Streamline Diffusion POD Models in Optimization

Dissertation

zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften

dem Fachbereich IV der Universität Trier
vorgelegt von

Bret Kragel

Trier, 2005

Eingereicht am 23. Mai 2005

1. Gutachter: Prof. Dr. Ekkehard Sachs
2. Gutachter: Prof. Dr. John A. Burns

Tag der mündlichen Prüfung: 18. Juli 2005

Acknowledgements

I would like to thank Prof. Ekkehard Sachs, my supervisor, for his many suggestions and constant support during this research. I am also thankful to Prof. John A. Burns for making a special trip to Trier to serve as examiner.

During work on this thesis I was employed as a research associate for Prof. Dieter Sadowski in the Dept. of Economics at the University of Trier. A special measure of gratitude is due Prof. Sadowski for providing ample research time and moral encouragement. On a related note, I wish to thank my colleagues at the University of Trier and the Institute for Labor Law and Labor Relations (IAAEG), especially Dipl.-Vw. Tanja Machalet and Anne Besslich, for maintaining a friendly working environment.

I am deeply indebted to my wife, Petra, and our children, Helena, Michael, Matthew and Hannah for their patience and support during the many long hours I spent working on this thesis.

Finally, I wish to thank Dipl.-Wirtsch.-Math. Bärbel Bender, who read through the final manuscript and made many helpful suggestions.

Trier, Germany
May 20, 2005

Bret Kragel

Contents

Chapter 1

Introduction

1.1 Motivation

The usual numerical approaches to the solution of partial differential equations, such as finite elements and finite differences, can lead to large algebraic systems that are difficult to handle computationally. This is especially true of time-dependent processes, in which the systems resulting from the spatial discretization must be solved at each time step. These difficulties have led researchers to develop low-order models, such as the *proper orthogonal decomposition (POD)*, that can describe the system of interest using a relatively small number of degrees of freedom. POD has been used successfully in a wide range of applications (cf. [3, 6, 7, 9, 41, 47, 72, 103, 126, 127, 149]) and continues to be an area of intense research interest (cf. [71, 94, 97, 98, 119, 120, 148, 147, 146]).

One of the more promising applications of the proper orthogonal decomposition is the optimal control of systems in which the dependence of the *system state* on *control variables* — often denoted by ϕ and g, respectively — is described by a partial differential equation. The goal of the optimal control problem is the minimization of a *cost functional* $\mathcal{J}(g) = \mathcal{J}(\phi(g), g)$, which may take any number of forms depending on the desired control action (see Section 2.2). In principle, one would like to replace the state equation(s) $\phi(g)$ with a POD-based model $\hat{\phi}(g)$ that can be solved with much less computational effort than needed for the high-order solution process. If the model $\hat{\phi}(g)$ accurately represents the action of system $\phi(g)$ throughout the optimization process, then the model optimal control problem $\hat{\mathcal{J}}(g) = \mathcal{J}(\hat{\phi}(g), g)$ can in theory be solved cheaply. In reality, some difficulties arise.

The main difficulty with POD-based methods as applied to optimal control problems is the issue of model fidelity. POD-based models for the state equations are derived from data provided by experiment or direct numerical simulation (DNS), making the ability of these models to accurately represent the system state dependent on the underlying problem data (e.g., initial and boundary conditions and Reynolds number). In order to guarantee fidelity, the model must be periodically reset during the optimization process (cf. [3, 41, 55, 56]). This requires renewed – and computationally expensive – high-order solution of the state equations.

Fahl [41] attempted to solve this quandary by embedding the optimization process in a *trust-region POD (TRPOD)* approach. The TRPOD method begins with a high-order solution of the state equations using some initial control g_0. A POD-based model derived from the high-order solution is then used to solve the optimal control problem, generating a potential optimal control g_{new}, with the set of admissible controls limited to some "trusted" neighborhood $\|g_{\text{new}} - g_0\| \leq \triangle_0$ of g_0, where $\triangle_0$ is known as the *trust-region radius.* A new high-order solution is subsequently generated using the updated control, and the actual decrease $\mathcal{J}(g_0) - \mathcal{J}(g_{\text{new}})$ in the cost functional achieved using the high-order solver is compared to the decrease $\hat{\mathcal{J}}(g_0) - \hat{\mathcal{J}}(g_{\text{new}})$ predicted by the model. If the ratio

$$\rho = \frac{\mathcal{J}(g_0) - \mathcal{J}(g_{\text{new}})}{\hat{\mathcal{J}}(g_0) - \hat{\mathcal{J}}(g_{\text{new}})}$$

is sufficiently large, the trust-region radius is decreased, left constant or increased depending on the size of ρ, a new POD-based model is generated using the updated high-order solution of the state equation, and the control problem is resolved using the new POD-based model. If ρ is too small, the new high-order solution is rejected, the trust-region radius is decreased and the optimal control problem is resolved using the original POD-based model. This process continues until convergence to a local stationary point is achieved.

Making the usual trust-region assumptions on the cost functional $\mathcal{J}$ and the model $\hat{\mathcal{J}}$ (cf. [35, 138]), along with conditions needed to ensure consistency between the gradients of the objective and model problems (see also [19, 26, 27]), Fahl was able to prove convergence of the TRPOD method to a local stationary point.

The TRPOD approach is quite sensible, but still requires repeated high-order solution of the state equations. In this sense, it would be advantageous if one could acquire the information for POD basis generation and augmentation with less computational effort. For data generated by numerical simulation this might be accomplished by using coarser grids to compute approximate solutions, which can then be used as starting points for optimization on finer grids (cf. [10, 11, 15, 58]). Since the coarser grids may still require considerable computational effort, it makes sense to extend the POD approach to the coarser grids as well. We propose to significantly improve the TRPOD approach by using recursive trust-region methods, recently introduced by Gratton et al. [57], combined with POD methods on coarse and fine grids. These methods have the potential to reduce the computational effort required for solving optimal control problems such as those discussed above, while maintaining the guaranteed convergence of trust-region methods. As far as we know, this work is the first attempt to combine the recursive trust-region methodology with POD-based models derived from numerical data at various mesh refinement levels.

The idea outlined above has immediate intuitive appeal; however, some theoretical and practical difficulties quickly become apparent. On the theoretical side, the convergence proof for the recursive trust-region methodology [57] is restricted to quadratic model functions of the form

$$m_k(x_k + s) = f(x_k) + (\nabla_x f(x_k), s) + \frac{1}{2}(s, \nabla_{xx} f(x_k)\, s),$$

where $f : \mathbb{R}^n \mapsto \mathbb{R}$ is the twice continuously differentiable objective function, and $x_k \in \mathbb{R}^n$ is the current iterate in the k-th optimization step. Nevertheless, we believe the theory can be extended to more general model functions as was done for nonrecursive trust-region procedures by Toint [138], Carter [26] and Conn et al. [35], and applied by Fahl to the TRPOD approach as described above (see also [19, 25, 27, 87, 88]).

On the practical side, it is well-known that mixed convection-diffusion problems with dominant convection may suffer from numerical instability problems (cf. Chapter 3) that can lead to oscillations in the solution and failure of the high-order numerical solution procedure to converge. This problem can be eliminated either by resorting to finer grids, which we wish to avoid, or by utilizing some sort of stabilization, e.g., upwinding or streamline diffusion. Though both techniques are sufficient to stabilize the solution, we are especially interested in the *streamline diffusion finite element method (SDFEM)*, because of its better theoretical convergence properties and because it is naturally formulated as a Petrov-Galerkin method, which allows easy incorporation into the POD-based model. It turns out though, that these stabilization procedures result in POD basis functions that are incompatible with the standard POD-based reduced-order model, as we will demonstrate in Section 5.1. As a remedy, in Section 5.2 we suggest and experiment with approaches for incorporating the stabilization action into the reduced-order model. We show that the resulting procedure leads to a POD-based model that is tuned to the high-order solver, so that models derived from rougher discretizations can be used with confidence. As far as we know, this thesis introduces the idea of adding stabilization from the high-order numerical solution process to the POD-based model.

1.2 Outline

In this section, we present a structural overview of the contents of this thesis.

Section 1.3 reviews some standard and specialized function spaces and notation we will need later for the theoretical treatment of the Navier-Stokes equations and related problems. Section 1.4 closes the introductory chapter by extending the concept of condition – well-known for matrices themselves – to the eigenvalues and eigenvectors of Hermitian matrices.

In Chapter 2 we present an optimal control problem that will serve as the basis for numerical testing of our POD methods in later chapters. We will concentrate on the velocity-tracking problem for fluid flow in a bounded two-dimensional region of $\mathbb{R}^2$, in which the system state, denoted here by $\mathbf{u}$, is determined from the boundary control $\mathbf{g}$ by solution of the nonstationary viscous incompressible Navier-Stokes equations

$$\begin{aligned} \mathbf{u}_t - \nu\triangle\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla p = \mathbf{0} \quad & \text{in } (0,T]\times\Omega \\ \nabla\cdot\mathbf{u} = 0 \quad & \text{in } (0,T]\times\Omega \\ \mathbf{u} = \mathbf{g} \quad & \text{on } [0,T]\times\Gamma \\ \mathbf{u}(0,\mathbf{x}) = \mathbf{u}_0(\mathbf{x}) \quad & \text{in } \Omega, \end{aligned} \tag{1.1}$$

where $\mathbf{u}_0$ is the initial condition, Γ denotes the boundary of Ω and $[0,T]$ is some time period of interest. The objective of the velocity tracking problem is to drive a candidate velocity field to some given target velocity $\mathbf{u}_d \in (0,T]\times\Omega$ by appropriately controlling the velocity along a portion of the flow domain boundary. This objective is reflected in the minimization of the cost functional

$$\mathcal{J}(\mathbf{u}(\mathbf{g}),\mathbf{g}) = \frac{1}{2}\left\|\mathbf{u}-\mathbf{u}_d\right\|^2_{L^2(0,T;L^2(\Omega))}. \tag{1.2}$$

Section 2.1 introduces a weak formulation of (1.1) and gives conditions on $\mathbf{g}$, $\mathbf{u}_0$ and Γ that ensure the existence of a unique weak solution of (1.1). In Section 2.2 we consider an abstract formulation of the velocity tracking problem due to Gunzburger/Manservisi [65], and state a theorem from the same guaranteeing the existence of a solution to the problem.

We close Chapter 2 with a description of FEATFLOW, a finite element solver developed by Turek [141] for the numerical solution of the incompressible Navier-Stokes equations in two and three dimensions. The discretization process is separated in space and time. Semi-discretization in time leads to a generalized stationary Navier-Stokes problem with prescribed boundary values for each time step. The stationary problems are discretized using piecewise rotated bilinear shape functions on a quadrilateral mesh. The solution procedure for the resulting algebraic systems is described briefly in Section 2.3.3.

Chapter 3 introduces stabilization methods required for the numerical solution of convection-diffusion problems with dominant convection. Section 3.1 begins with a simple example illustrating how instabilities of purely numerical character can arise for such problems. The example is then supplemented by a more detailed examination of a situation involving the discretization of the linearized Navier-Stokes equations. In the remainder of the chapter we discuss some methods for stabilizing the numerical solution procedure, beginning in Section 3.2 with a streamline diffusion method for a linear convection-diffusion problem. We study the application of streamline diffusion to convection-diffusion problems for two reasons. First, although most of the work in this thesis will concern the Navier-Stokes equations, the modified POD methods we introduce in Chapter 5 are also applicable to convection-diffusion problems. Second, application of streamline diffusion to such problems is conceptually and technically simpler than is the case for Navier-Stokes equations. In this way, we can introduce the main features of streamline diffusion without turning a short chapter into a long one. In Section 3.3 we extend the discussion to the Navier-Stokes equations by describing an interesting streamline diffusion formulation from

Tobiska/Verfürth [137] for the generalized Navier-Stokes equations. This is followed by a detailed description of the streamline diffusion method implemented in the FEATFLOW solver mentioned earlier.

Chapter 4 is dedicated to an extensive discussion of proper orthogonal decomposition. Generally speaking, the POD method uses data – a so-called snapshot ensemble $u_i \in H$, $i = 1, \ldots, n$ from some solution space H – generated either experimentally or from the numerical solution of the system of interest, to build an orthonormal system of basis elements that reflect the salient characteristics of the expected solution. In the context of fluid flow problems, the dynamical system of interest is subsequently projected onto the POD basis – the Galerkin POD method – to derive a POD-based model for the system.

We begin the discussion of POD methods in Section 4.1, where the POD basis is derived from a minimization problem based on the approximation error between the snapshot ensemble and the projection of the snapshots onto the POD basis – a procedure that is well-known from the canonical POD literature (cf. [13, 72, 126]). In Section 4.1.1 we introduce methods for computing the POD basis from the snapshot ensemble. We will use the so-called method of snapshots introduced by Sirovich [126] that allows computation of the POD basis vectors ψ_i, $i = 1, \ldots, p$, by solving the eigenvalue problem for the correlation matrix $\mathcal{K}$, defined by $\mathcal{K}_{ij} = (u_i, u_j)$, $1 \leq i, j \leq n$, and setting

$$\psi_i = \frac{1}{\sqrt{\lambda_i}} \sum_{j=1}^{n} v_{ij} u_j, \quad i = 1, \ldots, p \tag{1.3}$$

where $p \leq n$ is the rank of $\mathcal{K}$, λ_i, $i = 1, \ldots, p$, are the positive eigenvalues of $\mathcal{K}$ and v_i, $i = 1, \ldots, p$, the corresponding eigenvectors with components v_{ij}, $j = 1, \ldots, n$.

In the last part of Section 4.1 we note some of the interesting properties of the POD basis, emphasizing its superiority to other linear decompositions of the space spanned by the snapshots. Finally, we note that the eigenvalues in (1.3) often decline rapidly, with a large measure of the information or energy in the snapshots captured by a small number of eigenvalues. In this case, the POD basis can be truncated, resulting in a significant decrease in model order with little loss in fidelity.

As discussed above, we are interested in building and using POD-based models generated from numerical simulations on meshes of various courseness/fineness. In this sense, it would be comforting to have some idea how well the POD-based models derived from coarse meshes approximate the models derived from finer meshes. Though it appears that little research has been done in this area to date, Kunisch, Volkwein and Hinze ([91, 93, 92, 147, 71]) have produced some initial results, which we review in Section 4.2.

Specifically, we are interested in three separate but related phenomena:

1. The dependence of the POD basis on the intervals at which the POD snapshots are taken (Section 4.2.1).

2. The dependence of the POD basis on the spatial discretization of the domain Ω (Section 4.2.2).

3. Error estimates for Galerkin POD methods (Section 4.2.3).

In Section 4.2.1 we state a proposition from Kunisch/Volkwein [91, 93] on the convergence of the POD eigenvalues as the time intervals between the POD snapshots become smaller, or equivalently, as the number of snapshots becomes larger. Section 4.2.2 reviews some results from Volkwein concerning the dependence of the correlation matrix $\mathcal{K}$ on the finite element mesh parameter h. Section 4.2.3 gives some error bounds from Kunisch/Volkwein [92] for the Galerkin POD procedure.

In Section 4.3 we perform extensive numerical testing of the results of Section 4.2 at Reynolds numbers of $Re = 100$, $Re = 400$, $Re = 10,000$ and $Re = 20,000$. These results indicate that at lower Reynolds numbers POD-based models derived from data generated on course meshes will likely provide good approximations for POD-based models derived from finer meshes. This is not so clear at $Re = 10,000$ and $Re = 20,000$, suggesting the need for trust-region methodology to guide the optimization process.

Chapter 5 deals directly with POD-based models for the Navier-Stokes equations. In the Navier-Stokes context, the POD method assumes that the velocity can be written as a linear combination of the POD basis functions. In Section 5.1 a Galerkin POD-based reduced-order model is formulated for the driven cavity problem described in Section 2.2.3 by expanding the velocity field into a linear combination of the POD basis of Section 4.1, resulting in a model of the form

$$\mathbf{u}(t,\mathbf{x}) = \mathbf{u}_n(\mathbf{x}) + \gamma(t)\mathbf{u}_c(\mathbf{x}) + \sum_{i=1}^{m} y_i(t)\boldsymbol{\Psi}_i(\mathbf{x}), \tag{1.4}$$

where $\mathbf{u}$ denotes the system velocity, $\mathbf{u}_c$ is a reference velocity field, γ is the boundary control along the top of the cavity, $\mathbf{u}_n$ is the average of the snapshots and the coefficients (or modes) y_i, $i = 1, \ldots, m$, are determined from a system of ordinary differential equations arising from the projection of the Navier-Stokes equations onto the POD basis.

Numerical testing in Section 5.1 reveals that the streamline diffusion needed to stabilize the high-order Navier-Stokes solution procedure used for the generation of the snapshot ensemble results in POD basis functions that are incompatible with the standard POD-based reduced-order model for the Navier-Stokes equations. Consider, for instance, Figure 1.1 on Page 7. The graphic on the left compares the first mode y_1 of a POD-based model derived from numerical data at Reynolds number $Re = 10,000$, where streamline diffusion was used for stabilization, with the direct projection of the snapshot ensemble onto the corresponding POD basis vector. As explained in Section 5.1, the curves should be superimposed on one another if the POD-based model is accurate; however, this is clearly not the case in the figure. This dissonance can even result in the failure of the ordinary differential equation solver to converge, as illustrated in the right graphic of Figure 1.1, making it impossible to compute the modes y_i in (1.4). Clearly, the POD-based model (1.4) is useless if the modes cannot be determined.

As a remedy, in Section 5.2 we suggest and experiment with approaches for incorporating the stabilization action into the reduced-order model. We show that the resulting procedure, which we call *streamline diffusion POD (SDPOD)*, leads to a POD-based model that is tuned to the high-order Navier-Stokes solver, so that models derived from rougher discretizations can be used with confidence. For use in the optimization process, we derive gradient information for our POD-based models in Section 5.3 using the adjoint method.

We begin Chapter 6 by presenting a recursive multilevel trust-region method recently suggested and analyzed by Gratton et al. [57]. The method is constructed using quadratic model functions, so the method and the accompanying theoretical analysis are not directly applicable to POD-based model functions. Nevertheless, since nonrecursive trust-region methods have successfully been adapted to more general model functions, including POD-based models (cf. [26, 35, 41, 138]), we are hopeful that the theoretical results from the recursive procedure can be adapted to the SDPOD-based models as well. We limit ourselves to numerical tests at Reynolds numbers of $Re = 400$ and $Re = 10,000$. The results, which are detailed in Section 6.2, are encouraging.

Appendix A provides a detailed derivation of the POD-based models of Chapter 5, showing how they can be computed using the finite element basis functions of the high-order Navier-Stokes solver. The derivations are relatively straightforward, but laborious.

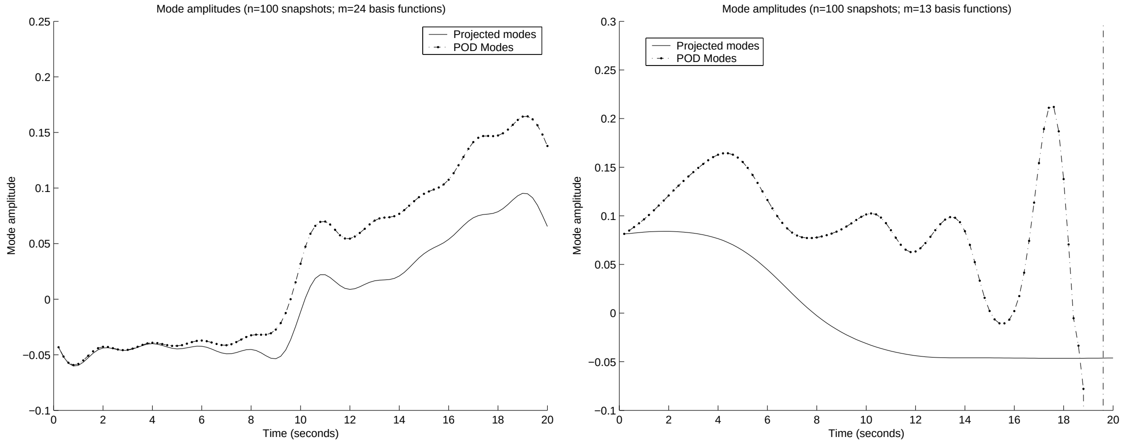


Figure 1.1: Projected and predicted modes at $Re = 10,000$ using streamline diffusion in the Navier-Stokes solver. In the graphic on the left, the POD modes are inaccurate when generated on a 49×49 mesh. The graphic on the right shows how the ODE solver fails to converge for snapshots generated on a 13×13 mesh.

1.3 Notation and Function Spaces

In this section we present some notation and results from functional analysis that will be used frequently in the sequel. More detailed information and discussion can be found in Adams [2], Ciarlet [32], Dautray/Lions [37] and Lions/Magnaes [100], and the first chapters of the books by Girault/Raviart [51], Grisvard [60], Showalter [125] and Temam [129].

