (i.e. to invert the residual), the standard least squares method is not accurate as the resulting matrix is ill-conditioned due to the imposition of the equations enforcing the normalisation. For that reason, a SVD strategy has been used. However, the Newton-Raphson method using this over-determined residual lacks of convergence. We guess that is due to that the constraints imposing the normalisation of the functions. One possible strategy to overcome the problem of ill-conditioning is to multiply every equation imposing the normalisation by a scalar c_α obtaining an equivalent system with lower condition number.

7.2.3 Vanishing diffusivity

After devising all previous methods, we come up with a simpler idea to ensure the convergence of the rank-one problem. The method is restricted to problems where one can introduce a diffusivity operator.

First we noted that for most cases, if the initial guess is close enough to the solution, the standard fixed-point converges. We also noted that for high enough values of the diffusivity parameter, the solution converged independently of the initial guess. Finally, as the problem is assumed to be well-posed, its solution must depend continuously on the diffusivity parameter (except maybe at $\nu = 0$). We may also assume that the rank-one problem is well posed in the sense that its solution also depends continuously on the diffusivity parameter. Having noted that points, we can devise a simple method to solve the rank-one problem. The rank one problem is to find the rank one solution of

$$B(u_{R1},v;\nu) = L(v) - \tilde{B}(u_M,v)\forall v\in\mathcal{V}^{test}.$$

Where we have written explicitly the dependence of the bilinear form with respect to the diffusivity.

The method consists in first solve the problem with a high enough value for the diffusivity ν. Note that the bilinear form taking into account the contribution of the previous modes is not be modified. Then we use the obtained solution as initial guess of the same problem with a lower value of ν. We construct in that way a sequence of decreasing values of ν until reaching the originally prescribed value of ν.

In the example of chapter 5, the diffusivity is set to $\nu = 0$. We have used as first value $\nu = 10^3$. Then, we reduced the diffusivity exponentially in 16 steps until $\nu = 10^{-5}$. That is, at each step the diffusivity is multiplied by $(10^-8)^{\frac{1}{16}}$. Finally, from we use the solution with $\nu = 10^{-5}$ as initial guess for the problem with $\nu = 0$. With this, to compute every mode we have to compute 18 auxiliary problems with approximately the same computational cost. This results in an increase of the computational cost of the PGD algorithm by a factor of 18. This choice of the sequence of values of ν is not optimised and for most rank-one problems of the same problem only about 3 steps are necessary if chosen optimally. However, the optimisation of the computational cost is beyond the scope of this thesis.

7.2.4 Concluding remarks

We conclude this chapter on algorithms for solving the rank-one problem with some remarks on each of the devised methods. We present the conclusions obtained solving the examples presented in chapters 5 and 6. As stated in previous chapters, we guess that the difficulty in solving such examples arises from the lack of stabilisation of the rank-one problems.

– **Fixed point:** This is the simplest algorithm and is very competitive in terms of computational efficiency. However, in the examples of chapters 5 and 6, this method lacks of convergence. The sub-relaxation of the fixed point produces satisfactory results with the appropriate value of the relaxation parameter r. In most iterative algorithms where the iterations are relaxed, the iterations converge provided the relaxation factor is small enough. This is not the case for the mentioned examples. For the example in which the boundary conditions are time dependent, the appropriate value of r is approximately in the interval $[0.7, 0.8]$ for most of the modes computed. However, for some particular modes the relaxation parameter interval has to be reduced to $[0.3, 0.5]$ to obtain convergence. This tuning of the parameter has been done manually. As future work we note the study of an adaptive algorithm optimising the relaxation parameter. This is not a trivial task as there is no simple criteria to determine if the relaxation factor is too high or too low for the non-converging cases.

– **Newton-Raphson:** This method has the drawback (as all methods computing the solution in a monolithic way) that, as the solution is not unique, the algorithm leads to singular matrices. The stabilisation proposed solves that problem. However, the resulting matrix has full sparsity, resulting in an unaffordable computational cost for most cases. The same happens with the normalisation method proposed. As the system is overdetermined, the method of least squares uses the matrix $A^T A$ that in general is full. In addition, the problem should be preconditioned. Alternatively one can use a SVD strategy, but the computational cost is also to high and the iterations do not converge.

– **Vanishing diffusivity:** This algorithm has proved to converge in all cases for both examples if tuned correctly. The main drawback of the method is that, if the values of diffusivity used to obtain the different approximations to the solution are not chosen optimally, to solve each mode of the approximation of the solution one has to solve several rank-one problems incurring a high computational cost. The design of an optimal adaptive algorithm is not obvious. In the example of chapter 6 we found that for some cases particular values of the diffusivity, the fixed point becomes unstable making the solution diverge however close the initial guess is. The solution to this problem is to *skip* the unstable value of the diffusivity. Although this is simple to do if the values of diffusivity are chosen manually it is not obvious how to program an algorithm able to do that. Finally, we note that the main advantage of this method is that is not code intrusive. That is, each rank-one problem can be solved using a conventional Galerkin PGD solver. However, the best results are obtained by combining the vanishing diffusivity with sub-relaxed fixed point algorithm. That is, every rank-one problem arising in the vanishing diffusivity method is solved using a sub-relaxed fixed point method.

Finally, let us remind that in the PGD methods, it is not required to converge all rank-one problems. For example, one can chose a fixed relaxation parameter for the fixed point such that most of the rank-one problems converge and hope that the non-convergent modes do not disturb the convergence properties of the PGD method.

Chapter 8

Further examples

In this chapter, we include two further examples based on the transient advection-diffusion problem over the unit square. What makes the examples different is that they are parametric. That is, we introduce an additional dimension to enable a parameter dependency of the PDE.

8.1 Parametric diffusion

This problem is identical to the advection-diffusion problem of chapter 3 with the exception that the diffusivity is a function of the parameter p_ν. The strong form of the problem is to find $u(p_\nu) \in C^1\big(T; C^2(\Omega)\big)$ $\forall p_\nu \in \mathcal{P}_\nu$.

$$\begin{aligned} u_t(p_\nu) + \nabla \cdot (cu(p_\nu) - \nu(p_\nu)\nabla u(p_\nu)) &= 0 \quad \text{in } \Omega \times T, \\ u(p_\nu) &= 0 \quad \text{on } \partial\Omega \times T, \\ u(p_\nu) &= u_0 \quad \text{on } \Omega \times \{0\}. \end{aligned} \tag{8.1}$$

where the domain of the parametric PDE is defined as

$$\begin{aligned} \Omega &= (0,1)^2, \\ T &= (0,1), \\ \mathcal{P}_\nu &= (-5,-2). \end{aligned}$$

The parameters are defined as

$$\begin{aligned} c(x,y) &= \pi\Big(-y + \frac{1}{2}, x - \frac{1}{2}\Big), \\ \nu(p_\nu) &= 10^{p_\nu}, \\ u_0(x,y) &= \exp\left(- \frac{(x-\frac{2}{3})^2 + (y-\frac{1}{2})^2}{0.07^2} \right). \end{aligned}$$

The space of weak solutions is defined as

$$\mathcal{V} = \mathcal{V}_\Omega \otimes \mathcal{V}_T \otimes \mathcal{V}_{\mathcal{P}_\nu} = \mathcal{H}^1_0(\Omega) \otimes \mathcal{H}^1(T) \otimes \mathcal{L}_2(\mathcal{P}_\nu). \tag{8.2}$$

The space $\mathcal{V}$ consists of all functions such that for every dimension, the function defined by fixing the rest of dimensions is in the corresponding sectional space. To make it precise, $\mathcal{V}$ consists of all functions $u : \Omega \times T \times \mathcal{P}_\nu \to \mathbb{R}$ such that

$$\begin{aligned} \big(\Omega \ni x \mapsto u(x,t,p_\nu)\big) &\in \mathcal{V}_\Omega \quad \text{for a.e. } (t,p_\nu) \in T \times \mathcal{P}_\nu, \\ \big(T \ni t \mapsto u(x,t,p_\nu)\big) &\in \mathcal{V}_T \quad \text{for a.e. } (x,p_\nu) \in \Omega \times \mathcal{P}_\nu, \\ \big(\mathcal{P}_\nu \ni p_\nu \mapsto u(x,t,p_\nu)\big) &\in \mathcal{V}_{\mathcal{P}_\nu} \quad \text{for a.e. } (x,t) \in \Omega \times T. \end{aligned}$$