1.3.1 Function Spaces

For a function $u : \mathbb{R}^n \to \mathbb{R}$ and $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}_0^n$, we set

$$|\alpha| = \sum_{i=1}^{n} \alpha_n \quad \text{and} \quad \partial^\alpha u = \frac{\partial^{|\alpha|} u}{\partial x_1^{\alpha_1} \ldots \partial x_n^{\alpha_n}}.$$

We adopt the usual notation for the gradient and Laplace operators

$$\nabla u = (\partial u/\partial x_1, \ldots, \partial u/\partial x_n) \quad \text{and} \quad \Delta u = \sum_{i=1}^{n} \frac{\partial^2 u}{\partial x_i^2},$$

respectively. For vector-valued functions $\mathbf{u} = (u_1, \ldots, u_n)$ we define the divergence operator $\nabla\cdot$ by

$$\nabla \cdot \mathbf{u} = \sum_{i=1}^{n} \frac{\partial u_i}{\partial x_i}$$

so that $\nabla \cdot (\nabla u) = \Delta u$ for real-valued functions.

1.3.2 Domain Regularity

We denote by Ω a bounded connected open subset of $\mathbb{R}^n$ with boundary Γ. The boundary Γ is called Lipschitz-continuous (or Lipschitz), if Γ is locally the graph of a Lipschitz function. We say Γ is of class C^r, with $r \geq 1$ to be specified, if Γ is a manifold of dimension $n-1$ of class C^r. The boundary is of class $C^{r,1}$ if it is of class C^r and the derivative of order r is Lipschitz continuous. In all cases we assume that Ω is locally on one side of Γ.

In general, we would like to work with Lipschitz-continuous boundaries as these allow domains with corners, which are standard in nearly all applications; however, such domains are insufficient for rigorous study of boundary value problems with nonhomogeneous boundary conditions, which require more regularity on the boundary. Where necessary, we approximate the domain of interest with a domain meeting the regularity demands of the theoretical treatment.

1.3.3 Sobolev Spaces

For any open bounded subset $\Omega \subseteq \mathbb{R}^n$, we denote by $L^p(\Omega)$ the space of $\mathbb{R}$-valued functions on Ω, for which $\left(\int_\Omega |u|^p \, d\mathbf{x}\right)^{1/p} < \infty$. For $m \in \mathbb{N}_0$ and $1 \leq p \leq \infty$, we denote the usual Sobolev spaces by

$$W^{m,p} = \{u \in L^p(\Omega) \mid \partial^\alpha u \in L^p(\Omega) \quad \forall\ |\alpha| \leq m\}.$$

These become Banach spaces when outfitted with the norms

$$\|u\|_{m,p,\Omega} = \left(\sum_{|\alpha| \le m} \int_\Omega |\partial^\alpha u|^p \, d\mathbf{x} \right)^{1/p}, \quad \text{for } p < \infty$$

and

$$\|u\|_{m,p,\Omega} = \max_{|\alpha| \le m} \left(\operatorname*{ess\,sup}_{\mathbf{x} \in \Omega} |\partial^\alpha u| \right), \quad \text{for } p = \infty.$$

We can also define the seminorm

$$|u|_{m,p,\Omega} = \left(\sum_{|\alpha| = m} \int_\Omega |\partial^\alpha u|^p \, d\mathbf{x} \right)^{1/p}, \quad \text{for } p < \infty,$$

with the corresponding modification for $p = \infty$.

Treatment of boundary conditions requires the introduction of Sobolev spaces of fractional order. The notation and theory needed to introduce these spaces coherently is extensive. We refer the reader to the thorough exposition by Grisvard [60, Sections 1.3-1.4], in which the notion of Sobolev spaces is extended to nonintegral values of m.

For $p = 2$ we denote the space $W^{m,2}(\Omega)$ by $H^m(\Omega)$ dropping the subscript $p = 2$ from the notation for the norm and seminorm. With the scalar product

$$(u, v)_{m,\Omega} = \sum_{|\alpha| \le m} \int_\Omega \partial^\alpha u \, \partial^\alpha v \, d\mathbf{x},$$

$H^m(\Omega)$ becomes a Hilbert space. We denote the space of continuous functions defined in Ω by $C^0(\Omega)$ and set

$$C^m(\Omega) = \{u \in C^0(\Omega) \mid \partial^\alpha u \in C^0(\Omega) \quad \forall \, |\alpha| \le m\}.$$

We will often suppress Ω from the notation, as the domain of interest will usually be known. For vector-valued functions $\mathbf{u} : \Omega \to \mathbb{R}^d$, $d \ge 2$, we add a superscript to the notation for the vector-valued counterparts of the function spaces defined above; e.g., $L^2(\Omega)^d$ for the space of square-integrable $\mathbb{R}^d$-valued functions. We shall use $H^{-m}(\Omega)$ to denote the dual space of $H_0^m(\Omega)$, which is normed by

$$\|f\|_{-m,\Omega} = \sup_{\substack{v \in H_0^m(\Omega) \\ v \ne 0}} \frac{(f, v)_{m,\Omega}}{\|v\|_{m,\Omega}}. \tag{1.5}$$

If the domain Ω is connected and bounded in at least one direction, then for each nonnegative integer m, there exists a constant $c = c(m, \Omega) > 0$ such that the Poincaré-Friedrichs inequality

$$\|v\|_{m,\Omega} \le c \, |v|_{m,\Omega} \qquad \forall \, v \in H_0^m(\Omega) \tag{1.6}$$

holds. The inequality (1.6) implies that the mapping $v \mapsto |v|_{m,\Omega}$ is a norm on $H_0^m(\Omega)$, equivalent to $\|\cdot\|_{m,\Omega}$.

We will repeatedly – often without mention – make use of Young's inequality

$$ab \leq \sigma \frac{a^p}{p} + \sigma^{-q/p} \frac{b^q}{q}, \tag{1.7}$$

which holds for all $a, b, \sigma > 0$ and all $p \in (1, \infty)$ with $q = p/(p-1)$.

Specialized Sobolev Spaces

For the derivation of weak or variational formulations for Navier-Stokes problems in two and three dimensions (d=2,3), we will also require the divergence-free spaces

$$\begin{aligned}
\mathbf{H}(\Omega) &= \{\mathbf{v} \in L^2(\Omega)^d \mid \nabla \cdot \mathbf{v} = 0,\ \mathbf{v} \cdot \mathbf{n} = 0 \text{ on } \Gamma\}, \\
\mathbf{V}(\Omega) &= \{\mathbf{v} \in H_0^1(\Omega)^d \mid \nabla \cdot \mathbf{v} = 0\}, \\
\mathbf{W}(\Omega) &= \{\mathbf{v} \in L^2(\Omega)^d \mid \nabla \cdot \mathbf{v} = 0\},
\end{aligned}$$

and the space

$$L_0^2(\Omega) = \{q \in L^2(\Omega) \mid \int_\Omega q \, d\mathbf{x} = 0\},$$

where $\mathbf{n}$ denotes the outward normal at the boundary Γ (see Girault/Raviart [51] or Temam [129] for detailed discussion of these spaces).

For bounded open sets Ω of class C^2, we define the spaces

$$\begin{aligned}
H^2_{\text{curl}}(\Omega) &= \{\mathbf{v} \in H^1(\Omega)^d \mid \nabla \cdot \mathbf{v} = 0, \quad \int_\Gamma \mathbf{v} \cdot \mathbf{n} \, d\mathbf{x} = 0\}, \\
H^1_n(\Gamma) &= \{\mathbf{g} \in H^1(\Gamma)^d \mid \int_\Gamma \mathbf{g} \cdot \mathbf{n} \, d\mathbf{x} = 0\} \text{ and} \\
H^1_{n0}(\Gamma) &= H^1_0(\Gamma) \cap H^1_n(\Gamma).
\end{aligned}$$

Note that $H^2_{\text{curl}}(\Omega)$ is a closed subspace of $H^1(\Omega)^d$, while $H^1_n(\Gamma)$ and $H^1_{n0}(\Gamma)$ are closed subspaces of $H^1(\Gamma)^d$ (cf. Dautray/Lions [37]). We drop the superscript d for the spaces $H^2_{\text{curl}}(\Omega)$, $H^1_n(\Gamma)$ and $H^1_{n0}(\Gamma)$ as we have only limited use for them and the dimension of the space will be clear from the context and their definition in terms of $H^1(\Omega)^d$ and $H^1(\Gamma)^d$.

The Trace Theorem

We now describe the sense in which functions in Sobolev spaces can be restricted to the boundary of the domain. In particular, we wish to know how smooth the boundary data must be in order for a function in $H^m(\Omega)$ to assume this data. Denoting by γ the operator defined by $\gamma u = u \mid_\Gamma$ for u and Γ smooth enough, we have the following result from Grisvard [60, Theorem 1.5.1.2]:

Theorem 1.1. *Let Ω be a bounded open subset of $\mathbb{R}^n$ with a $C^{k,1}$ boundary Γ. Assume that $s - 1/2$ is not an integer, $s \leq k+1$, $s - 1/2 = l + \sigma$, $0 < \sigma < 1$ with $l \in \mathbb{N}_0$. Then the mapping*

$$u \mapsto \left\{ \gamma u, \gamma \frac{\partial u}{\partial \mathbf{n}}, \ldots, \gamma \frac{\partial^l u}{\partial \mathbf{n}^l} \right\},$$

which is defined for $u \in C^{k,1}(\overline{\Omega})$*, has a unique continuous extension as an operator from*

$$H^s(\Omega) \ \textit{onto} \ \prod_{j=0}^{l} H^{s-j-1/2}(\Gamma),$$

with a right continuous inverse. In particular, we have

$$\gamma : H^1(\Omega) \mapsto H^{1/2}(\Gamma).$$

1.3.4 Vector-valued Distributions

For $T > 0$ and some Banach space X with norm $\|\cdot\|_X$, we denote the Bochner space of measureable functions $\mathbf{u} : [0,T] \to X$ for which $\int_0^T \|\mathbf{u}(\tau)\|_X^p \, d\tau < \infty$ by $L^p(0,T;X)$. The space $L^p(0,T;X)$ is itself a Banach space with respect to the norm

$$\|\mathbf{u}\|_{L^p(0,T;X)} = \begin{cases} \left(\int_0^T \|\mathbf{u}(\tau)\|_X^p \, d\tau\right)^{1/p}, & 1 \leq p < \infty \\ \operatorname*{ess\,sup}_{\tau \in (0,T)} \|\mathbf{u}(\tau)\|_X, & p = \infty. \end{cases}$$

Similarly, we denote by $C([0,T];X)$ the space of continuous functions from $[0,T]$ into X, and by $C^m([0,T];X)$ the space of m-times continuously differentiable functions from $[0,T]$ into X. These are likewise Banach spaces for the norms

$$\|\mathbf{u}\|_{C([0,T];X)} = \operatorname*{ess\,sup}_{\tau \in [0,T]} \|\mathbf{u}(\tau)\|_X,$$
$$\|\mathbf{u}\|_{C^m([0,T];X)} = \sum_{i=1}^{m} \left\|\partial^i \mathbf{u}/\partial t^i\right\|_{C([0,T];X)}.$$

We need to extend these spaces somewhat for the theory of Navier-Stokes equations. To this end, we define for $r, s \geq 0$ and $Q = (0,T) \times \Omega$ the anisotropic Sobolev spaces

$$H^{r,s}(Q) = L^2(0,T;H^r(\Omega)) \cap H^s(0,T;L^2(\Omega)),$$

with the norm

$$\|u\|_{H^{r,s}(Q)} = (\|u\|^2_{L^2(0,T;H^r(\Omega))} + \|u\|^2_{H^s(0,T;L^2(\Omega))})^{1/2}.$$

The spaces $H^{r,s}(S)$ and $H^{r,s}(S_c)$ are defined analogously for $S = (0,T) \times \Gamma$ and $S_c = (0,T) \times \Gamma_c$ (cf. Lions/Magnaes [100, Vol. II, Chapter 4]).

1.3.5 Domain Decompositions

For finite element discretizations, we will need appropriate decompositions of the domain Ω into triangles or quadrilaterals. Following Roos et al. [122], we denote by $\mathcal{T}^h$ a family of decompositions of the domain Ω into quasiuniform meshes with polyhedral elements

$T \in \mathcal{T}^h$. Let $\mathcal{E}^h(T)$ denote the set of all edges of an element $T \in \mathcal{T}^h$ with $\mathcal{E}^h = \bigcup_{T \in \mathcal{T}^h} \mathcal{E}^h(T)$, and set

$$h_T = \sup_{\mathbf{x},\mathbf{y} \in T} |\mathbf{x} - \mathbf{y}| \quad \text{for each } T \in \mathcal{T}^h,$$
$$h_E = \sup_{\mathbf{x},\mathbf{y} \in E} |\mathbf{x} - \mathbf{y}| \quad \text{for each } E \in \mathcal{E}^h.$$

Then the quasiuniformity of the triangulation $\mathcal{T}^h$ implies that the ratio h_T/h_E is bounded independently of h, T and E. For any $E \in \mathcal{E}^h$ with $E = T_1 \cap T_2$, $T_1, T_2 \in \mathcal{T}^h$ and $q \in L^2(\Omega)$ with $q\,|_{T_i} \in C(\overline{T_i})$, $i = 1, 2$, let the jump $[q]_E$ and average $A_E q$ across and along an edge $E \in \mathcal{E}^h$ be defined by

$$[q]_E(\mathbf{x}) := \begin{cases} \lim\limits_{t \to +0} q(\mathbf{x} + t\mathbf{n}_E) - \lim\limits_{t \to +0} q(\mathbf{x} - t\mathbf{n}_E) & E \not\subset \Gamma \\ -\lim\limits_{t \to +0} q(\mathbf{x} - t\mathbf{n}_E) & E \subset \Gamma \end{cases}, \tag{1.8}$$

and

$$A_E q(\mathbf{x}) := \begin{cases} \frac{1}{2}\big(\lim\limits_{t \to +0} q(\mathbf{x} + t\mathbf{n}_E) + \lim\limits_{t \to +0} q(\mathbf{x} - t\mathbf{n}_E)\big) & E \not\subset \Gamma \\ \frac{1}{2}\big(\lim\limits_{t \to +0} q(\mathbf{x} - t\mathbf{n}_E)\big) & E \subset \Gamma \end{cases}, \tag{1.9}$$

where $\mathbf{n}_E$ is a normal unit vector on edge E and $\mathbf{x} \in E$. For $E \subset \Gamma$ the orientation of $\mathbf{n}_E$ is outward with respect to Ω, otherwise $\mathbf{n}_E$ has an arbitrary but fixed orientation.

1.4 Condition of Eigenvalues and Eigenvectors

Given a vector norm $\|\cdot\|$ on $\mathbb{C}^n$, the *operator (matrix) norm* of square matrix $A \in \mathbb{C}^{n,n}$ is defined by

$$\|A\| := \max_{\substack{\mathbf{x} \in \mathbb{C}^n \\ \mathbf{x} \neq 0}} \frac{\|A\mathbf{x}\|}{\|\mathbf{x}\|}.$$

For an invertible matrix $A \in \mathbb{C}^{n,n}$ the *condition number* is defined by

$$\operatorname{cond}(A) = \|A\| \left\|A^{-1}\right\| \geq 1.$$

The condition number of matrix is a well-known concept used in studying the sensitivity of solutions of systems of linear equations to perturbations in the system data (matrix coefficients and right-hand side), and can be found in any book on numerical linear algebra (cf. Demmel [38] or Meyer [105]).

When considering the suitability of POD-based surrogate models derived from numerical data acquired from rough spatial discretizations, we will have occasion to examine the stability of the eigenvectors and eigenvalues of symmetric matrices in the presence of perturbations in the matrix coefficients. This will require extension of the concept of condition number to eigenvalues and eigenvectors. Summarizing the discussion in Chatelin

[30], we (very) briefly motivate the use of the term *condition* as applied to eigenvalue problems.

Consider a simple eigenvalue $\lambda \in \mathbb{C}$ of a matrix $S \in \mathbb{C}^{n,n}$ with associated left and right eigenvectors $\phi \in \mathbb{C}^n$ and $\psi \in \mathbb{C}^n$; that is,

$$(A - \lambda I)\phi = \psi^T(A - \lambda I) = 0. \tag{1.10}$$

The eigenvectors ϕ and ψ can be chosen so that $\|\phi\|_2 = \psi^T\phi = 1$. Then the spectral projection operator P, projecting $\mathbb{C}^n$ onto the subspace spanned by ϕ, is given by $P = \phi\psi^H$, and there exists a generalized inverse $S \in \mathbb{C}^{n,n}$ of $(A - \lambda I)$ relative to the spectral projection P:

$$S(A - \lambda I) = (A - \lambda I)S = I - P.$$

Now, let the perturbation H be given, and set $A' := A + H$, where $\varepsilon = \|H\|_2$ is assumed to be small. Then λ and ϕ are approximate eigenelements for A', and the associated residual vector for A' is given by

$$A'\phi - \lambda\phi = (A' - A)\phi = H\phi.$$

The following theorem gives some perturbation bounds for the eigenvalues and eigenvectors of A' (see Chatelin [30, Chapter 1, Theorem 1.7]).

Theorem 1.2. *Set $\varepsilon' = \|H\phi\|_2$. Then $\varepsilon' \leq \varepsilon$, and if ε is small enough, there exists a simple eigenvalue λ' of A' with an eigenvector ϕ' normalized by $\psi^H\phi' = 1$, such that*

$$\lambda' = \lambda + \psi^H H\phi + \mathcal{O}(\varepsilon^2) \text{ and} \tag{1.11}$$

$$\phi' = \phi - SH\phi + \mathcal{O}(\varepsilon^2). \tag{1.12}$$

From (1.11) we get

$$\left|\psi^T H\phi\right| \leq \varepsilon' \|\psi\|_2 \quad \text{and} \quad \left|\lambda' - \lambda\right| \leq \varepsilon' \|\psi\|_2 + \mathcal{O}(\varepsilon^2). \tag{1.13}$$

So, considering the definition of H, we can say that λ is ill-conditioned if $\|\psi\|_2$ is large. In this sense, $\|\psi\|_2$ is a condition number for λ when ψ is normalized by $\psi^H\phi = \|\phi\|_2 = 1$; hence $\|\psi\|_2 \geq 1$. Note that in the symmetric case, we have $\psi = \phi$ and, as a result, $\|\psi\|_2 = \|\phi\|_2 = 1$ in (1.13), meaning that the simple eigenvalues of Hermitian matrices are well-conditioned.

From (1.12) we get

$$\left\|\phi' - \phi\right\|_2 \leq s\varepsilon' + \mathcal{O}(\varepsilon^2),$$

where we have set $s := \|S\|_2$. We see that ϕ is ill-conditioned if s is large, so that s is a condition number for ϕ. Furthermore, it can be shown (cf. [30]) that $s = 1/d(\lambda)$ for Hermitian matrices, where

$$d(\lambda) = \min_{\mu \in \sigma(A)\backslash\lambda} |\lambda - \mu|, \tag{1.14}$$

with $\sigma(A)$ denoting the spectrum of A. Thus, the only source of ill-conditioning for Hermitian matrices with simple eigenvalues is the presence of close eigenvalues. We call $1/d(\lambda)$

the *condition number* for the eigenvectors of Hermitian matrices with simple eigenvalues, the latter being well-conditioned with a condition number of 1.

We note that the above analysis does not apply to multiple eigenvalues; that is, eigenvalues with algebraic multiplicity greater than one. The above results will prove sufficient for our purposes. We refer the reader to [30] for deeper analysis, including the case of multiple eigenvalues (see also Golub [53, Section 7.2] or Meyer [105, Section 7.3]).

Chapter 2

A Flow Control Problem

Though the reduced-order methods presented in detail later in this work are generally adaptable to any type of problem in which the accuracy of the numerical solution is dependent on the order of the discretization, we are especially interested in flow control problems for which the system state is described by partial differential equations of mixed convection-diffusion type. Such problems suffer from certain numerical difficulties on coarse grids making them ideal for robust testing of our methods. In the following, we discuss flows governed by the especially challenging nonstationary viscous incompressible Navier-Stokes equations, also known as the evolution Navier-Stokes equations. These equations describe the state of the optimal control problem we shall use in our numerical investigations.

Optimal control problems involving the Navier-Stokes equations have been studied extensively, resulting in a vast literature on the subject (cf. Gunzburger [61], Sritharan [128] and Gad-al-Hak et al. [49]). In particular, optimal control problems for the stationary Navier-Stokes problem with boundary controls have been investigated, e.g., by Hou, Gunzburger and Svobodny [75, 74], Burkardt/Peterson [23], Desai/Ito [39], Heinkenschloss [69] and Hou/Ravindran [76, 77]. The more difficult and interesting case of flow control problems involving the nonstationary Navier-Stokes equations with distributed and boundary controls has been considered, e.g., by Abergel and Temam [1], Fattorini/Sritharan [42, 43], Manservisi [104], Fursikov et al. [48], Berggren [12], Gunzburger/Manservisi [63, 65, 64], Hinze/Kunisch [70], Bewley et al. [18, 17, 16], Li et al. [99] and Ulbrich [143].

In Section 2.1 we formulate the Dirichlet problem for the Navier-Stokes equations, introduce the functional framework needed for the derivation of variational formulations of the Navier-Stokes equations and discuss the existence, uniqueness and regularity of solutions. Section 2.2 presents a control problem – the driven cavity problem – that will serve as the model problem for numerical testing throughout this thesis. In Section 2.3, we present and discuss in some detail the software package we will utilize to generate numerical solutions of the driven cavity problem.