The forms defining the weak problem are defined on rank-one elements as

$$B\big(u_\Omega \otimes u_T \otimes u_{\mathcal{P}_\nu}, v_\Omega \otimes v_T \otimes v_{\mathcal{P}_\nu}\big) = \int_\Omega u_\Omega v_\Omega \; d\Omega \int_T (u_T)_t v_T \; dT \int_{\mathcal{P}_\nu} u_{\mathcal{P}_\nu} v_{\mathcal{P}_\nu} \; d\mathcal{P}_\nu + \int_\Omega \nabla \cdot (c u_\Omega) v_\Omega \; d\Omega \int_T u_T v_T \; dT \int_{\mathcal{P}_\nu} u_{\mathcal{P}_\nu} v_{\mathcal{P}_\nu} \; d\mathcal{P}_\nu + \int_\Omega \nabla u_\Omega \cdot v_\Omega \; d\Omega \int_T u_T v_T \; dT \int_{\mathcal{P}_\nu} \nu u_{\mathcal{P}_\nu} v_{\mathcal{P}_\nu} \; d\mathcal{P}_\nu + \int_\Omega u_\Omega v_\Omega \cdot \big(u_T(0) v_T(0)\big) \; \cdot \int_{\mathcal{P}_\nu} u_{\mathcal{P}_\nu} v_{\mathcal{P}_\nu} \; d\mathcal{P}_\nu, \qquad (8.3)$$

$$L\big(v_\Omega \otimes v_T \otimes v_{\mathcal{P}_\nu}\big) = \int_\Omega u_0 v_\Omega \; d\Omega \cdot \big(v_T(0)\big) \cdot \int_{\mathcal{P}_\nu} v_{\mathcal{P}_\nu} \; d\mathcal{P}_\nu. \qquad (8.4)$$

At this point, we use the Petrov-Galerkin PGD algorithm to obtain a separated approximation. The discretisation used consists of a 40 × 40 discretisation of regular triangular elements for the spatial domain and a uniform mesh of 100 elements for the temporal and parametric dimensions. The reference solution has not been computed due to the high computational cost required. Instead, the modal amplitude has been used as the classical error estimator in the PGD algorithms. In figure 8.1 the first 10 modes of the separated approximation are plotted. From top to bottom, the solution in the spatial, temporal and parametric dimensions. On the left hand side the primal solutions have been plotted and on the right hand side, the dual ones.

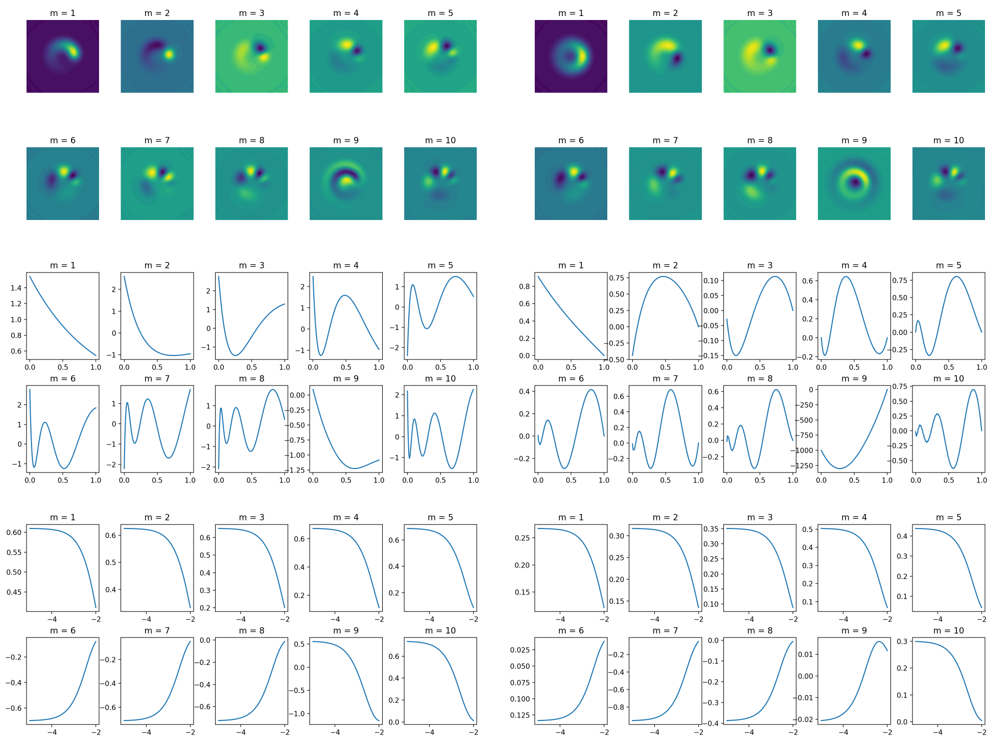

Figure 8.1 PGD approximation for the parametric diffusion problem

Next, we plot the reconstructed solutions for different values of the parameter p_ν. In particular for $p_\nu \in \{-5, -3.5, -2\}$ in decreasing order from top to bottom.