2.1 The Navier-Stokes Equations

Consider a bounded domain $\Omega \subset \mathbb{R}^2$ with boundary Γ of class C^2 and $T > 0$. Assuming only Dirichlet boundary conditions, possibly effected only over some subset $\Gamma_c \subset \Gamma$, the two-dimensional time-dependent Navier-Stokes equations for incompressible viscous fluid flows are given by

$$\begin{aligned} \mathbf{u}_t - \nu\Delta\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla p = \mathbf{f} \quad & \text{in } (0,T]\times\Omega, && (2.1)\\ \nabla\cdot\mathbf{u} = 0 \quad & \text{in } (0,T]\times\Omega, && (2.2)\\ \mathbf{u} = \mathbf{g} \quad & \text{on } [0,T]\times\Gamma_c, && (2.3)\\ \mathbf{u} = 0 \quad & \text{on } [0,T]\times(\Gamma\setminus\Gamma_c), && (2.4)\\ \mathbf{u}(0,\mathbf{x}) = \mathbf{u}_0(\mathbf{x}) \quad & \text{in } \Omega, && (2.5) \end{aligned}$$

where $\mathbf{u}(t,\mathbf{x})$ and $p(t,\mathbf{x})$ denote the unknown two-dimensional velocity field and the pressure, respectively. The known problem data of this initial boundary-value problem are the body force per unit mass $\mathbf{f}(t,\mathbf{x})$, the kinematic viscosity $\nu > 0$, the initial velocity $\mathbf{u}_0(\mathbf{x})$, and the boundary velocity $\mathbf{g}(t,\mathbf{x})$. Note that the pressure p in (2.1) can be determined only up to a constant. This constant can be fixed by choosing a pressure p whose mean value is zero, i.e., $\int_\Omega p\, d\mathbf{x} = 0$.

In view of the incompressibility condition (2.2) and in order to obtain the appropriate regularity for the solution of the Navier-Stokes system, we require the control $\mathbf{g}$ to effect zero mass flow across the boundary and match the initial flow $\mathbf{u}_0$ at time $t = 0$. To this end, we assume the compatibility conditions

$$\int_{\Gamma_c} \mathbf{g}\cdot\mathbf{n}\, d\mathbf{x} = 0, \tag{2.6}$$

where $\mathbf{n}$ is the unit outward normal vector on Γ, and

$$\mathbf{g}\mid_{t=0} = \mathbf{u}_0 \mid_{\Gamma_c} . \tag{2.7}$$

The two-dimensional velocity field $\mathbf{u}$ and the pressure field p in the *momentum equation* (2.1) are coupled through the *incompressibility constraint* or *continuity equation* (2.2), where the condition

$$\nabla\cdot\mathbf{u} = \frac{\partial}{\partial x_1}u_1 + \frac{\partial}{\partial x_2}u_2 = 0$$

describes the conservation of mass. The term $\mathbf{u}\cdot\nabla\mathbf{u}$ in the momentum equation is known as the *convective part*, and is defined by

$$\mathbf{u}\cdot\nabla\mathbf{u} = \begin{pmatrix} u_1\frac{\partial}{\partial x_1}u_1 + u_2\frac{\partial}{\partial x_2}u_1 \\ u_1\frac{\partial}{\partial x_1}u_2 + u_2\frac{\partial}{\partial x_2}u_2 \end{pmatrix}.$$

2.1.1 Weak Formulation of the Navier-Stokes Problem

To derive an appropriate weak or variational formulation for the problem (2.1)-(2.5), we will use the function spaces of Section 1.3. The usual bilinear and trilinear forms associated with the d-dimensional Navier-Stokes equations are defined by

$$a(\mathbf{u},\mathbf{v}) = \sum_{i=1}^{d} \int_{\Omega} \nabla u_i \cdot \nabla v_i \, d\mathbf{x} \qquad \forall\, \mathbf{u},\mathbf{v} \in H^1(\Omega)^d, \tag{2.8}$$

$$b(\mathbf{u},q) = -\int_{\Omega} q \nabla \cdot \mathbf{u} \, d\mathbf{x} \qquad \forall\, \mathbf{u} \in H^1(\Omega)^d,\ \forall\, q \in L^2(\Omega) \tag{2.9}$$

and

$$n(\mathbf{u},\mathbf{v},\mathbf{w}) = \int_{\Omega} (\mathbf{u} \cdot \nabla \mathbf{v}) \cdot \mathbf{w} \, d\mathbf{x} \qquad \forall\, \mathbf{u},\mathbf{v},\mathbf{w} \in H^1(\Omega)^d. \tag{2.10}$$

2.1.2 Homogeneous Boundary Conditions

We begin by considering Problem 2.1 with the homogeneous boundary conditions

$$\mathbf{g}(t,\mathbf{x}) = 0 \quad \text{for } t \in [0,T], \mathbf{x} \in \Gamma \tag{2.11}$$

(equivalently $\Gamma_c = \varnothing$). In the Navier-Stokes context, this corresponds to the case in which the flow is driven exclusively by body forces. Though less interesting than the case of boundary-driven flow, the homogeneous boundary conditions are easier to handle mathematically and existence and uniqueness results for this problem are well-known in the canonical Navier-Stokes literature (cf. Ladyzhenskaya [95], Girault/Raviart [50] or Temam [131, 129]).

By multiplying both sides of (2.1) by a divergence-free test function $\mathbf{v} \in \mathbf{V}$ and integrating over Ω, we derive the following variational or weak formulation for the problem (2.1)-(2.5) with $\Gamma_c = \varnothing$, for which $d = 2$.

Problem 2.1. *Find* $\mathbf{u} \in L^2(0,T;\mathbf{V})$ *such that*

$$(\mathbf{u}_t,\mathbf{v}) + \nu a(\mathbf{u},\mathbf{v}) + n(\mathbf{u},\mathbf{u},\mathbf{v}) = (\mathbf{f},\mathbf{v}) \quad \forall\, \mathbf{v} \in \mathbf{V} \tag{2.12}$$

$$\mathbf{u}(0) = \mathbf{u}_0. \tag{2.13}$$

Note that the pressure has been eliminated by embedding the incompressibility constraint into the space $\mathbf{V}$ of test functions and using integration by parts so that $(\nabla p, \mathbf{v}) = -(p, \nabla \cdot \mathbf{v}) = 0$ for all $\mathbf{v} \in \mathbf{V}$. As customary, we call a function $\mathbf{u} \in L^2(0,T;\mathbf{V})$ that satisfies (2.12)-(2.13) a *weak solution* of the homogeneous Navier-Stokes problem (2.1)-(2.5) with $\Gamma_c = \varnothing$.

For Problem 2.1 we have the following result from Temam [130, Theorem III.2.1], which holds for a Lipschitz-continuous boundary Γ.

Theorem 2.2 (Temam). *Given* $\mathbf{f} \in \mathbf{H}$ *and* $\mathbf{u}_0 \in \mathbf{H}$ *there exists a unique solution of Problem 2.1 satisfying*

$$\mathbf{u} \in L^2(0,T;\mathbf{V}) \cap C([0,T];\mathbf{H}) \quad \forall T > 0. \tag{2.14}$$

Furthermore, $\mathbf{u}$ *is analytic in* t *with values in* $H^2(\Omega)^2 \cap \mathbf{V}$ *for* $t > 0$*, and the mapping*

$$\mathbf{u}_0 \to \mathbf{u}(t)$$

is continuous from $\mathbf{H}$ *into* $H^2(\Omega)^2 \cap \mathbf{V}$*,* $\forall\, t > 0$*. Finally, if* $\mathbf{u}_0 \in \mathbf{V}$*, then*

$$\mathbf{u} \in L^2(0,T;H^2(\Omega)^2 \cap \mathbf{V}) \cap C([0,T];\mathbf{V}) \quad \forall T > 0. \tag{2.15}$$

Theorem 2.2 presents some difficulties from a practical point of view due to the embedding of the incompressibility constraint in the function space $\mathbf{V}$. In principle, one could discretize the divergence-free space $\mathbf{V}$ using solenoidal elements (cf. Cuvelier et al. [36]); however, such elements are difficult to construct and program, so that it is customary to use separate spaces for the velocity and pressure as described in the next section.

2.1.3 Inhomogeneous Boundary Conditions

As stated above, the divergence-free function spaces are not practical for numerical approximations. For this reason, we shall extend the results of Theorem 2.2 for nonhomogeneous boundary conditions and $\mathbf{f} = 0$ by considering the following mixed finite element formulation for the Navier-Stokes equations (2.1)-(2.5), which can be found in Manservisi [104] or Gunzburger/Manservisi [65].

Problem 2.3. *Find* $\mathbf{u} \in L^2(0,T;H^1(\Omega)^2)$ *and* $p \in L^2(0,T;L^2_0(\Omega))$ *such that*

$$(\mathbf{u}_t, \mathbf{v}) + \nu a(\mathbf{u},\mathbf{v}) + n(\mathbf{u},\mathbf{u},\mathbf{v}) + b(\mathbf{v},p) = 0 \quad \forall\, \mathbf{v} \in H^1(\Omega)^2_0 \tag{2.16}$$

$$b(\mathbf{u},q) = 0 \quad \forall\, q \in L^2_0(\Omega) \tag{2.17}$$

$$(\mathbf{u},\mathbf{s})_\Gamma = (\mathbf{g}(t,\mathbf{x}),\mathbf{s})_{\Gamma_c} \quad \forall \mathbf{s} \in H^{-1/2}(\Gamma)^2, \tag{2.18}$$

$$\mathbf{u} = 0 \quad \mathbf{x} \in \Gamma \setminus \Gamma_c \tag{2.19}$$

$$\mathbf{u}(0,\mathbf{x}) = \mathbf{u}_0(\mathbf{x}), \tag{2.20}$$

where $\mathbf{g} \in H^{1,1}(S_c) \cap L^2(0,T;H^1_{n0}(\Gamma_c))$ *and* $\mathbf{u}_0 \in H^2_{curl}(\Omega)$*.*

We call a solution pair $(\mathbf{u},p) \in L^2(0,T;H^1(\Omega)^2) \times L^2(0,T;L^2_0(\Omega))$ of Problem 2.3 a *weak solution* of the Navier-Stokes equations with $\Gamma_c \neq \varnothing$.

If $\mathbf{u}$ is a solution of Problem 2.1 then it is also a weak solution of Problem 2.3. Conversely, if $\mathbf{u}$ satisfies Problem 2.3 then it also satisfies Problem 2.1 in the distributional sense on $(0,T)$. Moreover, if $\mathbf{g}$ and $\mathbf{u}_0$ are given as above, then it can be shown (cf. Dautray/Lions [37]) that there exists a unique weak solution $(\mathbf{u},p)$ of Problem 2.3 such that $\mathbf{u} \in L^\infty(0,T;\mathbf{W}) \cap L^2(0,T;H^1(\Omega)^2)$ and $\mathbf{u}_t \in L^2(0,T;H^{-1}(\Omega)^2)$; that is, it is a.e. equal to a continuous function.

The following result is proved in Manservisi [104, Theorem 5.3].

Theorem 2.4 (Manservisi). *Let $\Omega \subset \mathbb{R}^2$ be of class C^2, and let $\mathbf{g} \in H^{1/2,1}(S)$ satisfy the compatibility conditions*

$$\int_\Gamma \mathbf{g} \cdot \mathbf{n} \, d\mathbf{x} = 0 \tag{2.21}$$

and

$$\mathbf{g}(0, \mathbf{x}) = \mathbf{u}_0 \mid_\Gamma . \tag{2.22}$$

Then there exists a unique $\mathbf{u} \in L^2(0,T; H^1(\Omega)^2) \cap L^\infty(0,T; L^2(\Omega)^2)$ and a $p \in L^2(0,T; L^2_0(\Omega))$ that solve the nonhomogeneous Navier-Stokes problem

$$\begin{aligned} (\mathbf{u}_t, \mathbf{v}) + \nu a(\mathbf{u}, \mathbf{v}) + n(\mathbf{u}, \mathbf{u}, \mathbf{v}) + b(\mathbf{v}, p) = 0 \quad &\forall \, \mathbf{v} \in H^1_0(\Omega)^2 \\ b(\mathbf{u}, q) = 0 \quad &\forall \, q \in L^2_0(\Omega) \\ \mathbf{u} = \mathbf{g}(t, \mathbf{x}) \quad &\forall \mathbf{x} \in \Gamma \\ \mathbf{u}(0, \mathbf{x}) = \mathbf{u}_0(\mathbf{x}) & \end{aligned} \tag{2.23}$$

for almost all $t \in (0, T)$. Moreover,

$$\|\mathbf{u}\|^2_{L^2(0,T;H^1(\Omega)^2)} + \|\mathbf{u}\|^2_{L^\infty(0,T;L^2(\Omega))} \leq K \, \|\mathbf{g}\|^2_{H^{1/2,1}(S)} , \tag{2.24}$$

where K is independent of $\mathbf{g}$.

The proof of Theorem 2.4 proceeds by proving the existence of a solution $\tilde{\mathbf{u}}$ of a linear Stokes problem with the boundary conditions and initial conditions of (2.23). Setting $\hat{\mathbf{u}} = \mathbf{u} - \tilde{\mathbf{u}}$ leads to a Navier-Stokes problem with homogeneous boundary conditions, which can be shown to be solvable by the usual methods for homogeneous problems. The theorem then follows from $\mathbf{u} = \hat{\mathbf{u}} + \tilde{\mathbf{u}}$.

2.2 A Velocity Tracking Problem with Boundary Control

We introduce in this section our model control problem, the classic driven cavity problem with flow inside the cavity described by the Navier-Stokes equations.

2.2.1 Structure of Flow Control Problems

To help fix notation and assist in the mathematical formulation, we begin with a short review of the general structure of flow control problems as described by Gunzburger [62]. The variables of an optimal flow control problem can generally be divided into two classes, the *state variables* and the *control variables* (or *design parameters*). As one might surmise based on the terminology, the state variables (often denoted by ϕ) describe the system state, which is determined by the action of the control variables. The effect of the control variables on the system state is typically described mathematically in flow problems by *flow equations* of the form $F(\phi, g) = 0$. Some states of interest for systems described by the Navier-Stokes equations are the system velocity $\mathbf{u}$, pressure p, temperature τ and process time T. Some typical controls for Navier-Stokes systems include *boundary controls* (denoted here by g), such as some kind of prescribed boundary velocity or heating controls at the boundary; *distributed controls*, such as heat sources or magnetic fields; and *shape controls*, such as leading or trailing edge flaps in airfoil design, moveable walls, propeller pitch, etc. The type of flow problem (inviscid or viscous flow, compressible or incompressible flow, stationary or time-dependent flow, etc.) that one is working with is reflected in the relationship between the control variables and the system state as described by the flow equations. Often, the problem variables are also bound by some a priori constraints, e.g., physical or budget constraints, which are expressed mathematically by *constraint equations* of the form $C(\phi, g) \leq 0$.

The objective of a flow control problem is expressed mathematically in terms of a *cost functional* $\mathcal{J}(\phi, g)$, which is to be minimized under the constraints described by the flow equations. The cost functional is sometimes written in the form

$$\mathcal{J}(\phi, g) + \beta \left\| g \right\|^{\gamma} ,$$

with the second part of the functional $\beta \left\| g \right\|^{\gamma}$ designed to achieve regularization for the control problem, that is, to balance the control costs against the actual control objective. Judicious choices for the parameters β, γ and the norm on g can simultaneously limit the size of the control and obtain states such that the value of $\mathcal{J}$ is small. Some of the more popular formulations include tracking-type objectives (cf. Gunzberger/Manservisi [65]), drag minimization (cf. Fursikov et al. [48]), vorticity reduction and flow mixture objectives.

We are especially interested in a certain tracking-type objective. Letting $\mathbf{u}$ denote the flow velocity, $\mathbf{u}_d$ some prescribed desired velocity field and $\mathbf{g}$ the boundary velocity, we can formulate the *flow tracking* or *velocity tracking* problem

$$\mathcal{J}(\mathbf{u}, \mathbf{g}) = \frac{1}{2} \left\| \mathbf{u} - \mathbf{u}_d \right\|^2_{L^2(0,T;L^2(\Omega))} , \tag{2.25}$$

where we have assumed some of the notation of the previous section. Note that the boundary control does not appear explicitly in the right-hand side of the objective (2.25); the effect of the boundary control on the objective functional manifests itself indirectly through the flow equations. One could also modify this formulation to match the flow on only part of the domain, on some surface of the domain, or at a particular time point, e.g., the terminal velocity profile.

2.2.2 An Abstract Control Problem

Similar to the situation for the Navier-Stokes system, we wish to ensure in this section that we can formulate a well-posed optimal flow control problem with a tracking-type objective of the form (2.25). This requires specification of the space $\mathbf{U}_d$ of admissible target velocities and the space $\mathbf{A}_d$ of admissible solutions if we are to form a coherent problem. We say that $\mathbf{u}_d$ is in $\mathbf{U}_d$ if

$$\begin{aligned} \mathbf{u}_d = \mathbf{u}_d(t,\mathbf{x}) \in C([0,T];H^1(\Omega)^2) \text{ and} \\ F_{\mathbf{u}_d}(t,\mathbf{x}) \in L^\infty(0,T;L^2(\Omega)^2), \end{aligned} \tag{2.26}$$

where $F_{\mathbf{u}} = \mathbf{u}_t - \nu\Delta\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u}$. The space $\mathbf{A}_d$, consisting of the state and control spaces, can be characterized using the results of Section 2.1.3: Given $\mathbf{u}_0 \in H^2_{\text{curl}}(\Omega)$ and $\mathbf{u}_d \in \mathbf{U}_d$, the triple $(\mathbf{u},p,\mathbf{g})$ is called an *admissible solution* for the optimal control problem if $(\mathbf{u},p,\mathbf{g}) \in L^2(0,T;H^1(\Omega)^2) \times L^2(0,T;L^2_0(\Omega)) \times H^{1,1}(S_c) \cap L^2(0,T;H^1_{n0}(\Gamma_c))$ is a solution of (2.23), the control $\mathbf{g}$ satisfies the compatibility conditions (2.21) and (2.22), and the functional $\mathcal{J}(\mathbf{u},\mathbf{g})$ is bounded.

The optimal control problem can now be formulated as follows (cf. [65]):

Problem 2.5. *Given $\mathbf{u}_0 \in H^2_{curl}(\Omega)$ and $\mathbf{u}_d \in \mathbf{U}_d$, find $(\mathbf{u},p,\mathbf{g}) \in \mathbf{A}_d$ such that the control $\mathbf{g}$ minimizes the cost functional*

$$\mathcal{J}(\mathbf{u},\mathbf{g}) = \frac{\alpha}{2}\int_0^T\int_\Omega (\mathbf{u}-\mathbf{u}_d)^2\,d\mathbf{x}\,dt + \frac{\beta}{2}\int_0^T\int_{\Gamma_c} (|\mathbf{g}|^2 + \beta_1\,|\mathbf{g}_t|^2 + \beta_2\,|\mathbf{g}_{\mathbf{x}}|^2)\,d\mathbf{x}\,dt \tag{2.27}$$

with $\alpha,\beta,\beta_1,\beta_2 > 0$.

As discussed in the previous section, the goal of Problem 2.5 is the minimization of the term involving $(\mathbf{u}-\mathbf{u}_d)^2$, where $\mathbf{u}_d$ is the desired flow field. The second term serves to bound the control function and facilitate the proof of an optimal control. The positive constants β_1 and β_2 are necessary to ensure $\mathbf{g} \in H^{1,1}(S_c)$.

The cost functional (2.27) is clearly bounded below and weakly lower semicontinuous. A detailed proof of the following existence theorem can be found in [65], where similar to Theorem 2.4 the proof is facilitated by first considering for the flow equations the solution $\tilde{\mathbf{u}}$ of a related linear Stokes problem, which is then used to transform the control problem with inhomogeneous flow equations to one with homogeneous flow equations.

Theorem 2.6. *Given $T > 0$ and $\mathbf{u}_0 \in H^2_{curl}(\Omega)$, there exists a solution $(\mathbf{u},p,\mathbf{g}) \in \mathbf{A}_d$ of Problem 2.5.*

2.2.3 Model Control Problem

As a model problem for the methods proposed in this work, we consider the classical problem of driven cavity flow in two dimensions, which has been used extensively in the study of reduced-order models involving the Navier-Stokes equations (see Peterson [114], Ito/Ravindran [79], Allan [5], Fahl [41], Cazemier et al. [28] and Jørgensen et al. [86]).

All simulations will be performed using the two-dimensional driven cavity problem; that is, a fluid-filled square cavity of unit dimension in the two-dimensional plane, bounded by rigid walls at $x_1 = 0$, $x_1 = 1$ and $x_2 = 0$ and open at the top ($x_2 = 1$). The boundary conditions are given by $\mathbf{g} = (g_1,g_2)^T \equiv (0,0)^T$ on the sides and bottom, with the velocity $\mathbf{g} = (g_1,g_2)^T$ along the top of the cavity driving the flow inside. For time-dependent problems, it is necessary for the POD-based model to assume that $g(t, \mathbf{x})$ can be written in the separable form

$$g(t, \mathbf{x}) = \gamma(t)\mathbf{h}(\mathbf{x}). \tag{2.28}$$

The spatial velocity profile along the top of the cavity for our model problem will be $\mathbf{h}(\mathbf{x}) \equiv (1, 0)^T$ representing a purely tangential force along the top of the cavity. The time-dependent boundary profile $\gamma(t)$ and the simulation time T will vary according to our testing requirements and other problem parameters. One such parameter is the Reynolds number Re, defined by $Re = \bar{U}L/\nu$, where $\bar{U}$ is a measure of the mean velocity, L is the characteristic length of the domain and ν is the kinematic viscosity. For the cavity of unit dimension with $\gamma \equiv 1.0$ we have $Re = 1/\nu$.