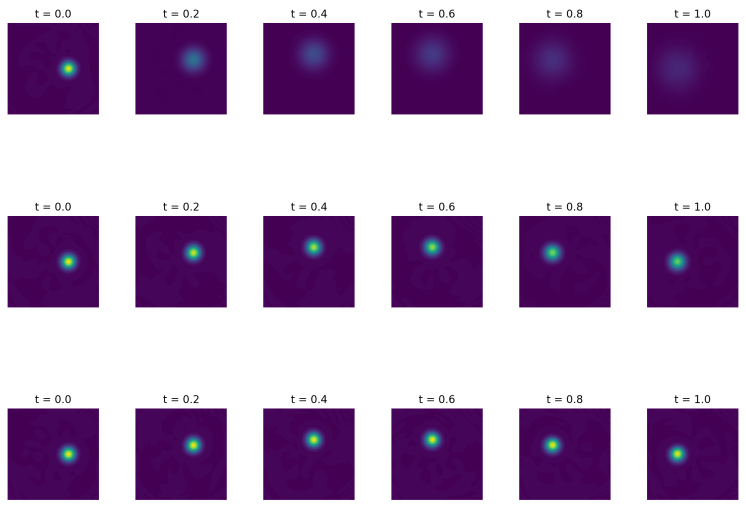

Figure 8.2 Reconstructed solution for the parametric diffusion problem

Finally, we plot the modal amplitude.

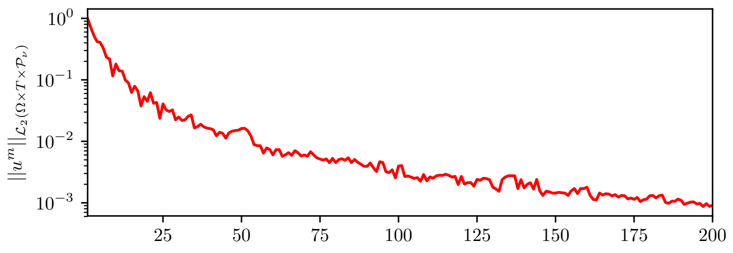

Figure 8.3 Modal amplitude for the parametric diffusion problem

In this example we have successfully computed the approximation of the parametric transient advection-diffusion problem.

8.2 Parametric advection

This second example is very similar to the previous one. In this case, the parameter modules the amplitude of the advection. For the example, we replace the domain $\mathcal{P}_\nu$ by $\mathcal{P}_c = (0.1, 1)$. The parameters involving the parametric PDE are modified but they remain rank-one functions:

$$\begin{aligned} c(x, y, p_c) &= 2\pi p_c\Big(-y+\frac{1}{2}, x-\frac{1}{2}\Big), \\ \nu &= 10^{-3}, \\ u_0(x, y) &= \exp\left(-\frac{(x-\frac{2}{3})^2+(y-\frac{1}{2})^2}{0.07^2}\right). \end{aligned}$$

Note that the parameter p_c represents the revolutions that a particle does over the interval of time.

The solution has been computed using the same discretisation up to 1155 modes. We show the plot of the first 10 modes of the PGD approximation. The layout is the same than in the previous example.

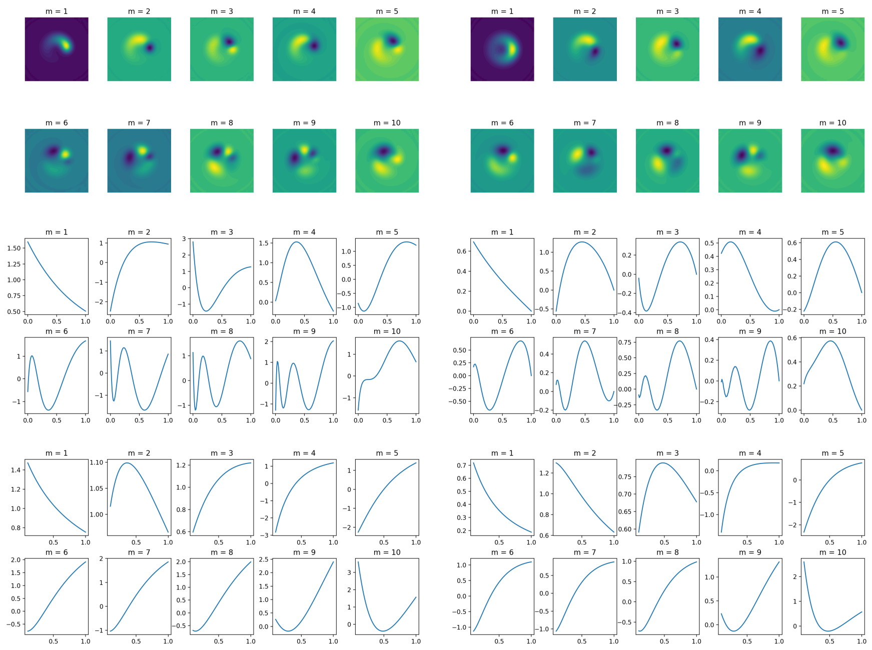
Figure 8.4 PGD approximation for the parametric advection problem