For convenience, computations will be carried out on uniform meshes of varying fineness (coarseness), ranging from the 4×4 ($h \approx 3.33334 \times 10^{-1}$) mesh shown in Figure 2.1 up to 193×193 ($h \approx 5.20833 \times 10^{-3}$) meshes, where h denotes the mesh width. Results from experiments with nonuniform grids adding refinement in the boundary areas resulted in no substantial differences with the results reported for uniform grids. The finer grids are constructed from the 4×4 grid by connecting the midpoints of the quadrilaterals of each succeeding grid, thus maintaining uniformity and halving the mesh width h at each new refinement level. As described in Section 2.3.2, we will be using quadrilateral elements

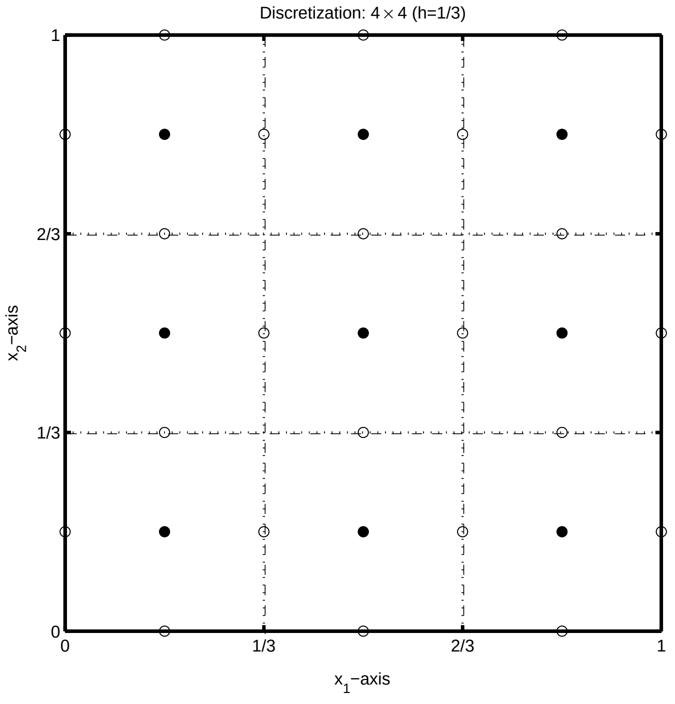


Figure 2.1: The 4×4 discretization of the driven cavity problem. The velocity nodes are marked with small circles on the element edges, the pressure nodes with a solid dot at the element centers.

with rotated bilinear shape functions for the velocity and piecewise constant functions for the pressure approximation. These elements are discussed in greater detail in Section 2.3.2; however, we summarize here in Table 2.1 the resulting complexity data, i.e. the *degrees of freedom* (*dof*), for our specific grids: The nodal values for the pressure are taken at the center of the element (cell, quadrilateral) faces, so that $dof(pressure) = nel$, where nel is the number of elements or quadrilaterals. Since we are using rotated elements, the nodal values for the velocity are taken on the element edges, so that the for the velocity we have $dof(vel) = 2 * \sqrt{nvt} * (\sqrt{nvt} - 1)$, where nvt denotes the number of grid vertices. The total degrees of freedom for each grid is then given by $dof = 2 \cdot dof(vel) + dof(pressure)$.

Mesh	4×4	7×7	13×13	25×25	49×49	97×97	193×193
h	1/3	1/6	1/12	1/24	1/48	1/96	1/192
nel	9	36	144	576	2,304	9,216	36,864
nvt	16	49	169	625	2,401	9,409	37,249
$dof(vel)$	24	84	312	1,200	4,704	18,624	74,112
dof	57	204	768	2,976	11,712	46,464	185,088

Table 2.1: Degrees of freedom information for the discretization of the two-dimensional driven cavity problem at various levels of refinement.

2.3 Numerical Solution of the Navier-Stokes System

For the numerical simulation of the Navier-Stokes equations we utilized FEATFLOW, a finite element solver developed by Turek[1] [141] for the incompressible Navier-Stokes equations. Following Turek [141], [142] and Rannacher [117] and the sources cited therein, we give a brief summary of the solution process.

The discretization process is separated in space and time. Semi-discretization in time leads to a generalized stationary Navier-Stokes equation with prescribed boundary values for each time step. These are discretized in space using finite element methods leading to a coupled system of nonlinear equations, which are solved using a discrete projection scheme that decouples the computation of velocity and pressure.

2.3.1 Temporal Discretization

The usual time-stepping schemes for time-dependent differential equations applied to the Navier-Stokes equations result in a scheme of the following type, where we suppress boundary conditions to simplify notation.

Given the current solution $\mathbf{u}^n := \mathbf{u}(t_n)$ *for velocity and* $p^n := p(t_n)$ *for pressure at time* t_n*, and the time step* $\tau = t_{n+1} - t_n$*, solve*

$$\frac{\mathbf{u} - \mathbf{u}^n}{\tau} + \theta[-\nu\Delta\mathbf{u} + \mathbf{u} \cdot \nabla\mathbf{u}] + \nabla p = \mathbf{g}^{n+1}, \quad \nabla \cdot \mathbf{u} = 0 \quad in\ \Omega \tag{2.29}$$

[1] The FEATFLOW package and documentation are available at http:\\www.featflow.com.

for $\mathbf{u} = \mathbf{u}^{n+1}$ *and* $p = p^{n+1}$, *where the right-hand side is given by*

$$\mathbf{g}^{n+1} = \theta \mathbf{f}^{n+1} + (1-\theta)\mathbf{f}^n - (1-\theta)[-\nu\Delta\mathbf{u}^n + \mathbf{u}^n \cdot \nabla\mathbf{u}^n]. \tag{2.30}$$

The parameter $\theta \in [0,1]$ defines a group of so-called *one-step-θ-schemes.* Due to the inherent stability problems of explicit time-stepping schemes ($\theta = 0$) one usually opts for an implicit scheme, such as the Backward-Euler (BE) ($\theta = 1$), or Crank-Nicolson (CN) ($\theta = 1/2$), both of which have well-known advantages and disadvantages. The Crank-Nicolson scheme, for instance, is second-order accurate, allowing for large time steps when the solution is smooth, but is not *strongly* A-stable, making it susceptible to oscillations when the solution is non-smooth. The Backward Euler, on the other hand, is strongly A-stable, but only first-order accurate (cf. [36, 96, 116, 117]).

The temporal discretization in FEATFLOW is accomplished using a variation of the fractional-step-θ-scheme first proposed by Glowinski et al. [52] and Bristeau et al. [21].

Given the current solution $\mathbf{u}^n := \mathbf{u}(t_n)$ for velocity at time t_n and θ chosen appropriately, each (macro) time step $\tau = t_{n+1} - t_n$ is split into three consecutive subintervals $[t_n, t_{n+\theta})$, $[t_{n+\theta}, t_{n+(1-\theta)})$ and $[t_{n+(1-\theta)}, t_{n+1}]$ of length $\theta\tau$, $(1-2\theta)\tau$ and $\theta\tau$, respectively. The solution $\mathbf{u}^{n+1}$ at time t_{n+1} is obtained via intermediate solutions $\mathbf{u}^{n+\theta}$ and $\mathbf{u}^{n+(1-\theta)}$ at times $t_{n+\theta}$ and $t_{n+(1-\theta)}$, respectively.

Specifically, setting $\theta = 1 - \frac{\sqrt{2}}{2}$, $\theta' = 1 - 2\theta$, $\alpha = \frac{1-2\theta}{1-\theta}$ and $\beta = 1 - \alpha$, the following three nonlinear saddle point problems are discretized in space and solved for each time level (with $\tilde{\theta} = \alpha\theta\tau = \beta\theta'\tau$):

$$\begin{aligned} [\mathrm{I} + \tilde{\theta}\mathcal{N}(\mathbf{u}^{n+\theta})]\mathbf{u}^{n+\theta} + \theta\tau\nabla p^{n+\theta} &= [\mathrm{I} - \beta\theta\tau\mathcal{N}(\mathbf{u}^n)]\mathbf{u}^n + \theta\tau\mathbf{f}^n \\ \nabla\cdot\mathbf{u}^{n+\theta} &= 0, \end{aligned} \tag{2.31}$$

$$\begin{aligned} [\mathrm{I} + \tilde{\theta}\mathcal{N}(\mathbf{u}^{n+(1-\theta)})]\mathbf{u}^{n+(1-\theta)} &+ \theta'\tau\nabla p^{n+(1-\theta)} \\ &= [\mathrm{I} - \alpha\theta'\tau\mathcal{N}(\mathbf{u}^{n+\theta})]\mathbf{u}^{n+\theta} + \theta'\tau\mathbf{f}^{n+(1-\theta)} \\ \nabla\cdot\mathbf{u}^{n+(1-\theta)} &= 0, \end{aligned} \tag{2.32}$$

$$\begin{aligned} [\mathrm{I} + \tilde{\theta}\mathcal{N}(\mathbf{u}^{n+1})]\mathbf{u}^{n+1} &+ \theta\tau\nabla p^{n+1} \\ &= [\mathrm{I} - \beta\theta\tau\mathcal{N}(\mathbf{u}^{n+(1-\theta)})]\mathbf{u}^{n+(1-\theta)} + \theta\tau\mathbf{f}^{n+(1-\theta)} \\ \nabla\cdot\mathbf{u}^{n+1} &= 0, \end{aligned} \tag{2.33}$$

where for convenience we have combined the diffusive and convective terms by setting $\mathcal{N}(\mathbf{u})\mathbf{u} = -\nu\Delta\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u}$. Note that the body forces are treated fully explicitly in (2.31) and (2.33) but implicitly in (2.32), while the pressure is treated fully implicitly throughout.

This time discretization scheme combines the advantages of some traditional implicit schemes; rigorous analysis of the fractional-step-θ-scheme for the given choices of the parameters θ, α and β have shown it to be strongly A-stable, as is the Backward-Euler scheme, and second-order accurate, as is the Crank-Nicolson scheme (see Müller-Urbaniak [108], Klouček/Rys [90], Müller, et al. [107], Rannacher [117]).

In order to simplify the description of the spatial discretization in the following section, we note now that each of the problems (2.31)-(2.33) can be written in the form of a generalized stationary Navier-Stokes problem

$$\alpha \mathbf{u} - \nu \Delta \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p = \mathbf{f}, \qquad \nabla \cdot \mathbf{u} = 0. \tag{2.34}$$

2.3.2 Spatial Discretization

For the spatial discretization of (2.34) the finite element method is used. To this end, let $\mathcal{T}^h$ be a regular decomposition of the domain Ω into convex quadrilaterals $T \in \mathcal{T}^h$, with h_T denoting the diameter of element T and h the maximum of all h_T for $T \in \mathcal{T}^h$ as described in Section 1.3.5. Based on this decomposition, finite-dimensional subspaces $V_h \subset H_0^1(\Omega)^2$ and $Q_h \subset L_0^2(\Omega)$ are utilized to form the following discrete variational formulation of the stationary problem (2.34) (with $\alpha = 1$ and homogeneous boundary conditions for notational simplicity):

Find $\mathbf{u}_h \in V_h$ and $p_h \in Q_h$, such that

$$(\mathbf{u}_h, \mathbf{v}_h) + \nu a(\mathbf{u}_h, \mathbf{v}_h) + n(\mathbf{u}_h, \mathbf{u}_h, \mathbf{v}_h) + b(\mathbf{v}_h, p_h) = (\mathbf{f}, \mathbf{v}_h) \quad \forall\, \mathbf{v}_h \in V_h \tag{2.35}$$

$$b(\mathbf{u}_h, q_h) = 0 \qquad \forall\, q_h \in Q_h \tag{2.36}$$

with the terms $a(\cdot,\cdot)$, $b(\cdot,\cdot)$ and $n(\cdot,\cdot,\cdot)$ as defined in (2.8)-(2.10).

To ensure that (2.35)-(2.36) is a stable approximation to (2.34) as $h \to 0$ the spaces V_h and Q_h must be chosen to fulfil the *inf-sup* or *Babuška-Brezzi stability condition*

$$\inf_{q_h \in Q_h} \left\{ \sup_{\mathbf{v}_h \in V_h} \frac{b(\mathbf{v}_h, q_h)}{\|\nabla \mathbf{v}_h\|_0 \, \|q_h\|_0} \right\} \geq \gamma > 0, \tag{2.37}$$

with a constant γ that is independent of h. This ensures that solutions of (2.35)-(2.36) are uniquely determined in $V_h \times Q_h$ and stable (cf. Girault/Raviart [51, Theorem I.4.1] and Cuvelier et al. [36, Section 7.2]).

The finite element used by FEATFLOW is the 2D variant of the so-called $\tilde{Q}_1/Q_0$-element, a quadrilateral adaptation of the triangular Stokes element of Crouzeix-Raviart (see Cuvelier et al. [36] or Girault/Raviart [50]) that uses piecewise rotated bilinear shape functions (spanned by $< x_1^2 - x_2^2, x_1, x_2, 1 >$) for the velocities and piecewise constant functions for the pressure approximation. The nodal points for the velocity are located at the midpoints of the cell edges with the nodal points for the pressure located at the center of the cell.

The parametric version of the $\tilde{Q}_1/Q_0$-element is constructed using a reference element $\hat{T} = [0,1]^2$ in coordinates $(\hat{x}_1, \hat{x}_2)$, with the nodal points at the coordinates $\hat{m}_1 = (0,-1)$,

$\hat{m}_2 = (1,0)$, $\hat{m}_3 = (0,1)$ and $\hat{m}_4 = (-1,0)$ as shown in Figure 2.2. The rotated bilinear finite element functions are constructed on the reference element by finding functions $\hat{\phi}_i \in \text{span}\{\hat{x}_1^2 - \hat{x}_2^2, \hat{x}_1, \hat{x}_2, 1\}$, $i = 1, \dots, 4$ such that $\hat{\phi}_i(\hat{m}_i) = \delta_{ij}$ for $i, j = 1, \dots, 4$. Easy calculations then give

$$\hat{\phi}_i(\hat{x}_1, \hat{x}_2) = \begin{cases} \frac{1}{4} - \frac{1}{2}\hat{x}_2 - \frac{1}{4}(\hat{x}_1^2 - \hat{x}_2^2) & \text{for } i = 1 \\ \frac{1}{4} + \frac{1}{2}\hat{x}_1 + \frac{1}{4}(\hat{x}_1^2 - \hat{x}_2^2) & \text{for } i = 2 \\ \frac{1}{4} + \frac{1}{2}\hat{x}_2 - \frac{1}{4}(\hat{x}_1^2 - \hat{x}_2^2) & \text{for } i = 3 \\ \frac{1}{4} - \frac{1}{2}\hat{x}_1 + \frac{1}{4}(\hat{x}_1^2 - \hat{x}_2^2) & \text{for } i = 4 \end{cases}$$

The so-constructed basis functions are then mapped from $\hat{T}$ onto a general quadrilateral T by the transformation $\phi : \hat{T} \to T$, defined by

$$\psi(\hat{x}_1, \hat{x}_2) = \frac{1}{2}[(q_2 - q_1)\hat{x}_1 + (q_4 - q_1)\hat{x}_2 + (q_4 + q_2)],$$

where $q_1, \dots, q_4$ are the vertices of T as illustrated in Figure 2.2.

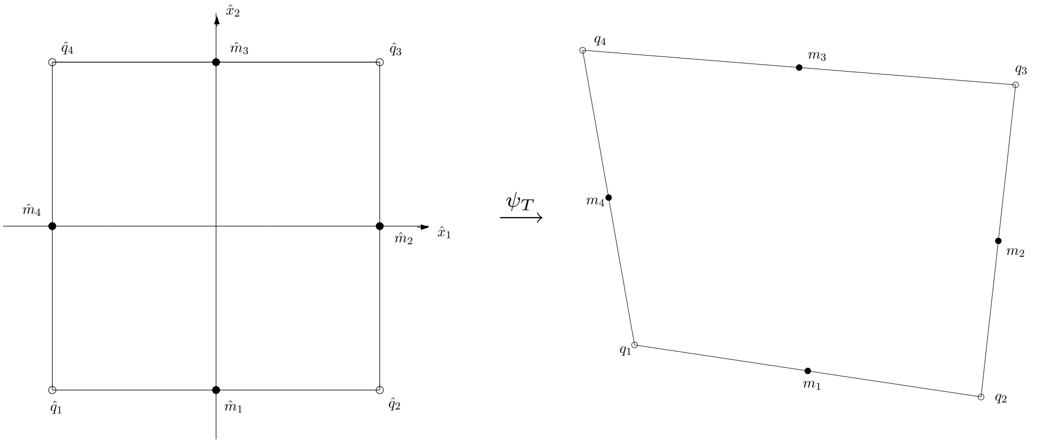


Figure 2.2: The transformation $\psi_T : \hat{T} \to T$ from the reference element $\hat{T}$ (left) to the general quadrilateral T (right).

The resulting element pair is nonconforming, since the shape functions are not continuous along the cell edges. We note that the spaces spanned by these nonconforming elements are not truly a subset of $H_0^1(\Omega) \times L_0^2(\Omega)$; nevertheless, we will continue to denote these spaces as above by V_h and Q_h. The $\tilde{Q}_1/Q_0$-element is known to satisfy the discrete inf-sup condition (2.37) on fairly general meshes [118]. More extensive analysis of the $\tilde{Q}_1/Q_0$-element can be found in Rannacher/Turek [118] and Turek [139], [140].

Selection of the desired nodal bases $\{\mathbf{v}_h^1, \dots, \mathbf{v}_h^N\} \subset V_h$ for the velocity space and $\{q_h^1, \dots, q_h^M\} \subset Q_h$ for the pressure space leads in the usual manner to the following large-scale system of nonlinear equations, which must be solved at each time step:

$$\mathrm{M}\mathbf{u} + \nu \mathrm{A}\mathbf{u} + \mathrm{N}(\mathbf{u})\mathbf{u} + \mathrm{B}p = \mathbf{f} \tag{2.38}$$

$$\mathrm{B}^T\mathbf{u} = 0, \tag{2.39}$$

where $\mathbf{u} \in \mathbb{R}^{2N}$, $p \in \mathbb{R}^M$ and $\mathbf{f} \in R^{2N}$ are now understood to be vectors resulting from the spatial discretization, M is the mass matrix, A is the stiffness matrix, B is the gradient matrix, $-\mathrm{B}^T$ is the divergence matrix, $\mathrm{N}(\cdot)$ is the nonlinear transport matrix and $\mathbf{f}$ is the discretized load vector.

2.3.3 Solution of the Nonlinear Systems

Summarizing the preceding sections, the discretization in space and time leads to a discrete coupled system of nonlinear equations for each time substep; that is, given $\mathbf{u}^n$ at time t_n, a time step τ, and constants $\vartheta_1, \ldots, \vartheta_4$ that depend on the substep, solve for $\mathbf{u} = \mathbf{u}(t_n + \tau)$ and $p = p(t_n + \tau)$ the system

$$\mathrm{S}(\mathbf{u}) + \tau \mathrm{B} p = \mathbf{g} \tag{2.40}$$

$$\mathrm{B}^T \mathbf{u} = 0, \tag{2.41}$$

where we have set

$$\mathrm{S}(\mathbf{u}) = [\mathrm{M} + \vartheta_1 \tau (\nu \mathrm{A} + \mathrm{N}(\mathbf{u}))]\mathbf{u}, \tag{2.42}$$

$$\mathbf{g} = [\mathrm{M} - \vartheta_2 \tau (\nu \mathrm{A} + \mathrm{N}(\mathbf{u}^n))]\mathbf{u}^n + \vartheta_3 \tau \mathbf{f} + \vartheta_4 \tau \mathbf{f}^n, \tag{2.43}$$

with $\mathbf{f}^n = \mathbf{f}(t_n)$ and $\mathbf{f} = \mathbf{f}(t_n + \tau)$.

The details of the procedure used to solve (2.40)-(2.41) are not absolutely essential for the remainder of this thesis, but we will give a brief summary in the interest of completeness.

For large Reynolds numbers and small time steps τ, we have the following statement from [140] pertaining to the operator S and the matrix M:

$$\mathrm{S} = \mathrm{M} + \mathcal{O}(\tau).$$

Thus, S can be interpreted as a nonsymmetric and nonlinear, but well-conditioned perturbation of the mass matrix M for small τ. This fact builds the essential basis for the following discrete projection scheme used in FEATFLOW:

Given the pressure p^n at time t_n and a time step τ:

Step 1. Solve the nonlinear transport diffusion equation

$$\mathrm{S}(\tilde{\mathbf{u}}) = \mathbf{g} - \tau B p^n \tag{2.44}$$

to obtain the intermediate velocity $\tilde{\mathbf{u}}$.

Step 2. Using the divergence of $\tilde{\mathbf{u}}$ as the right-hand, solve for q the discrete Poisson problem

$$\mathrm{B}^T \mathrm{M}_l^{-1} \mathrm{B} q = \frac{1}{\tau} \mathrm{B}^T \tilde{\mathbf{u}},$$

where the lumped mass matrix M_l *serves as a preconditioner.*

Step 3. Update $p = p^n + q$ *and calculate*

$$\mathbf{u} = \tilde{\mathbf{u}} - \tau \mathrm{M}_l^{-1} \mathrm{B} q.$$

As described in [140], the above method is an operator splitting method similar to those proposed by Chorin [31] or Van Kan [144] (see also Prohl [115]), that decouples the computation of $\mathbf{u}$ and p. Step 3 essentially projects the intermediate velocity field $\tilde{\mathbf{u}}$, which is not necessarily divergence-free, onto a divergence-free subspace such that $\mathbf{u}$ satisfies the discrete incompressibility constraint.

The solution of the nonlinear system (2.44) is achieved using an adaptive fixed point defect correction method that is described in detail in [142]. The resulting linear systems are solved using multigrid methods.

Chapter 3

The Streamline Diffusion Method

This chapter is dedicated to stabilization methods required for the numerical solution of convection-diffusion problems with dominant convection. Section 3.1 begins with a simple example illustrating how instabilities of purely numerical character can arise for such problems. The example is then supplemented by a more detailed examination of a situation involving the discretization of the linearized Navier-Stokes equations. In Sections 3.2-3.3 we discuss some methods for stabilizing the numerical solution procedure, choosing the so-called streamline diffusion technique for more detailed analysis. Because of its conceptual simplicity relative to the case for the Navier-Stokes equations, we use the linear convection-diffusion problem to introduce the streamline diffusion method in Section 3.2, then extend the discussion to the Navier-Stokes equations in Section 3.3.

3.1 The Necessity of Stabilization

We begin by examining instabilities in a simple one-dimensional transport-diffusion problem before moving on to an analysis of instabilities in more sophisticated problems.

3.1.1 A Simple Example

At medium and high Reynolds numbers, it is well-known that standard Galerkin finite element solutions for mixed transport-diffusion equations may suffer from numerical instabilities caused by dominance of convective terms if the exact solution is not smooth enough (cf. Fries/Matthies [46]).

Consider for example (cf. Brooks/Hughes [22]) the one-dimensional convection-diffusion problem

$$\begin{aligned} -\nu u'' + u' = 0 \quad & \text{in } \Omega = (0,1) \\ u(0) = 0, \quad & u(1) = 1, \end{aligned} \tag{3.1}$$

with $\nu > 0$, which has the exact solution $u(x) = (1 - e^{Pe})^{-1}(1 - e^{xPe})$, where $Pe = 1/\nu$ is the *global Peclét number*. The problem (3.1) is an example of a singularly perturbed problem with a boundary layer at $x = 1$ (cf. Roos et al. [122] for more examples and discussion of singularly perturbed problems). For $\nu \to 0$ the exact solution approaches a function that is discontinuous at $x = 1$.

To illustrate the numerical difficulties caused by the boundary layer for small ν, we apply the standard Galerkin finite element method with piecewise linear basis functions on a uniform mesh with mesh constant h to (3.1), resulting in a system of equations

$$\begin{gathered} -\frac{\nu}{h^2}\left(u_{i+1} - 2u_i + u_{i-1}\right) + \frac{1}{2h}(u_{i+1} + u_{i-1}) = 0, \qquad i = 1, \ldots, N-1, \\ u_0 = 0, \quad u_N = 1, \end{gathered} \tag{3.2}$$

which may be solved for the values u_i of the finite element approximation u_h at the grid points $x_i = ih$, $i = 0, 1, \ldots, N$, with $x_0 = 0$ and $x_N = 1$. Note that this system may also be interpreted as a finite difference scheme with central differences used for the convective term u'.

Similar to an argument in Cuvelier et al. [36] (see also [106]), one can show using difference equations that the solution to the system (3.2) is given by

$$u_i = (1 - (\xi/\eta)^i)(1 - (\xi/\eta)^N)^{-1}, \tag{3.3}$$

with $\xi = -\nu - h/2$ and $\eta = -\nu + h/2$. Since ξ is always negative, the sign of (3.3) will depend on the sign of η. For $\eta < 0$ the solution u_i will clearly be positive for all $\xi \neq \eta$. For $\eta > 0$ we have $\xi/\eta < -1$, and the solution will display oscillatory behavior. To avoid these oscillations, the grid spacing h must be chosen such that

$$\eta = -\nu + h/2 < 0 \quad \Leftrightarrow \quad \frac{h}{2\nu} < 1, \tag{3.4}$$

where $h/2\nu$ is the *local Peclét number.* At higher Peclét numbers (small ν), the resulting grids lead to algebraic systems of equations that are computationally intractable, especially for problems in two or three dimensions. In this sense, classical discretization methods fail.

The situation is illustrated in Figure 3.1, where we have plotted the exact solution of (3.1) with $\nu = 1/100$ against two solutions of the numerical approximation (3.2). Choosing $h = 1/10 >> \nu$ leads to large oscillations even far from the boundary layer at $x = 1$, while setting $h = \nu$ — which fulfils condition (3.4) — results in an accurate numerical approximation.

The condition (3.4) on h is too severe to be practicable at higher Reynolds numbers. One simple method for alleviating this difficulty is to avoid the situation completely by adding artificial diffusion to the problem formulation, that is, by replacing the term $\nu u''$ in (3.1) by hu''. The resulting Galerkin formulation fulfils condition (3.4) making it stable, and as $h > \nu$ becomes smaller, the modified problem approaches the original problem. The drawback of this method is twofold: It introduces a diffusion term acting in the direction perpendicular to the streamline direction (crosswind diffusion), so that a sharp jump across a streamline will be smeared out, and the added term $(h - \nu)u''$ makes the method at best first order accurate, even for smooth solutions (cf. [122]).

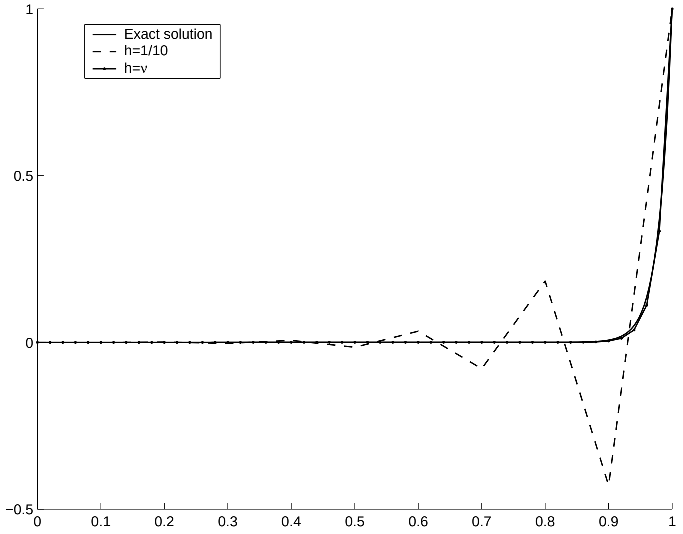


Figure 3.1: The exact solution of (3.1) compared with the numerical solution (3.3) for $h = 1/10$ and $h = \nu$.

3.1.2 Sources of Instabilities in Standard Finite Element Methods

Before introducing more effective stabilization methods for convection-diffusion equations, we take a deeper look at the stability and convergence of standard finite element methods. Following Tobiska [135], we consider the Stokes equation with a convection term

$$\begin{aligned} -\nu\Delta\mathbf{u} + \mathbf{b}\cdot\nabla\mathbf{u} + \nabla p &= \mathbf{f} \text{ in } \Omega \\ \nabla\cdot\mathbf{u} &= 0 \text{ in } \Omega \\ \mathbf{u} &= 0 \text{ on } \Gamma \end{aligned} \tag{3.5}$$

in a bounded polyhedral domain $\Omega \subset \mathbb{R}^2$ where $\mathbf{f} \in L^2(\Omega)^2$ and $\mathbf{b} \in H^1(\Omega)^2 \cap L^\infty(\Omega)^2$ with $\nabla\cdot\mathbf{b} = 0$, and the assumptions, definitions and notation of Section 1.3.5 hold. Using the spaces $V := H_0^1(\Omega)^2$ and $Q := L_0^2(\Omega)$ and setting $X := V \times Q$, the weak formulation of problem (3.5) can be written:

Find $\mathbf{u} \in V$ and $p \in Q$ such that

$$\begin{aligned} \nu(\nabla\mathbf{u},\nabla\mathbf{v}) + (\mathbf{b}\cdot\nabla\mathbf{u},\mathbf{v}) - (p,\nabla\cdot\mathbf{v}) &= (\mathbf{f},\mathbf{v}) \quad \forall\,\mathbf{v}\in V, \\ (q,\nabla\cdot\mathbf{u}) &= 0 \quad \forall\, q \in Q. \end{aligned} \tag{3.6}$$

Or equivalently:

Find a pair $[\mathbf{u},p] \in X$ such that

$$a([\mathbf{u},p],[\mathbf{v},q]) = (\mathbf{f},\mathbf{v}) \qquad \forall\,[\mathbf{v},q]\in X, \tag{3.7}$$

where the bilinear form $a(\cdot,\cdot) : X \mapsto \mathbb{R}$ is defined by

$$a([\mathbf{u},p],[\mathbf{v},q]) = \nu(\nabla\mathbf{u},\nabla\mathbf{v}) + (\mathbf{b}\cdot\nabla\mathbf{u},\mathbf{v}) - (p,\nabla\cdot\mathbf{v}) + (q,\nabla\cdot\mathbf{u}).$$

The pressure can be eliminated from (3.6) by introducing

$$W = \{\mathbf{v} \in V \mid (q, \nabla \cdot \mathbf{v}) = 0 \quad \forall\, q \in L_0^2(\Omega)\},$$

a subspace of the divergence-free functions. The space W allows us to split the problem (3.5) into subproblems of the form:

Find $\mathbf{u} \in W$ *such that*

$$\nu(\nabla \mathbf{u}, \nabla \mathbf{v}) + (\mathbf{b} \cdot \nabla \mathbf{u}, \mathbf{v}) = (\mathbf{f}, \mathbf{v}) \quad \forall\, \mathbf{v} \in W \tag{3.8}$$

for the velocity, and (with a known velocity):

Find $p \in Q$ *such that*

$$(p, \nabla \cdot \mathbf{v}) = \nu(\nabla \mathbf{u}, \nabla \mathbf{v}) + (\mathbf{b} \cdot \nabla \mathbf{u}, \mathbf{v}) - (\mathbf{f}, \mathbf{v}) \quad \forall\, \mathbf{v} \in W^{\perp} \tag{3.9}$$

for the pressure.

Indeed, if $[\mathbf{u}, p]$ is a solution pair of (3.7) then $\mathbf{u}$ clearly solves (3.8). Conversely, the Lax-Milgram theorem and the positivity of the bilinear form

$$\nu(\nabla \mathbf{v}, \nabla \mathbf{v}) + (\mathbf{b} \cdot \nabla \mathbf{v}, \mathbf{v}) \geq \nu\, |\mathbf{v}|_1^2 \quad \forall\, \mathbf{v} \in V$$

guarantee the existence of a unique solution $\mathbf{u}$ to (3.8). The unique solvability of (3.9), and therewith (3.6), is now a consequence of the *Babuška-Brezzi condition*

$$\inf_{q \in Q} \sup_{\mathbf{v} \in V} \frac{(q, \nabla \cdot \mathbf{v})}{\|q\|_0\, |\mathbf{v}|_1} \geq \gamma > 0, \tag{3.10}$$

which holds for the space X (cf. Girault/Raviart [51, Lemma I.4.1 and Theorem I.5.1]).

Let $V_h \subset V$ and $Q_h \subset Q$ be two families of finite element spaces corresponding to the family $\mathcal{T}^h$ of partitions of Ω as described in Section 1.3. The discrete conforming finite element method corresponding to (3.7) reads

Find a pair $[\mathbf{u}_h, p_h] \in X_h := V_h \times Q_h$ *such that*

$$a([\mathbf{u}_h, p_h], [\mathbf{v}_h, q_h]) = (\mathbf{f}, \mathbf{v}_h) \qquad \forall\, [\mathbf{v}_h, q_h] \in X_h. \tag{3.11}$$

If one assumes the discrete version of (3.10) with a mesh-independent parameter $\gamma > 0$, then (3.11) obviously admits a stable unique solution; however, not all sets of finite element spaces V_h and Q_h satisfy (3.10) with a constant γ that is independent of the mesh size h. The next result, which was proved in [135], gives some approximation estimates that demonstrate the dependence of the approximation error on the diffusion constant ν — as was demonstrated numerically for a simple problem in the earlier example — and the failure of condition (3.10) to hold uniformly in h.

Theorem 3.1. *Let the condition (3.10) be fulfilled with a constant γ_h that may depend on h. Then problem (3.11) has a unique solution $[\mathbf{u}_h, p_h]$, which satisfies the stability estimate*

$$\nu \left(|\mathbf{u}_h|_1 + \gamma_h \|p_h\|_0 \right) \leq c \|\mathbf{f}\|_{-1} \tag{3.12}$$

and the error estimates

$$\nu |\mathbf{u} - \mathbf{u}_h|_1 \leq \kappa_1 \inf_{\mathbf{v}_h \in V_h} |\mathbf{u} - \mathbf{v}_h|_1 + \kappa_2 \inf_{q_h \in Q_h} \|p - q_h\|_0 \tag{3.13}$$

$$\|p - p_h\|_0 \leq \kappa_3 \inf_{\mathbf{v}_h \in V_h} |\mathbf{u} - \mathbf{v}_h|_1 + \kappa_4 \inf_{q_h \in Q_h} \|p - q_h\|_0 \,, \tag{3.14}$$

and the constants $\kappa_1, \ldots, \kappa_4$ behave like

$$\begin{aligned} \kappa_1 &\sim c\big(1 + \frac{1}{\gamma_h}\big), & \kappa_2 &\sim c, \\ \kappa_3 &\sim c\frac{1}{\nu\gamma_h}\big(1 + \frac{1}{\gamma_h}\big), & \kappa_4 &\sim c\big(1 + \frac{1}{\gamma_h} + \frac{1}{\nu\gamma_h}\big) \end{aligned}$$

for $\nu \to 0$ and/or $\gamma_h \to 0$.

We see from the stability inequality (3.12) that stability breaks down for both velocity and pressure in the presence of dominate convection, but only the pressure is destabilized by the failure to bound γ_h away from 0. Nevertheless, in order to achieve a stable formulation for mixed finite element spaces, one must obviously deal with both difficulties.

There are a number of methods available for dealing with the difficulties outlined in Theorem 3.1. Several upwind methods leading to algebraic systems with M-matrices have been proposed and studied (e.g., Ohmori/Ushijima [113] or Roos et al. [122]); however, these also contain a large amount of artificial diffusion leading to a restricted order of convergence. Good stability properties and better theoretical convergence can be achieved using the streamline diffusion and Galerkin least squares methods. We will concentrate on the streamline diffusion method, which as its name implies, adds an artificial diffusion term acting only in the streamline direction. It turns out that this provides sufficient stabilization to reduce oscillations in the standard Galerkin method while avoiding the artificial crosswind diffusion that causes difficulties with the upwind methods. We refer to Lube [101] for a description of a related Galerkin least squares method.

3.2 The SDFEM for Linear Convection-Diffusion Problems

In the streamline diffusion finite element method (SDFEM), introduced by Hughes/Brooks [78, 22], an artificial diffusion operator is added to the convective term in a tensorial form so as to act only in the streamline direction. Since its introduction, streamline diffusion has been applied to a variety of stationary and time-dependent convection-diffusion problems. For conforming finite elements, theoretical and numerical investigations have demonstrated a near optimal $\mathcal{O}(h^{k+1/2})$ order of L^2-convergence for piecewise polynomials of degree k (cf. Nävert [110], Johnson et al. [82, 84], Roos et al. [122], Niijima [111], Tobiska/Verfürth [137] or Zhou [150]).

Assumptions

We will assume throughout the remainder of this section that $\Omega \subset R^2$ is a bounded, polyhedral domain with boundary Γ, and that $\mathcal{T}^h$ is a family of quasiuniform decompositions of Ω as described in Section 1.3. Moreover, let $V_h \subset V := H_0^1(\Omega)$ be a conforming finite element space consisting of piecewise polynomials of degree k, that is,

$$V_h = \{v_h \in V : v_h\big|_T \in P_k(T) \ \forall T \in \mathcal{T}^h\}. \tag{3.15}$$

If $u \in H^{k+1}(T)^2$ for $k \geq 1$, then it can be shown (cf. Ciarlet [32]; Clément [33]) that its interpolant $I_h u$ from V_h satisfies the approximation properties

$$|u - I_h u|_{m,T} \leq C h^{k+1-m} \, |u|_{k+1,T} \quad \text{for } m = 0, 1, 2 \tag{3.16}$$

on each $T \in \mathcal{T}^h$, and the inverse estimates

$$\|\Delta v_h\|_{0,T} \leq \mu h_T^{-1} \, |v_h|_{1,T} \quad \forall v_h \in V_h \tag{3.17}$$

and

$$\|v_h\|_\infty \leq \zeta h^{-\kappa} \, \|\nabla v_h\|_0 \quad \forall \, v_h \in V_h, \tag{3.18}$$

where $\kappa > 0$ and the constants μ and ζ are independent of T and h.

3.2.1 A Typical Application of Streamline Diffusion

For orientation, we summarize a streamline diffusion method presented in Roos et al. [122] for a stationary linear convection-diffusion problem. The convergence analysis and proofs can be found in the cited references; nevertheless, we include them here for the following exemplary problem to facilitate the general discussion of variants and extensions that follows. Consider the problem

$$Lu := -\nu\Delta u + \mathbf{b} \cdot \nabla u + cu = f \qquad \text{in } \Omega \tag{3.19}$$

$$u = 0 \qquad \text{on } \Gamma, \tag{3.20}$$

where $\nu > 0$, and $\mathbf{b}$, c, and f are sufficiently smooth and satisfy the assumption

$$\inf_{x \in \Omega} \left(c(x) - \frac{1}{2} \nabla \cdot \mathbf{b}(x) \right) \geq c_0 > 0. \tag{3.21}$$

We will call a function $u \in V$ that satisfies

$$a^{\mathrm{G}}(u, v) := \nu(\nabla u, \nabla v) + (\mathbf{b} \cdot \nabla u + cu, v) = (f, v) \quad \forall v \in V \tag{3.22}$$

a *weak solution* of (3.19)-(3.20). Note that under the assumption (3.21) the existence of a unique solution of the problem (3.22) follows easily from the Lax-Milgram theorem.
The standard discrete Galerkin finite element formulation for (3.22) reads:

Find $u_h \in V_h$ such that

$$a^G(u_h, v_h) = (f, v_h) \quad \forall v_h \in V_h. \tag{3.23}$$

As we saw in Section 3.1, this formulation may suffer from numerical instabilities in the presence of dominant convective terms. The SDFEM operates by adding weighted residuals to the usual Galerkin finite element methodas follows: Under the assumptions on the problem data, a solution $u \in V$ of (3.22) obviously satisfies $Lu = f$ in $L^2(\Omega)$, and we can conclude that if $\psi(v) \in L^2(\Omega)\ \forall\, v \in V$ for some transformation ψ, then

$$a^{\mathrm{G}}(u, v) + \sum_{T \in \mathcal{T}^h} (Lu - f, \psi(v))_T = (f, v) \quad \forall v \in V, \tag{3.24}$$

where $(\cdot, \cdot)_T$ denotes the inner product in $L^2(T)$. Note that since in general $\Delta u_h \notin L^2(\Omega)$, but $\Delta u_h \in L^2(T)$ for each T, we must calculate the term Δu in Lu element by element. Using (3.24), we formulate the following *streamline diffusion finite element method*:

Find $u_h \in V_h$ such that

$$a_h(u_h, v_h) = l_h(v_h) \quad \forall v_h \in V_h, \tag{3.25}$$

where

$$\begin{aligned} a_h(u, v) &:= \nu(\nabla u, \nabla v) + (\mathbf{b} \cdot \nabla u + cu, v) \\ &\qquad + \sum_{T \in \mathcal{T}^h} (-\nu \Delta u + \mathbf{b} \cdot \nabla u + cu, \psi(v))_T, \end{aligned} \tag{3.26}$$

$$l_h(u, v) := (f, v) + \sum_{T \in \mathcal{T}^h} (f, \psi(v))_T \tag{3.27}$$

and

$$\psi(v) \mid_T := \delta_T \mathbf{b} \cdot \nabla v \quad \forall\, T \in \mathcal{T}^h. \tag{3.28}$$

Remark 3.2. (i) Note that because of (3.24) the formulation (3.25)-(3.28) is *consistent* for $u \in H^2(\Omega)$ in the sense that a solution of (3.25) is also a solution of (3.23), i.e., the *projection property*

$$a_h(u - u_h, v_h) = 0 \qquad \forall v_h \in V_h \tag{3.29}$$

holds for the SDFEM. This fact makes it possible to obtain convergence properties that are superior to the classical artificial diffusion methods, which add a perturbation to the solution.
(ii) The streamline diffusion method (3.25)-(3.28) can also be derived by multiplication of (3.22) with test functions of the form $v_h + \delta_T \mathbf{b} \cdot \nabla v_h$, that is, (3.25)-(3.28) can be

interpreted as a *Petrov-Galerkin method*, in which the test functions belong to a space which is different from the space of trial functions V_h where the discrete solution u_h is sought.
(iii) By setting

$$\psi(v_h) \mid_T := \delta_T(-\nu\Delta v_h + \mathbf{b}\cdot\nabla v_h + cv_h)_T \quad \forall\, T \in \mathcal{T}^h \tag{3.30}$$

in (3.28), we obtain a Galerkin least squares method (cf. [101]).