Now we show the reconstruction of the solution with all the modes. The chosen values of the parameter are $0.1, 0.235, 0.496, 0.748, 1.0$.

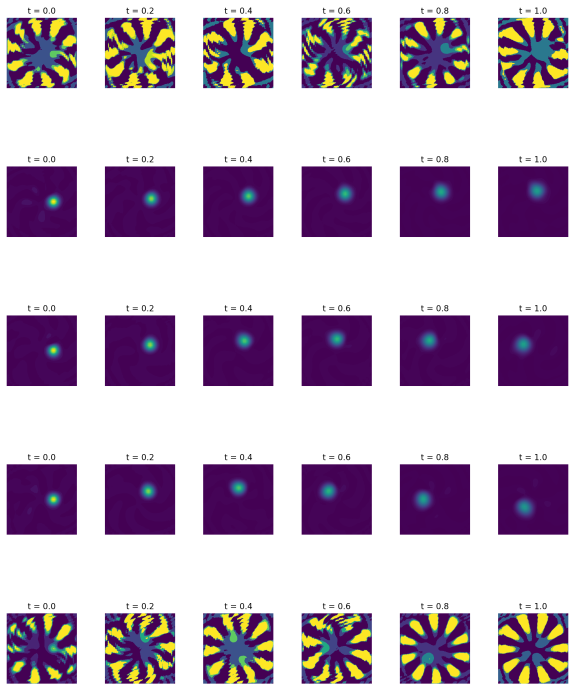
Figure 8.5 Reconstructed solution for the parametric advection problem

The solution is clearly not satisfactory. For the values of the parameter near the extremes of the domain, the solution presents large spurious oscillations. If we plot the modal amplitude we see that it presents large oscillations and does not converge to 0.

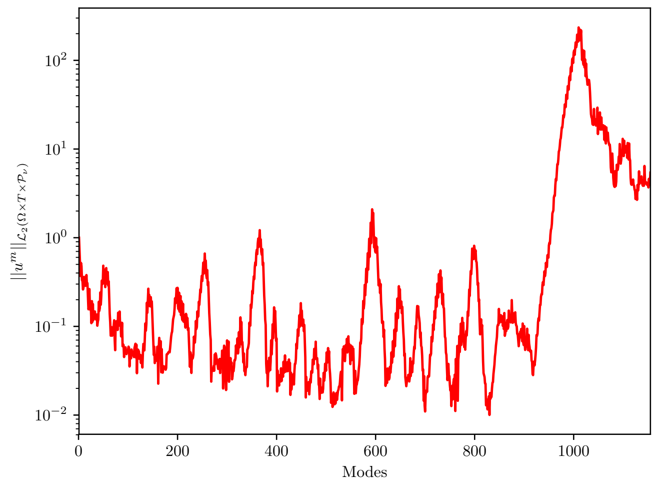

Figure 8.6 Modal amplitude for the parametric diffusion problem

Finally, we recreate the same plot of the reconstructed solution by showing the reconstruction of only the first 830 modes. This corresponds to the minimum in the modal amplitude.

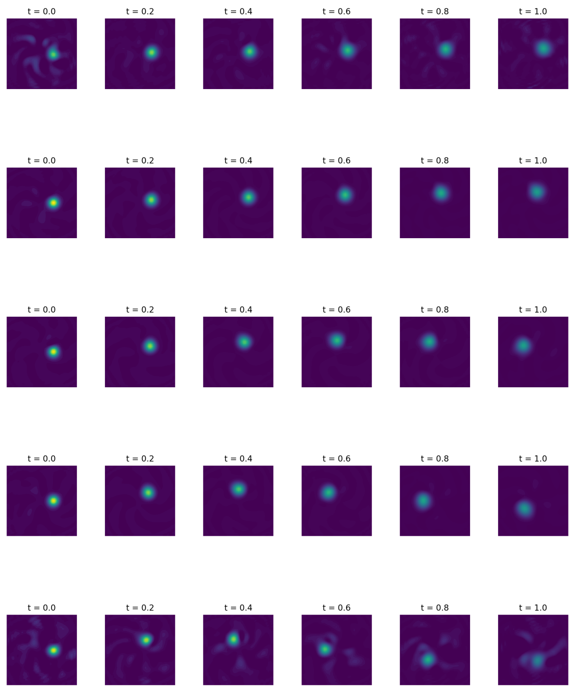

Figure 8.7 Reconstructed solution of the first 830 modes for the parametric advection problem

In this case, the approximation is qualitatively much more accurate. However, it stills presents *noise* on the solutions where the parameter is near the boundary. To conclude, we must say that the results are clearly not satisfactory for this last example. We guess that the reason of that is because the solution is difficult to be represented using a separated approximation. In any case, maybe weighting more the extremes of the parametric interval would help to mitigate the problem of convergence.

Chapter 9

Concluding Remarks

The capabilities of the Petrov-Galerkin PGD methodology have been explored. As expected, it supposes a great improvement compared to the Galerkin PGD in cases of non self-adjoint operators. In particular, it has been successfully applied to solve the transient advection-diffusion problem.

A relatively non-intrusive algorithm has been introduced to implement the Petrov-Galerkin and the update methods. In the Petrov-Galerkin case, this implementation results in the doubling of each space of functions. Alternatively, the method can be implemented such that its computational cost is approximately equal to the Galerkin PGD method. This makes the method very competitive compared to the Galerkin method.

Using the Petrov-Galerkin method, the transient advection-diffusion problem has been solved in an advection dominated problem using a SUPG stabilisation. Also the problem of imposing temporal/parameter BCs type has been solved using an weak imposition of the essential BCs. Although the convergence properties of the rank-one problems arising in such problems have poor convergence properties, several alternative solvers have been proposed and successively implemented.

Chapter 10

Future Work

In the course of the thesis we have noted several points that are out of the scope of the present thesis. Most of them focus on the reduction of the computational cost of the methods presented. We list them.

1. The update procedure is relatively cheap computationally speaking if the dimensions to update are small compared with the rest and the number of modes to update is small. We note as future work to study if performing the update method only to the last M modes (M being relatively small), the method improves the convergence properties as well as conserving the low computational cost.

2. We have implemented the update procedure using a non-intrusive strategy. The method used implies enlarging every space of dimensions to update by a factor M (M being the number of dimensions to update) and imposing N coupled equations (N being the number of dimensions to update). An alternative approach is to not enlarge the spaces and impose $N \cdot M$ coupled equations. In that way, each equation has the original size and using the computed modes as intial point the method may converge faster than the alternative adopted.

3. The rank-one problems arising in the advection dominated problem stabilised using the SUPG method lack of convergence using standard methods. In chapter 5 we propose an alternative approach that consists in stabilising every rank-one iteration independently. This is possible to do in the case of the Galerkin PGD method but it has to be studied if in the Petrov-Galerkin the problems that arise are well-posed.

4. In chapter 6 the problem of temporal/parametric BCs type is assumed to be well posed. We note as future work the study of the well posedness of the problem. That is, to study if the weak form defined has indeed a solution, it is unique and depends continuously on the parameters.

REFERENCES

[1] Jean Donea and Antonio Huerta. *Finite Element Methods for Flow Problems.* Wiley, 2003.

[2] L. C. Evans. *Partial Differential Equations.* American Mathematical Society, 2010.

[3] Anthony Nouy. "A priori model reduction through proper generalized decomposition for solving time-dependent PDEs". In: *Computer Methods in Applied Mechanics and Engineering* 199 (Apr. 2010), pp. 1603–1626. DOI: 10.1016/j.cma.2010.01.009.

[4] Alberto Sibileau et al. "Explicit parametric solutions of lattice structures with proper generalized decomposition (PGD): Applications to the design of 3D-printed architectured materials". In: *Computational Mechanics* 62 (Jan. 2018). DOI: 10.1007/s00466-017-1534-9.

[5] Sergio Zlotnik et al. "Effect of the separated approximation of input data in the accuracy of the resulting PGD solution". In: *Advanced Modeling and Simulation in Engineering Sciences* 2 (Dec. 2015). DOI: 10.1186/s40323-015-0052-6.