The correct choice of the *local damping parameter (SD-parameter)* δ_T in (3.28) is critical for optimal convergence of the streamline diffusion model. We will provide more discussion of this parameter later in this section; for now, in order to facilitate analysis of the stability and convergence properties of (3.25)-(3.28), we set

$$c_\infty = \sup_{x\in\Omega} |c(x)|$$

and assume

$$0 < \delta_T \le \frac{1}{2}\min\left(\frac{c_0}{c_\infty^2}, \frac{h_T^2}{\nu\mu^2}\right) \tag{3.31}$$

for each $T \in \mathcal{T}^h$.

We can now prove error estimates for (3.25)-(3.28) in the norm

$$|||v|||^2 := \nu\,|v|_1^2 + c_0\,\|v\|_0^2 + \sum_{T\in\mathcal{T}^h} \delta_T\,\|\mathbf{b}\cdot\nabla v\|_{0,T}^2 \,.$$

The choice of norm is motivated by the following stability property of the bilinear form (3.25).

Lemma 3.3. *Let V_h be the finite element space defined by (3.15) and assume the control parameter δ_T satisfies (3.31). Then the discrete bilinear form a_h is coercive on V_h, i.e.,*

$$a_h(v_h, v_h) \ge C_0\,|||v_h|||^2 \quad \forall v_h \in V_h, \quad \text{with } C_0 = \frac{1}{2}. \tag{3.32}$$

Proof. Setting $u_h = v_h$ in (3.26) and using partial integration and (3.21), we find that

$$\begin{aligned} a_h(v_h, v_h) &= \nu(\nabla v_h, \nabla v_h) + ((c - \frac{1}{2}\nabla\cdot\mathbf{b})v_h, v_h) \\ &\quad + \sum_{T\in\mathcal{T}^h} \delta_T(-\nu\Delta v_h + \mathbf{b}\cdot\nabla v_h + cv_h, \mathbf{b}\cdot\nabla v_h)_T \\ &\ge \nu\,|v_h|_1^2 + c_0\,\|v_h\|_0^2 + \sum_{T\in\mathcal{T}^h} \delta_T\,\|\mathbf{b}\cdot\nabla v_h\|_{0,T}^2 \\ &\quad + \sum_{T\in\mathcal{T}^h} \delta_T(-\nu\Delta v_h + cv_h, \mathbf{b}\cdot\nabla v_h)_T \\ &\ge |||v_h|||^2 - \left|\sum_{T\in\mathcal{T}^h} \delta_T(-\nu\Delta v_h + cv_h, \mathbf{b}\cdot\nabla v_h)_T\right|. \end{aligned}$$

The inverse estimate (3.17) and the conditions on δ_T imply

$$\nu\delta_T \left\|\Delta v_h\right\|_{0,T}^2 \leq \frac{1}{2}\left|v_h\right|_{1,T}^2 \quad \forall v_h \in \mathcal{T}^h,$$

so that using Young's inequality (1.7), the assumptions on δ_T and the inverse estimate (3.17), we have

$$\begin{aligned}
&\left|\sum_{T\in\mathcal{T}^h} \delta_T(-\nu\Delta v_h + cv_h, \mathbf{b}\cdot\nabla v_h)_T\right| \\
&\qquad \leq \Big(\sum_{T\in\mathcal{T}^h} \nu^2\delta_T \left\|\Delta v_h\right\|_{0,T}^2\Big)^{1/2}\Big(\sum_{T\in\mathcal{T}^h} \delta_T \left\|\mathbf{b}\cdot\nabla v_h\right\|_{0,T}^2\Big)^{1/2} \\
&\qquad\quad + \Big(\sum_{T\in\mathcal{T}^h} \delta_T \left\|cv_h\right\|_{0,T}^2\Big)^{1/2}\Big(\sum_{T\in\mathcal{T}^h} \delta_T \left\|\mathbf{b}\cdot\nabla v_h\right\|_{0,T}^2\Big)^{1/2} \\
&\qquad \leq \sum_{T\in\mathcal{T}^h} \nu^2\delta_T \left\|\Delta v_h\right\|_{0,T}^2 + \sum_{T\in\mathcal{T}^h} \delta_T c_\infty^2 \left\|v_h\right\|_{0,T}^2 + \frac{1}{2}\sum_{T\in\mathcal{T}^h} \delta_T \left\|\mathbf{b}\cdot\nabla v_h\right\|_{0,T}^2 \\
&\qquad \leq \frac{\nu}{2}\left|v_h\right|_1^2 + \frac{c_0}{2}\left\|v_h\right\|_0^2 + \frac{1}{2}\sum_{T\in\mathcal{T}^h} \delta_T \left\|\mathbf{b}\cdot\nabla v_h\right\|_{0,T}^2,
\end{aligned}$$

which delivers the desired result. □

We may now derive an error estimate that shows the convergence of the streamline diffusion method for the appropriate choice of the parameter δ_T. Using I_hu to denote the interpolant from V_h to the exact solution u, we first split the error into two parts using the triangle inequality

$$|||u - u_h||| \leq |||u - I_hu||| + |||I_hu - u_h|||. \tag{3.33}$$

Now, for the error between u_h and the interpolant I_hu, we have:

Lemma 3.4. *Let $u_h \in V_h$ be a solution of (3.25)-(3.28). Then we have the error estimate:*

$$\begin{aligned}
|||I_hu - u_h|||^2 &= \nu\left|I_hu - u_h\right|_1^2 + c_0\left\|I_hu - u_h\right\|_0^2 + \sum_{T\in\mathcal{T}^h} \delta_T\left\|\mathbf{b}\cdot\nabla(I_hu - u_h)\right\|_{0,T}^2 \\
&\leq Ch^k\Big(\sum_{T\in\mathcal{T}^h} \lambda_T\left|u\right|_{k+1,T}^2\Big),
\end{aligned} \tag{3.34}$$

if $u \in V \cap H^{k+1}(\Omega)$, $k \geq 1$. Furthermore, the constant C is independent of ν, and the parameter $\lambda_T = \lambda_T(\nu, h_T, \delta_T)$ is given by

$$\lambda_T = \nu + \delta_T + \delta_T^{-1}h_T^2 + h_T^2. \tag{3.35}$$

Proof. Setting $\vartheta_h := I_hu - u_h$, and using the coercivity (3.32) and the projection property of the bilinear form, we have

$$\begin{aligned}
\frac{1}{2}|||I_hu - u_h|||^2 &\leq a_h(I_hu - u, \vartheta_h) + a_h(u - u_h, \vartheta_h) \\
&= a_h(I_hu - u, \vartheta_h).
\end{aligned} \tag{3.36}$$

We now set $\eta_h := I_h u - u$ and estimate $a_h(\eta_h, \vartheta_h)$ term by term, using the interpolation properties (3.16) for $u \in V \cap H^{k+1}(\Omega)$. For the first term, we have

$$\begin{aligned} |\nu(\nabla\eta_h, \nabla\vartheta_h)| &\le \nu^{1/2} |\eta_h|_1 \, \nu^{1/2} |\vartheta_h|_1 \\ &\le C\nu^{1/2} h^k |u|_{k+1} \, |||\vartheta_h||| \, . \end{aligned} \tag{3.37}$$

Using integration by parts, the second term can be split into two terms

$$(\mathbf{b} \cdot \nabla\eta_h + c\eta_h, \vartheta_h) = ((c - \nabla \cdot \mathbf{b})\eta_h, \vartheta_h) - (\mathbf{b} \cdot \nabla\vartheta_h, \eta_h), \tag{3.38}$$

which are estimated as follows:

$$\begin{aligned} |((c - \nabla \cdot \mathbf{b})\eta_h, \vartheta_h)| &\le \sum_{T\in\mathcal{T}^h} |((c - \nabla \cdot \mathbf{b})\eta_h, \vartheta_h)_T| \\ &\le C \sum_{T\in\mathcal{T}^h} \|\eta_h\|_{0,T} \, |||\vartheta_h||| \\ &\le C\Big(\sum_{T\in\mathcal{T}^h} \|\eta_h\|_{0,T}^2 \Big)^{1/2} |||\vartheta_h||| \\ &\le Ch^k \Big(\sum_{T\in\mathcal{T}^h} h_T^2 \, |u|_{k+1,T}^2 \Big)^{1/2} |||\vartheta_h||| \, , \end{aligned}$$

and similarly,

$$\begin{aligned} |(\mathbf{b} \cdot \nabla\vartheta_h, \eta_h)| &\le \sum_{T\in\mathcal{T}^h} \delta_T^{1/2} \|\mathbf{b} \cdot \nabla\vartheta_h\|_{0,T} \, \delta_T^{-1/2} \|\eta_h\|_{0,T} \\ &\le Ch^k \Big(\sum_{T\in\mathcal{T}^h} \delta_T^{-1} h_T^2 \, |u|_{k+1,T}^2 \Big)^{1/2} |||\vartheta_h||| \, . \end{aligned}$$

The streamline diffusion term is estimated by

$$\begin{aligned} &\Big| \sum_{T\in\mathcal{T}^h} (-\nu\Delta\eta_h + \mathbf{b} \cdot \nabla\eta_h + c\eta_h, \delta_T \mathbf{b} \cdot \nabla\vartheta_h)_T \Big| \\ &\quad \le C \sum_{T\in\mathcal{T}^h} \delta_T^{1/2} (\nu \|\Delta\eta_h\|_{0,T} + \|\nabla\eta_h\|_{0,T} + \|\eta_h\|_{0,T}) \delta_T^{1/2} \|\mathbf{b} \cdot \nabla\vartheta_h\|_{0,T} \\ &\quad \le C \sum_{T\in\mathcal{T}^h} \delta_T^{1/2} (\nu h_T^{k-1} + h_T^k + h_T^{k+1}) \, |u|_{k+1,T} \, \delta_T^{1/2} \|\mathbf{b} \cdot \nabla\vartheta_h\|_{0,T} \\ &\quad \le C\Big(\sum_{T\in\mathcal{T}^h} (\nu^2 \delta_T h_T^{2k-2} + \delta_T h_T^{2k} + \delta_T h_T^{2k+2}) \, |u|_{k+1,T}^2 \Big)^{1/2} |||\vartheta_h||| \\ &\quad \le C\Big(\sum_{T\in\mathcal{T}^h} (\nu + \delta_T) h_T^{2k} \, |u|_{k+1,T}^2 \Big)^{1/2} |||\vartheta_h||| \, , \end{aligned}$$

where we have used $\nu^2\delta_T \le C\nu h_T^2$, which follows from (3.31). Combining the above estimates and dividing both sides of (3.36) by $|||\vartheta_h|||$ results in

$$|||\vartheta_h||| \le Ch^k \Big(\sum_{T\in\mathcal{T}^h} (\nu + \delta_T + \delta_T^{-1} h_T^2 + h_T^2) \, |u|_{k+1,T}^2 \Big)^{1/2} , \tag{3.39}$$

which proves the assertion. □

To obtain the best possible rate of convergence from (3.39), we must somehow balance the terms ν, δ_T and δ_T^{-1}. For the choice

$$\delta_T \sim \begin{cases} h_T & \text{for } \nu < h_T \quad \text{(dominant convection)} \\ h_T^2/\nu & \text{for } \nu \geq h_T \quad \text{(dominant diffusion)} \end{cases} \tag{3.40}$$

we get the following global error estimate for the streamline diffusion finite element method.

Theorem 3.5. *Under the assumptions of Lemma 3.3 and with δ_T chosen according to (3.40), the solution u_h of the SDFEM satisfies*

$$|||u - u_h||| \leq C(\nu^{1/2} + h^{1/2})h^k \, |u|_{k+1} \, . \tag{3.41}$$

Proof. It follows from (3.39) and (3.40) that

$$|||I_h u - u_h||| \leq C(\nu^{1/2} + h^{1/2})h^k \, |u|_{k+1} \, . \tag{3.42}$$

Moreover, using the interpolation estimates (3.16) for $\eta_h = u - I_h u$, we have

$$\begin{aligned} |||\eta_h||| &= \big(\nu \, |\eta_h|_1^2 + c_0 \, \|\eta_h\|_0^2 + \sum_{T \in \mathcal{T}^h} \delta_T \, \|\mathbf{b} \cdot \nabla \eta_h\|_{0,T}^2\big)^{1/2} \\ &\leq Ch^k \bigg(\sum_{T \in \mathcal{T}^h} (\nu + h_T^2 + \delta_T) \, |u|_{k+1}^2 \bigg), \end{aligned}$$

which combined with $\nu + h_T^2 + \delta_T \leq \lambda_T$, (3.42) and (3.33) results in (3.41). □

Remark 3.6. (i) For the more interesting convection-dominated case, we have $\nu < h_T$ and $\delta_T \sim h_T$, so that

$$|||u - u_h||| \leq Ch^{k+1/2} \, |u|_{k+1} \, . \tag{3.43}$$

Looking at the interpolation errors

$$\|u - I_h u\|_0 \leq Ch^{k+1} \, |u|_{k+1} \qquad \text{and} \qquad |u - I_h u|_1 \leq Ch^k \, |u|_{k+1} \, , \tag{3.44}$$

we see that the L^2-error is half a power of h from being optimal, while the L^2-error of the derivative in the streamline direction is in fact optimal.

(ii) A more exact analysis of the approximation error (3.33) yields the choice

$$\delta_T = \begin{cases} \delta_0 h_T & \text{if } Pe_T > 1, \quad \text{(dominant convection)}, \\ \delta_1 h_T^2/\nu & \text{if } Pe_T \leq 1, \quad \text{(dominant diffusion)}, \end{cases} \tag{3.45}$$

for δ_T, where

$$P_{e_T} := \frac{\|b\|_{0,\infty,T} \, h_T}{2\nu}$$

denotes the *local (mesh) Peclét number* and δ_0 and δ_1 are positive constants (cf. Lube [101]). This choice is still not optimal, since the constants δ_0 and δ_1 must be determined. In general, the optimal choice of δ_T remains an open problem (see below).

3.2.2 Extensions to Nonconforming Elements

The SDFEM formulation (3.25)-(3.28) assumed conforming finite element spaces. John et al. [81] showed that the $\mathcal{O}(h^{k+1/2})$ convergence rate of (3.44) could be preserved for a class of nonconforming elements by adding certain jump terms to the bilinear form; however, because the error analysis relied on the existence of a conforming finite element subspace within the nonconforming approximation space, the method could not be used for the $\tilde{Q}_1$ rotated quadrilateral elements described in Section 2.3.2. The following extension to the $\tilde{Q}_1$ elements was made by John et al. in [80] through the use of more flexible jump terms.

Find $u_h \in V_h \not\subset V$, such that for all $v_h \in V_h$

$$a_h(u_h, v_h) = l_h(v_h), \tag{3.46}$$

where the bilinear form a_h and the linear form l_h are given by

$$\begin{aligned} a_h(u_h, v_h) := \sum_{T \in \mathcal{T}^h} \Big(& \nu(\nabla u_h, \nabla v_h)_T + (\mathbf{b} \cdot \nabla u_h + c u_h, v_h)_T && (3.47) \\ & + (-\nu \Delta u_h + \mathbf{b} \cdot \nabla u_h + c u_h, \delta_T \mathbf{b} \cdot \nabla v_h)_T \Big) && (3.48) \\ & + \sum_{E \in \mathcal{E}} \int_E \mathbf{b} \cdot n_E [u_h]_E \, A_E v_h \, d\gamma && (3.49) \\ & + \sum_{E \in \mathcal{E}} \int_E \big|\mathbf{b} \cdot n_E\big| [u_h]_E \, [v_h]_E \, d\gamma, && (3.50) \\ l_h(v_h) := (f, v_h) + \sum_{T \in \mathcal{T}^h} & \delta_T (f, \mathbf{b} \cdot \nabla v_h)_T. && (3.51) \end{aligned}$$

Though more involved because of the jump terms, the coercivity and convergence proofs are otherwise identical to those for the conforming case. We refer the reader to [80] for details, proofs and analysis.

On the Choice of the SD-Parameter δ_T.

The development of the SDFEM and related methods has progressed rapidly over the last 25 years. A comprehensive account of the material on the best choice of δ_T could fill a monograph by itself. We will content ourselves with providing a very short summary of some of the results in this area along with references to further literature on the subject.

Fries/Matthies [46] have reviewed some stabilization methods for convection-dominated problems, including an extensive discussion of methods for deriving δ_T. For one-dimensional problems of the form

$$-\nu u'' + b u' = 0$$

with constant coefficients, the optimal value of

$$\delta_T = \frac{h}{2b}\left(\coth\left(\frac{bh}{2\nu}\right) - \frac{2\nu}{bh}\right) = \frac{h}{2b}\left(\coth\left(Pe\right) - \frac{1}{Pe}\right), \tag{3.52}$$

where h is the mesh size, can be calculated and yields the so-called *Il'in-Allen-Southwell-scheme*, which is nodally exact. Moreover, the value of δ_T in (3.52) is independent of boundary conditions and dependent on the relative positions of the two neighboring nodes only [122]. Fries/Matthies [46] summarize several methods for obtaining (3.52), all of which involve knowledge of the exact solution. Practically speaking, it can be advantageous from a computational point of view to replace the optimal version (3.52) of δ_T with an approximation that can be computed more quickly (cf. [124, 45, 44, 133, 132, 134, 46] and the references contained therein).

The optimal choice of the streamline diffusion parameter δ_T for linear convection-diffusion problems in higher dimensions remains an open problem, depending on the particular problem and on the numerical method used to approximate the solution, with a variety of methods used to make the choice [46]; however, based on the mathematical analysis in the literature one generally finds $\delta_T = \mathcal{O}(h^2)$ for the diffusion-dominated case and $\delta_T = \mathcal{O}(h)$ for the more interesting convection-dominated case. Note that (3.40) conforms with these results.

Remark 3.7. Referring to (3.40), we note that the streamline diffusion parameter δ_T can in principle be chosen independently of $T \in \mathcal{T}^h$, while maintaining the theoretical convergence properties of Theorem 3.4. Such a choice will likely be suboptimal compared to (3.45), but simplifies the implementation and is advantageous from the standpoint of the modified POD methods we will present later.

3.3 SDFEM for Navier-Stokes Problems

We now turn our attention to a nonlinear convection-diffusion problem, the Navier-Stokes equations. Compared to the linear convection-diffusion problems of Section 3.2, the Navier-Stokes equations present additional difficulties, not just because of their nonlinear nature, but because of additional sources of numerical instabilities associated with the incompressibility condition as well. As mentioned in Section 2.1, finite element approximations of Navier-Stokes problems generally use mixed finite element methods; that is, different finite element spaces for the velocity and pressure approximations. Numerical oscillations in such mixed methods might be generated not only by the presence of dominant convection, such that for the local Reynolds number Re_T

$$Re_T := \nu^{-1} \|\mathbf{u}\|_{0,\infty,T} \, h_T > 1, \tag{3.53}$$

holds, but also by inappropriate combinations of velocity/pressure interpolation functions that do not satisfy the Babuška-Brezzi condition (3.10) with a mesh independent constant γ.

Corresponding to the exceptional challenges presented by the numerical solution of the Navier-Stokes equations, a large number of streamline diffusion methods have been proposed and analyzed over the last 30 years. The number and technical complexity of the approaches is such that we must content ourselves here with no more than a cursory review of the proposed methods.

We begin by citing a small number of the many streamline diffusion techniques that have been introduced and studied for the incompressible Navier-Stokes equations, then move on to a (slightly) more detailed description of some formulations that are of particular interest to us. Some of the earliest work in this area was done by Johnson/Saranen [83], who suggested a streamline diffusion formulation for the time-dependent two-dimensional Navier-Stokes equations, assuming exactly divergence-free discrete velocity fields. Hansbo/Szepessy [67] later introduced and analyzed a streamline diffusion method for the incompressible Navier-Stokes equations in the velocity-pressure formulation. In [102], Lube and Tobiska circumvented exact divergence-free velocity fields by utilizing a nonconforming streamline diffusion finite element method. The review by Tezduyar [132] provides an extensive summary of some of the other early work in this area.

A Streamline Diffusion Formulation for the Stationary Navier-Stokes Problem

Building on work by Tobiska/Lube [136], Tobiska/Verfürth [137] proposed a streamline diffusion technique for the generalized stationary Navier-Stokes equations:

$$\begin{aligned} \tilde{\theta}\mathbf{u} - \nu\Delta\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla\tilde{p} &= \tilde{\mathbf{f}} \text{ in } \Omega \\ \nabla\cdot\mathbf{u} &= 0 \text{ in } \Omega \\ \mathbf{u} &= 0 \text{ on } \Gamma, \end{aligned} \tag{3.54}$$

where Ω is a bounded polyhedral domain in $\mathbb{R}^d$, $d = 2, 3$, with boundary Γ and $\tilde{\mathbf{f}} \in L^2(\Omega)^d$. This formulation of the Navier-Stokes problem is of special interest because it allows treatment of the case in which the stationary problem is obtained by the time discretization of a nonstationary Navier-Stokes problem (see Section 2.1). The following summary follows closely the expositions in [137] and [122].

By setting $\tilde{p} = \nu p$, $\tilde{\mathbf{f}} = \nu\mathbf{f}$, $\tilde{\theta} = \theta\nu$ and $\lambda = \nu^{-1}$ in (3.54), we obtain the scaled form

$$\begin{aligned} \theta\mathbf{u} - \Delta\mathbf{u} + \lambda\mathbf{u}\cdot\nabla\mathbf{u} + \nabla p &= \mathbf{f} \text{ in } \Omega \\ \nabla\cdot\mathbf{u} &= 0 \text{ in } \Omega \\ \mathbf{u} &= 0 \text{ on } \Gamma \end{aligned} \tag{3.55}$$

of the Navier-Stokes equations, which is better suited to the approximation of nonsingular branches of solutions to nonlinear problems [20].

The generalized spaces used for the continuous problem are $V := H_0^1(\Omega)^d$ for the velocity and $Q := L_0^2(\Omega) := \{q \in L^2(\Omega) \mid (q, 1) = 0\}$ for the pressure. Using these spaces, the standard weak formulation of problem (3.55) is:

Find a pair $[\mathbf{u}_\lambda, p_\lambda] \in V \times Q$ *such that*

$$a([\mathbf{u}_\lambda, p_\lambda], [\mathbf{v}, q]) = (\mathbf{f}, \mathbf{v}) \quad \forall\, [\mathbf{v}, q] \in V \times Q, \tag{3.56}$$

where

$$a([\mathbf{u}_\lambda, p_\lambda], [\mathbf{v}, q]) = \theta(\mathbf{u}_\lambda, \mathbf{v}) + (\nabla \mathbf{u}_\lambda, \nabla \mathbf{v}) + \lambda(\mathbf{u}_\lambda \cdot \nabla \mathbf{u}_\lambda, \mathbf{v}) - (p_\lambda, \nabla \cdot \mathbf{v}) + (q, \nabla \cdot \mathbf{u}_\lambda).$$

The following SDFEM for (3.55) is obtained by testing the momentum equation against test functions of the form $\lambda \mathbf{u} \cdot \nabla \mathbf{v} + \nabla q$ and adding least-squares control of the divergence:

Find a pair $[\mathbf{u}_{h,\lambda}, p_{h,\lambda}] \in V_h \times Q_h$ *such that*

$$a_{\delta,\alpha}([\mathbf{u}_{h,\lambda}, p_{h,\lambda}], [\mathbf{v}_h, q_h]) = l_\delta(\mathbf{u}_{h,\lambda}, \mathbf{v}_h) \quad \forall\, [\mathbf{v}_h, q_h] \in V_h \times Q_h, \tag{3.57}$$

where

$$\begin{aligned} a_{\delta,\alpha}([\mathbf{u}_{h,\lambda}, p_{h,\lambda}],[\mathbf{v}_h, q_h]) &= a([\mathbf{u}_{h,\lambda}, p_{h,\lambda}], [\mathbf{v}_h, q_h]) \\ &+ \delta \sum_{T \in \mathcal{T}^h} h_T^2 (\theta \mathbf{u}_{h,\lambda} - \Delta \mathbf{u}_{h,\lambda} + \lambda \mathbf{u}_{h,\lambda} \cdot \mathbf{u}_{h,\lambda} + \nabla p_{h,\lambda}, \lambda \mathbf{u}_{h,\lambda} \cdot \nabla \mathbf{v}_h + \nabla q_h)_T \\ &+ \delta \sum_{E \in \mathcal{E}^h} h_E ([p_{h,\lambda}]_E, [q_h]_E)_E + \alpha\delta(\nabla \cdot \mathbf{u}_{h,\lambda}, \nabla \cdot \mathbf{v}_h) \end{aligned}$$

and

$$l_\delta(\mathbf{u}_{h,\lambda}, \mathbf{v}_h) = (\mathbf{f}, \mathbf{v}_h) + \delta \sum_{T \in \mathcal{T}^h} h_T^2 (\mathbf{f}, \lambda \mathbf{u}_{h,\lambda} \cdot \nabla \mathbf{v}_h + \nabla q_h)_T.$$

The values $\alpha \geq 0$, $\delta > 0$ denote parameters that are independent of h and must be chosen to satisfy $\delta \leq \frac{1}{2}\mu^{-2}$ and $\delta\theta h^2 \leq \frac{1}{2}$, where μ is the constant from the inverse estimate (3.17). The pressure jumps across interelement boundaries are needed to allow discontinuous pressure approximations.

Analysis by Tobiska/Verfürth [137] shows that the velocity-pressure formulation (3.57) is sufficient to stabilize both the instability caused by dominant convection and the instability resulting from velocity/pressure approximations that do not fulfill condition (3.10); thus (3.57) allows arbitrary combinations of velocity and pressure spaces. The following existence and uniqueness result holds.

Theorem 3.8 (Tobiska/Verfürth). *There is a constant* ζ*, independent of* h *and* λ*, such that the problem (3.57) admits at least one solution* $[\mathbf{u}_{h,\lambda}, p_{h,\lambda}]$ *provided*

$$\lambda h^{1-\kappa} \left(\|\mathbf{f}\|_{-1}^2 + \delta \sum_{T \in \mathcal{T}^h} h_T^2 \, \|\mathbf{f}\|_{0,T}^2 \right)^{1/2} \leq \zeta,$$

with κ *from (3.18). Moreover, the solution of (3.57) is unique provided* λ *is sufficiently small.*

Remark 3.9. Note that in contrast to the scalar linear convection-diffusion problem studied in Section 3.2, the streamline diffusion parameter δ in (3.57) is independent of the local Reynolds number. We refer to [137]) for additional thoughts on the appropriate choice of δ for the scheme (3.57).

The FEATFLOW SDFEM Implementation

Since the FEATFLOW implementation relies on nonconforming elements, the streamline diffusion method used must be tailored to these elements. By modifying a Galerkin least-square finite element method introduced by Lube [101] for the stationary Navier-Stokes equations, Schreiber/Turek [123, 142] formulated a least-square streamline diffusion method for the nonconforming $\tilde{Q}_1/Q_0$-element, in which a residual term is added to the momentum equation in the discrete formulation of the stationary Navier-Stokes problem as follows:

Find $\mathbf{u}_h \in V_h := H_0^1(\Omega)^2$ *and* $p_h \in Q_h := L_0^2(\Omega)$*, such that*

$$\begin{aligned} \nu a(\mathbf{u}_h, \mathbf{v}_h) + n(\mathbf{u}_h, \mathbf{u}_h, \mathbf{v}_h) + b(\mathbf{v}_h, p_h) & \\ + \sum_{T \in \mathcal{T}^h} (-\nu \Delta \mathbf{u}_h + \mathbf{u}_h \cdot \nabla \mathbf{u}_h - \mathbf{f}, \psi(\mathbf{u}_h, \mathbf{v}_h))_{|T} &= (\mathbf{f}, \mathbf{v}_h) \quad \forall\, \mathbf{v}_h \in V_h \qquad (3.58) \\ b(\mathbf{u}_h, q_h) &= 0 \qquad \forall\, q_h \in Q_h. \qquad (3.59) \end{aligned}$$

We note that multiplying the perturbation ψ of the test function with the residual term $-\nu \Delta \mathbf{u}_h + \mathbf{u}_h \cdot \nabla \mathbf{u}_h - \mathbf{f}$ ensures consistency since the modified problem is also satisfied by the continuous solution. Now, choosing

$$\psi(\mathbf{u}_h, \mathbf{v}_h)_{|T} = \delta_T \mathbf{u}_h \cdot \nabla \mathbf{v}_h + \gamma_T(-\nu \Delta \mathbf{v}_h) \quad \forall\, T \in \mathcal{T}^h \qquad (3.60)$$

gives extra control over gradients in the streamline direction, and using the special characteristics ($\Delta \mathbf{v}^h = 0$, $\nabla q = 0$) of the $\tilde{Q}_1/Q_0$-element and assuming $\mathbf{f} = 0$ leads to the following discrete streamline diffusion formulation of the stationary Navier-Stokes problem:

Find $\mathbf{u}_h \in V_h$ *and* $p_h \in Q_h$*, such that*

$$\begin{aligned} \nu a(\mathbf{u}_h, \mathbf{v}_h) + \tilde{n}(\mathbf{u}_h, \mathbf{u}_h, \mathbf{v}_h) + b(\mathbf{v}_h, p_h) &= 0 \quad \forall\, \mathbf{v}_h \in V_h \qquad (3.61) \\ b(\mathbf{u}_h, q_h) &= 0 \quad \forall\, q_h \in Q_h, \qquad (3.62) \end{aligned}$$

where

$$\tilde{n}(\mathbf{u}_h, \mathbf{v}_h, \mathbf{w}_h) = n(\mathbf{u}_h, \mathbf{v}_h, \mathbf{w}_h) + \sum_{T \in \mathcal{T}^h} \delta_T (\mathbf{u}_h \cdot \nabla \mathbf{v}_h, \mathbf{u}_h \cdot \nabla \mathbf{w}_h)_{|T}. \qquad (3.63)$$

For the choice of the local damping parameter δ_T, Turek [142] recommends the value

$$\delta_T = \delta \cdot \frac{h_T}{\| \mathbf{u} \|_\Omega} \cdot \frac{2Re_T}{1 + Re_T}, \qquad (3.64)$$

where $Re_T = \|\mathbf{u}\|_T \cdot h_T / \nu$ is the local Reynolds number, $\|\mathbf{u}\|_T$ is an averaged velocity value over the quadrilateral T, h_T denotes a "local mesh width," $\|\mathbf{u}\|_\Omega$ is the maximum velocity norm on Ω, ν is the viscosity and δ is a parameter to be chosen by the user, with typical values between 0.1 and 2.0.

Chapter 4

Proper Orthogonal Decomposition

Accurate *direct numerical simulation (DNS)* of fluid flows governed by the nonstationary Navier-Stokes equations requires fine spatial discretization, leading in general to very large nonlinear systems, which must be solved at each time step. The resulting computational complexity makes it difficult to use DNS in settings (optimal control, optimization, etc.) that require repeated solution of the Navier-Stokes equations to determine the system state. This has led researchers to seek *reduced-order models (ROM)* that can serve as low-dimensional approximations to the discretized Navier-Stokes equations.

One method for generating low-order models that has been studied extensively (e.g., Holmes et al. [72], Sirovich [126], Aubry [8]) is the *proper orthogonal decomposition (POD)*, known in other contexts as *principle components analysis*, *Karhunen-Loève decomposition* or the *method of empirical eigenfunctions*. Generally speaking, the POD method uses data generated either experimentally or from the numerical solution of the system of interest to build an orthonormal system of basis elements that reflect the salient characteristics of the expected solution. The POD basis elements are optimal in the sense that they capture more of the system energy than any other admissible basis of the same dimension. POD is especially attractive as a method for deriving low-order models because it is in its nature a linear procedure, though it makes no assumptions about the linearity of the problem to which it is applied. POD has been used successfully in a range of applications including control theory, signal analysis and feedback design. Some recent examples include Leibfritz/Volkwein [97, 98] and Volkwein [147], and the examples cited therein.

In the Navier-Stokes context the POD method assumes that the velocity can be written as a linear combination of the POD basis functions. To determine the coefficients of this linear expansion, the dynamical system is projected onto the POD basis, resulting in a nonlinear system of ordinary differential equations that can be solved to determine the coefficients of the representation. The optimal correlation of the POD basis elements with properties of the flow field results in a considerable reduction in the degrees of freedom needed to represent the flow as compared to other techniques, e.g., finite-element methods, where the basis functions are uncorrelated with the physical properties of the system being simulated.

We begin the discussion of POD methods in Section 4.1, where the POD basis is derived from a minimization problem based on the approximation error between the snapshot ensemble and the projection of the snapshots onto the POD basis. A procedure for calculating the POD basis is presented and certain characteristics of the POD basis, such as its superiority to other linear decompositions of the snapshot ensemble, are discussed. Since we are interested in using POD-based models derived from data acquired from finite element approximations on meshes at differing refinement levels, we study certain convergence aspects of the Galerkin POD model in Section 4.2, backing up the theoretical results with extensive numerical testing in Section 4.3.

4.1 Mathematical Formulation of the POD

The introduction of the proper orthogonal decomposition in this section is based primarily on presentations by Berkooz et al. [13], Holmes et al. [72], Sirovich [126] and Volkwein [145].

Consider a real separable Hilbert space H endowed with the scalar product $(\cdot,\cdot)_H$ and corresponding norm $\|\cdot\|_H = (\cdot,\cdot)_H^{1/2}$. Given an ensemble of elements (or snapshots) $u_i \in H$, $i = 1,\dots,n$ (with $p = \dim\{u_1,\dots,u_n\} \geq 1$) we seek an orthonormal basis $\{\psi_j\}$ of the subspace $H_n = \operatorname{span}\{u_1,\dots,u_n\}$ that is optimal in the sense that all finite-dimensional representations of the form

$$v_m = \sum_{j=1}^{m} a_j \psi_j, \quad 1 \leq m \leq p \leq n$$

describe an "average" element of H_n better than than any other representation of the same dimension in any other basis. If we understand average here to mean an arithmetical average, we can make a precise mathematical formulation of the POD as follows.

If $\{\psi_j\}_{j=1}^{p}$ is any orthonormal basis for H_n then for any $m \in \{1,\dots,p\}$ we can express the projection $\hat{u}_i$ of each member of the snapshot ensemble $\{u_i\}_{i=1}^{n}$ onto the space spanned by $\{\psi_j\}_{j=1}^{m}$ as

$$\hat{u}_i = \sum_{j=1}^{m} (u_i,\psi_j)_H \psi_j, \quad i = 1,\dots,n. \tag{4.1}$$

The essence of POD lies in choosing the basis so that for every $m \in \{1,\dots,p\}$ the mean square error between the ensemble elements and the sum (4.1) is minimized. This leads to the following problem formulation for the derivation of the POD basis.

Problem 4.1. *Find an orthonormal basis* $\{\psi_j\}_{j=1}^{p}$ *of* H_n *that solves the minimization problem*

$$\min_{\psi_1,\dots,\psi_m} \sum_{i=1}^{n} \omega_i \big\| u_i - \sum_{j=1}^{m} (u_i,\psi_j)_H \psi_j \big\|_H \quad s.t.\ (\psi_i,\psi_j) = \delta_{ij} \tag{4.2}$$

for all $1 \leq i,j \leq m$, $1 \leq m \leq p$ *and* $\omega_i > 0$, $i = 1,\dots,n$.

A solution of (4.2) is called a *POD basis of rank* m. With the weights chosen according to $\omega_i = 1/n$, $i = 1, \dots, n$, we recover the usual notion of the arithmetic average; however, other choices, such as the length of the time interval for data generated at discrete time points, will also be of interest (see Section 4.2.1).

4.1.1 Construction of the POD Basis

To calculate the POD basis we follow Kunisch/Volkwein [93] and define the operator $\mathcal{U}_n \in \mathcal{L}(\mathbb{R}^n, H)$ by

$$\mathcal{U}_n a := \sum_{i=1}^{n} \omega_i a_i u_i \quad \text{for } a = (a_1, \dots, a_n)^T \in \mathbb{R}^n, \tag{4.3}$$

where $\mathcal{L}(\mathbb{R}^n, H)$ denotes the space of bounded linear operators from $\mathbb{R}^n$ into H. If $\mathbb{R}^n$ is endowed with the inner product

$$\langle a, b \rangle_{\mathbb{R}^n} := \sum_{i=1}^{n} \omega_i a_i b_i \quad \text{for } a, b \in \mathbb{R}^n, \tag{4.4}$$

then it is easily seen by

$$\big(\mathcal{U}_n a, w\big)_H = \big(\sum_{i=1}^{n} \omega_i a_i u_i, w\big)_H = \sum_{i=1}^{n} \omega_i a_i \big(u_i, w\big)_H, \quad a \in \mathbb{R}^n,\ w \in H,$$

and (4.4) that the adjoint $\mathcal{U}_n^* \in \mathcal{L}(H, \mathbb{R}^n)$ is given by

$$\mathcal{U}_n^* w := ((u_1, w)_H, \dots, (u_n, w)_H)^T \quad \forall\, w \in H. \tag{4.5}$$

With the above notation, the *autocorrelation operator* $\mathcal{R}_n := \mathcal{U}_n \mathcal{U}_n^* \in \mathcal{L}(H)$ is given by

$$\mathcal{R}_n w = \sum_{i=1}^{n} \omega_i (w, u_i)_H u_i, \quad \forall\, w \in H. \tag{4.6}$$

Example 4.2. Consider $n > 0$ elements $u_1, \dots, u_n$ from the Hilbert space $H = L^2(\Omega)$, where Ω is a bounded domain in $\mathbb{R}^d$, $d \in \mathbb{N}$. Then the autocorrelation operator $\mathcal{R}_n$ is given by

$$\begin{aligned} \mathcal{R}_n w(x) &= \sum_{i=1}^{n} \omega_i (w, u_i)_H u_i(x) \\ &= \sum_{i=1}^{n} \omega_i \Big(\int_\Omega w(x') u_i(x')\, dx' \Big) u_i(x) \\ &= \int_\Omega k(\cdot, x') w(x')\, dx' \quad \forall\, w \in H, \end{aligned}$$

where the kernel

$$k(x, x') = \sum_{j=1}^{n} \omega_i u_j(x) u_j(x')$$

is known as the *averaged autocorrelation function.*

The solution of Problem 4.1 is contained in the following theorem from Volkwein [145].

Theorem 4.3. *There exists a complete orthonormal basis $\{\psi_j\}_{j\in\mathbb{N}}$ for H and a sequence $\{\lambda_j\}_{j\in\mathbb{N}}$ of nonnegative real numbers, such that*

$$\mathcal{R}_n\psi_j = \lambda_j\psi_j, \quad \textit{with } \lambda_1 \geq \cdots \geq \lambda_p > 0 \quad \textit{and } \lambda_j = 0 \textit{ for } j > p \tag{4.7}$$

and $H_n = span\{\psi_j\}_{j=1}^p$. Moreover, $\{\psi_j\}_{j=1}^m$ is a POD basis of rank m for $1 \leq m \leq p$, that is, $\{\psi_j\}_{j=1}^p$ solves Problem 4.1, and the accumulated mean square error (truncation error) for each partial sum is given by the POD representation error

$$\epsilon(m) := \sum_{i=1}^n \omega_i \Big\| u_i - \sum_{j=1}^m (u_i,\psi_j)_H\psi_j \Big\|_H^2 = \sum_{j=m+1}^p \lambda_j, \tag{4.8}$$

where $\lambda_{m+1},\dots,\lambda_p$ are the smallest $p-m$ eigenvalues of $\mathcal{R}_n$.

Proof. We give a short sketch of the detailed proof in [145]. By formulating and solving the Lagrangian of the constrained minimization problem (4.2) for any $m \in \{1,\dots,p\}$, one derives the necessary optimality conditions

$$\mathcal{R}_n\psi_j = \lambda_j\psi_j \quad \text{for } j = 1,\dots,m \tag{4.9}$$

(see also [72]). The operator $\mathcal{R}_n$ can be shown to be bounded, self-adjoint, nonnegative and compact, so that the Hilbert-Schmidt theorem (cf. Reed/Simon [121, Theorem VI.16]) guarantees existence of a complete orthonormal basis $\{\psi_j\}_{j\in\mathbb{N}}$ for H and a sequence $\{\lambda_j\}_{j\in\mathbb{N}}$, such that

$$\mathcal{R}_n\psi_j = \lambda_j\psi_j, \qquad \lambda_1 \geq \lambda_2 \geq \cdots \quad \text{and } \lambda_j \to 0 \text{ for } j \to \infty. \tag{4.10}$$

The assertion (4.7) results from the degeneracy of $\mathcal{R}_n$ (rank $\mathcal{R}_n = p$). The proof that the set (4.10) is sufficient to solve Problem 4.1 is straightforward, but technical; we refer the reader to [145].

The derivation of the error formula uses the relation

$$\sum_{i=1}^n \omega_i \left|(u_i,\psi_j)_H\right|^2 = \lambda_j \quad \forall j \in \mathbb{N}, \tag{4.11}$$

which follows from (4.10) and the definition of $\mathcal{R}_n$. It follows that

$$\begin{aligned}
&\sum_{i=1}^n \omega_i \Big\| u_i - \sum_{j=1}^m (u_i,\psi_j)_H\psi_j \Big\|_H^2 \\
&\quad = \sum_{i=1}^n \omega_i \Big(\|u_i\|_H^2 - \sum_{j=1}^m |(u_i,\psi_j)_H|^2 \Big) \\
&\quad = \sum_{i=1}^n \omega_i \Big(\sum_{j=1}^p |(u_i,\psi_j)_H|^2 - \sum_{j=1}^m |(u_i,\psi_j)_H|^2 \Big)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{j=m+1}^{p} \sum_{i=1}^{n} \omega_i \left|(u_i, \psi_j)_H\right|^2 \\
&= \sum_{j=m+1}^{p} \lambda_j,
\end{aligned}
$$

□

The Method of Snapshots

The procedures of Theorem 4.3 present some difficulties. Even if, as will be motivated in Section 4.1.3, only a small number of basis functions are necessary to represent the function or flow in question, we must still solve an eigenvalue problem (4.9) of order equal to that of the original problem. One possibility lies in using iterative numerical procedures, Lanczos methods for instance (see Fahl [40]), that allow extraction of the eigenvectors corresponding to only the largest eigenvalues – those capturing the most system energy – at reduced computational cost; however, for situations in which the number of observations in the POD ensemble is much smaller than the dimension of the space from which the observations are extracted, the so-called *method of snapshots*, introduced by Sirovich [126], provides an efficient and elegant method for calculating all positive eigenvalues and the corresponding eigenfunctions.

Recalling the definition of the autocorrelation operator (4.6), we define the *weighted correlation matrix* $\mathcal{K}_n := \mathcal{U}_n^* \mathcal{U}_n \in \mathbb{R}^{n \times n}$, which can be written as

$$
(\mathcal{K}_n)_{ij} = \omega_j (u_i, u_j)_H. \tag{4.12}
$$

The matrix $\mathcal{K}_n$ is clearly symmetric positive semi-definite with rank p, and Problem 4.1 can now be solved by utilizing the following theorem, a proof of which can be found in [72] or [145].

Theorem 4.4. *Let $\lambda_1, \dots, \lambda_p > 0$ be the positive eigenvalues of the weighted correlation matrix $\mathcal{K}_n$ and $v_1, \dots, v_p$ the corresponding eigenvectors. A POD basis of rank $m \leq p$ is given by*

$$
\psi_i = \frac{1}{\sqrt{\lambda_i}} \sum_{j=1}^{n} v_{ij} u_j, \quad i = 1, \dots, m \tag{4.13}
$$

where v_{ij} is the j-th component of v_i, and the POD representation error is given by (4.8).

The eigenvalue problem for $\mathcal{K}_n$ can be solved numerically. For ensembles drawn from an infinite dimensional Hilbert space, the elements of $\mathcal{K}_n$ must themselves be computed by numeric quadrature.

4.1.2 Properties of the POD Basis

In this section we summarize some of the properties of POD that make it attractive from the standpoint of reduced-order modeling. As above, we will be interested only in finite dimensional POD subspaces and will formulate our results accordingly, but the theory and comments of this section can also be extended to the ideal case of infinite data sets (cf. Holmes, et al. [72]).

Span of the POD Basis

From Theorem 4.3, we have $H_n = \operatorname{span}\{u_1, \ldots, u_n\} = \operatorname{span}\{\psi_1, \ldots, \psi_p\}$, where $\{u_i\}_{i=1}^n$ is the snapshot ensemble and $\{\psi_i\}_{i=1}^p$ is the maximal POD basis, so that every member of the original snapshot ensemble can be reconstructed as a linear combination of the POD basis, and conversely, the POD basis elements can be expressed as linear combinations of the snapshots. The second observation is especially encouraging since it implies that any property of the snapshots that is preserved under linear combination is inherited by the POD basis functions. This includes such features as incompressibility and linear boundary conditions. Furthermore, since $p \leq n$, the POD basis will generally use fewer – sometimes far fewer – elements to represent the space spanned by the snapshots.

Optimality of the POD Basis

Generalizing the discussion somewhat, we follow Holmes et al. [72] and suppose we have some time dependent signal $u(t, x)$ mapping into some Hilbert space H for each t, which we wish to approximate linearly with respect to an arbitrary orthonormal basis $\{\phi_j(x)\}_{j=1}^p$ spanning a finite dimensional subspace of H:

$$u(t, x) = \sum_{j=1}^{p} b_j(t)\phi_j(x); \tag{4.14}$$

for instance, a Fourier series approximation for the solution of a time-dependent partial differential equation (cf. Haberman [66]).

Now, if the $\phi_j(x)$ are dimensionless, then the coefficients $b_j(t)$ carry the dimension of the quantity u. If, for instance, $u(t, x)$ denotes the velocity in a flowfield and $\langle\cdot\rangle$ denotes an averaging operation with respect to t, which is assumed to commute with the inner product $(\cdot, \cdot)_H$, then the average kinetic energy per unit mass for the flow field is given (with slight abuse of notation) by

$$\begin{aligned}\frac{1}{2}\left\langle (u(t, x), u(t, x))_H \right\rangle &= \frac{1}{2}\left\langle \sum_{j=1}^{p}\sum_{k=1}^{p} b_j(t)b_k(t)(\phi_j(x), \phi_k(x))_H \right\rangle \\ &= \frac{1}{2}\sum_{j=1}^{p}\left\langle b_j(t)b_j(t)\right\rangle\end{aligned}$$

so that the average kinetic energy in the j-th mode is given by $\frac{1}{2}\langle b_j^2(t)\rangle$.

The following proposition, the proof of which can be found e.g., in Berkooz et al. [13], establishes the optimality of the POD decomposition.

Proposition 4.5. *Let $\{\psi_j\}_{j=1}^p$ denote the orthonormal POD basis and $\{\lambda_j\}_{j=1}^p$ the associated set of eigenvalues. If*

$$u(t,x) = \sum_{j=1}^{p} a_j(t)\psi_j(x)$$

is the decomposition of $u(t,x)$ with respect to the POD basis, then the following hold:

1) $\langle a_j(t)a_k(t)\rangle = \delta_{jk}\lambda_j$, i.e. the POD coefficients are uncorrelated.

2) For every $1 \leq m \leq p$, we have

$$\sum_{j=1}^{m}\langle a_j(t)a_j(t)\rangle = \sum_{j=1}^{m}\lambda_j \geq \sum_{j=1}^{m}\langle b_j(t)b_j(t)\rangle. \tag{4.15}$$

This proposition shows that the POD basis is optimal among all linear decompositions in the sense that, for a given number of modes, the projection on the subspace used for modeling will capture the most kinetic energy in the average sense implied by the operator $\langle\cdot\rangle$.

Remark 4.6. Proposition 4.5 also implies that the eigenvalues λ_j associated with the POD basis give a measure of the mean system energy captured by the associated eigenfunction, or as stated by Sirovich [126], that the eigenvalue λ_j, $j \in 1,\ldots,p$ measures in a certain sense the average relative time spent by the dynamical system along the ψ_j-axis. As a result, one can expect the sum of the eigenvalues to equal the mean system energy. The eigenvalues typically fall rapidly toward zero, so that most of the energy is captured by the largest $m \ll p$ positive eigenvalues, which allows further reduction in the order of the POD model through truncation of the POD representation (see Subsection 4.1.3).

4.1.3 The Dimension of the POD Subspace

As indicated in Remark 4.6, we can control the accuracy and order of the POD model — equivalently, the dimension of the POD basis — by the choice of m in (4.8). The choice of the POD dimension is usually based on some sort of heuristic, the most popular being the so-called *energy method* [126]. By choosing m such that

$$\frac{\sum_{i=1}^{m}\lambda_i}{\sum_{i=1}^{p}\lambda_i} \geq \bar{e}, \tag{4.16}$$

where $\bar{e}$ is a predetermined percentage of the system energy, we achieve a further reduction in the dimension of our model with little or no loss in fidelity. Typical choices for $\bar{e}$ range from $98 - 99.9\%$ (see e.g., [97, 98, 103, 7, 55, 13]). Some justification for this procedure is given in Chatelin [30], where it is proved that while $cond(\lambda) = 1$ for an eigenvalue of the symmetric matrix $\mathcal{K}_n$, the condition number of the corresponding eigenvector v is given by

$$cond(v) = 1/d(\lambda), \quad \text{where } d(\lambda) := \min_{\mu\in\sigma(\mathcal{K}_n)\backslash\lambda} |\mu - \lambda|\,, \tag{4.17}$$

and $\sigma(\mathcal{K}_n)$ denotes the set of positive eigenvalues of $\mathcal{K}_n$. Since small eigenvalues will necessarily be close together, we can expect the eigenvectors corresponding to small eigenvalues to be numerically instable, so that dropping them from the POD basis may actually result in *improved* model fidelity as opposed to using the full set of positive eigenvalues.

A discussion of other truncation criteria based on the numerical rank of $\mathcal{K}_n$ can be found in Hansen [68].

4.2 Error Estimates for POD Methods

As indicated in Chapter 1, we are interested in using POD models generated from numerical data of of widely differing quality. In this spirit, we wish to assess the dependence of the approximation properties of POD methods on changes in the discretization parameters of the high-order numerical solution process. Though it appears that little research has been done to date in this area, Kunisch, Volkwein and Hinze ([91, 93, 92, 147, 71]) have produced some initial results, which we review in this section.

Let us first choose a (relatively general) example problem and fix some notation. We shall consider the nonstationary Navier-Stokes problem (2.1)-(2.5) of Section 2.1, and its Galerkin finite element approximation, which was discussed in Section 2.3. In the interest of brevity, the notation of Chapter 2 will be assumed in as far as practicable. We will be interested in three separate but related phenomena:

1. The dependence of the POD basis on the intervals at which the POD snapshots are taken (Section 4.2.1).

2. The dependence of the POD basis on the spatial discretization of the domain Ω (Section 4.2.2).

3. Error estimates for Galerkin POD methods (Section 4.2.3).

4.2.1 Perturbation Analysis of the POD Approximation Error

We begin by assuming the snapshot ensemble is taken directly from the spatial solution space of interest, that is, without any spatial discretization error[1], and investigate the dependence of the eigenvalues $\{\lambda_i\}_{i\in\mathbb{N}}$ on the temporal density of the snapshots. To this end, we divide the interval $[0,T]$ for a given $n \in \mathbb{N}$ into a grid $0 = \tau_0 < \tau_1 < \cdots < \tau_n = T$, where $\tau_0, \ldots, \tau_n$ are the time points at which the snapshots are taken. We set $\triangle\tau_j = \tau_j - \tau_{j-1}$, $j = 1, \ldots, n$ with $\tau_{\max} = \max\{\triangle\tau_1, \ldots, \triangle\tau_n\}$ and $\tau_{\min} = \min\{\triangle\tau_1, \ldots, \triangle\tau_n\}$. Our primary interest lies in the behavior of the error term $\epsilon(m) = \sum_{j=m+1}^{p} \lambda_j$ for $\triangle\tau \to 0$, i.e. $n \to \infty$.

[1] In this sense, "solution space of interest" refers not only to infinite dimensional spaces, such as $L^2(\Omega)$, but also to finite dimensional vector spaces resulting from the spatial discretization of an infinite dimensional space.

Consider the bounded linear operator $\mathcal{U} : L^2(0,T;\mathbb{R}) \to H$ defined by

$$\mathcal{U}\pi := \int_0^T \pi(\tau)u(\tau)\,d\tau \quad \text{for } \pi \in L^2(0,T;\mathbb{R}),\ u \in L^2(0,T;H).$$

Analogue to the situation in Section 4.1.1, the adjoint $\mathcal{U}^* : H \to L^2(0,T;\mathbb{R})$ is given (pointwise) by

$$(\mathcal{U}^*w)(\tau) = (w, u(\tau))_H \quad \text{for } w \in H,$$

and for the autocorrelation operator $R = \mathcal{U}\mathcal{U}^* \in \mathcal{L}(H)$ we have

$$Rw = \int_0^T (w, u(\tau))_H u(\tau)\,d\tau \quad \text{for } w \in H.$$

Now, if we choose the weights in (4.2) according to

$$\omega_0 = \frac{\triangle\tau_1}{2}, \quad \omega_j = \frac{\triangle\tau_{j+1} + \triangle\tau_j}{2},\ j = 1,\dots,n-1, \text{ and } \omega_n = \frac{\triangle\tau_n}{2} \tag{4.18}$$

and set

$$\mathcal{I}_n(u) = \sum_{j=0}^{n} \omega_j \big\| u(\tau_j) - \sum_{k=1}^{m} (u(\tau_j), \psi_k)_H \psi_k \big\|_H^2 = \sum_{j=m+1}^{p} \lambda_j, \tag{4.19}$$

$$\mathcal{I}(u) = \int_0^T \big\| u(\tau) - \sum_{k=1}^{m} (u(\tau), \psi_k)_H \psi_k \big\|_H^2\,d\tau \tag{4.20}$$

then $\mathcal{I}_n(u)$ is the trapezoidal approximation for the integral $\mathcal{I}(u)$, and for all $u \in C(0,T;H)$, we have $\lim_{n\to\infty} \mathcal{I}_n(u) = \mathcal{I}(u)$. Likewise, with the weights chosen according to (4.18), the finite dimensional operator $\mathcal{R}_n$ is the trapezoidal approximation to $\mathcal{R}$ and if $u_\tau \in L^2(0,T;H)$ we have $\lim_{n\to\infty} \|\mathcal{R}_n - \mathcal{R}\|_{\mathcal{L}(H)}$.

The following proposition extending the results of Theorem 4.3 to $\mathcal{R}$ and giving perturbation bounds for the eigenvalues of $\mathcal{R}_n$ was proved in [93].

Proposition 4.7 (Kunisch/Volkwein). *There exists a complete orthonormal basis $\{\psi_j\}_{j=1}^\infty$ for H and a sequence $\{\lambda_j\}_{j=1}^\infty$ of nonnegative numbers such that*

$$\mathcal{R}\psi_j = \lambda_j\psi_j, \quad \textit{with } \lambda_1 \ge \lambda_2 \ge \cdots \quad \textit{and } \lim_{j\to\infty}\lambda_j = 0, \tag{4.21}$$

and $\int_0^T \|u(\tau)\|_H^2\,d\tau = \sum_{j=1}^\infty \lambda_j$ for $u \in C(0,T;H)$.

Moreover, if we let $\{\lambda_j^n\}_{j=1}^\infty$ denote the sequence (4.7) from Theorem 4.3 and choose a fixed $m \in \mathbb{N}$ such that $\lambda_m \ne \lambda_{m+1}$, then

$$\lim_{\triangle\tau\to 0} \sum_{j=1}^{\infty} \lambda_j^n = \sum_{j=1}^{\infty} \lambda_j, \tag{4.22}$$

$$\lim_{\triangle\tau\to 0} \lambda_j^n = \lambda_j, \quad \textit{for } 1 \le j \le m, \tag{4.23}$$

and if $\sum_{j=m+1}^{\infty} \lambda_j > 0$*, there exists a* $\varpi_1 > 0$*, such that*

$$(4.24)$$

$$\sum_{j=m+1}^{\infty} \lambda_j^n \leq 2 \sum_{j=m+1}^{\infty} \lambda_j \quad \textit{for } \triangle\tau \leq \varpi_1. \tag{4.25}$$

Finally, for m *as chosen above, there is a* $\varpi_2 > 0$*, such that*

$$\sum_{j=m+1}^{p(n)} \left|(\psi_j^n, u_0)_H\right|^2 \leq 2 \sum_{j=m+1}^{\infty} |(\psi_j, u_0)_H|^2 \quad \textit{for } \triangle\tau \leq \varpi_2, \tag{4.26}$$

where u_0 *is the snapshot taken at time* t_0*.*

4.2.2 The Effect of Spatial Discretizations on the POD Basis

Consider the Galerkin approximation of an infinite dimensional Hilbert space H by a family of finite dimensional subspaces H^h given by

$$H^h = \operatorname{span}\{\varphi_1, \dots, \varphi_N\}, \quad N \in \mathbb{N},$$

where the set $\{\varphi_1, \dots, \varphi_N\}$ is linearly independent in H, and the mesh parameter $h = h(N) > 0$ is a measure of the mesh size with accumulation point zero. We define the family $\{\Pi^h\}_h$ by

$$\Pi^h u = \sum_{i=1}^{N}\sum_{j=1}^{N} (\mathrm{M}^{-h})_{ij}(u, \varphi_j)_H \varphi_i \quad \forall\, u \in H,$$

where $\mathrm{M}^h = ((\varphi_i, \varphi_j)_H)_{i,j=1}^N \in \mathbb{R}^{N\times N}$ denotes the finite element mass matrix in H^h, and M^{-h} its inverse. It is easily seen that Π^h is the bounded orthogonal projection of H onto H^h for each $h > 0$ (cf. [147]). It is worth emphasizing here that N is the order of the finite element discretization, and we have $h \to 0$ for $N \to \infty$, while n denotes the number of snapshots, which shall remain constant in this section.

Assume now that a fixed number of snapshots $\{u_1, \dots, u_n\}$ are taken from H at $n > 0$ fixed time points in $[0, T]$. Each member u_i of the ensemble is approximated in H^h by the projection of itself onto the space H^h:

$$u_j^h = \Pi^h u_j \in H^h, \quad j = 1, \dots, n. \tag{4.27}$$

Using the sets $\{u_1, \dots, u_n\}$ and $\{u_1^h, \dots, u_n^h\}$, define the family $\{\mathcal{K}(h)\}_h$ of matrices by (cf. (4.12))

$$\begin{aligned} (\mathcal{K}(h))_{ij} &= \omega_j(u_i^h, u_j^h)_H \quad \text{and} \\ (\mathcal{K}(0))_{ij} = \mathcal{K}_{ij} &= \omega_j(u_i, u_j)_H \end{aligned} \tag{4.28}$$

for $\{u_1, \dots, u_n\} \in H$. Then the results of Section 4.1.1, including Theorem 4.4, apply and we have the following proposition from Volkwein [147], giving a sufficient condition for the right-continuity of $\mathcal{K}(h)$ at $h = 0$.

Proposition 4.8 (Volkwein). *If the family of restrictions $\{\Pi^h\}_h$ is pointwise convergent in H, that is,*

$$\lim_{h\to 0} \Pi^h u = u \quad \forall u \in H, \tag{4.29}$$

then the family $\{\mathcal{K}(h)\}_h$ defined in (4.28) is right continuous at $h = 0$. Moreover, if there is an $\epsilon > 0$ such that

$$\max_{1\le j\le m} \left\|\Pi^h u_j - u_j\right\|_H = \mathcal{O}(h^\epsilon) \textit{ for } h \to 0, \tag{4.30}$$

then

$$\|\mathcal{K} - \mathcal{K}(h)\|_2 = \mathcal{O}(h^\epsilon) \textit{ for } h \to 0, \tag{4.31}$$

where $\|\cdot\|_2$ denotes the spectral norm for matrices.

Proof. With $k := \arg \max_{1\le i\le m} \omega_i \|u_i\|_H$ we have

$$\begin{aligned}
\|\mathcal{K} - \mathcal{K}(h)\|_2 \le \|\mathcal{K} - \mathcal{K}(h)\|_\infty &= \max_{1\le i\le m} \sum_{j=1}^m \left|\mathcal{K}_{ij} - \mathcal{K}_{ij}^h\right| \\
&\le \max_{1\le i\le m} \omega_i \sum_{j=1}^m \left(\left|(u_i, u_j - \Pi^h u_j)\right| + \left|(u_i - \Pi^h u_i, u_j)\right|\right) \\
&\le \max_{1\le i\le m} \omega_i \sum_{j=1}^m \left(\|u_i\|_H \left\|u_j - \Pi^h u_j\right\|_H + \left\|\Pi^h u_j\right\|_H \left\|u_i - \Pi^h u_i\right\|_H\right) \\
&\le \|u_k\|_H \sum_{j=1}^m \left\|u_j - \Pi^h u_j\right\|_H + m\,\|u_k\|_H \left\|\Pi^h\right\|_{\mathcal{L}(H,H^h)} \left\|u_i - \Pi^h u_i\right\|_H .
\end{aligned}$$

It follows from (4.29) that $\left\|\Pi^h\right\|_H$ is bounded for all $u \in H$, so that according to the principle of uniform boundedness (cf. Reed/Simon [121, Theorem III.9]) there is a constant $C > 0$ such that $\left\|\Pi^h\right\|_{\mathcal{L}(H,H^h)} \le C$ for all h. Hence, we have

$$\lim_{h\to 0} \|\mathcal{K} - \mathcal{K}(h)\|_2 = 0.$$

The second assertion is now obvious. □

Note that if the snapshot set from H is linearly independent, then $\mathcal{K}(0)$ will be positive definite, while the matrices $\mathcal{K}(h)$, $h > 0$ will in general be only positive semidefinite. If (4.29) holds, however, $\mathcal{K}(h)$ will be positive definite for sufficiently small h, and the set $\{\Pi^h u_j\}_{j=1}^n$, where $\{u_1, \ldots, u_n\}$ is the snapshot set from H, will necessarily be linearly independent.

As was mentioned in Section 4.1.3, the condition of the eigenvectors of the family $\{\mathcal{K}_h\}$ depends on the gap of the corresponding eigenvalues. This is made more precise in the following theorem, which also gives a perturbation bound on the eigenvalues. See Demmel [38] for a proof.

Theorem 4.9. *For any $i \in \{1, \ldots, n\}$ let λ_i and $\lambda_i(h)$ represent the i-th eigenvalue of $\mathcal{K}$ and $\mathcal{K}(h)$, respectively. Then $|\lambda_i - \lambda_i(h)| \leq \|\mathcal{K}(0) - \mathcal{K}(h)\|_2$, and if (4.29) and (4.30) hold it follows that*

$$|\lambda_i - \lambda_i(h)| = \mathcal{O}(h^{\epsilon}) \quad \textit{for } h \to 0. \tag{4.32}$$

For any $i \in \{1, \ldots, n\}$ let ψ_i and $\psi_i(h)$ represent the i-th eigenvector of $\mathcal{K}$ and $\mathcal{K}(h)$, respectively. Then

$$\frac{1}{2} sin(2\theta_i) \leq \frac{\|\mathcal{K} - \mathcal{K}(h)\|_2}{\min\limits_{i \neq j} |\lambda_j - \lambda_i|} \quad \textit{if} \min_{i \neq j} |\lambda_j - \lambda_i| > 0 \tag{4.33}$$

and

$$\frac{1}{2} sin(2\theta_i) \leq \frac{\|\mathcal{K} - \mathcal{K}(h)\|_2}{\min\limits_{i \neq j} |\lambda_j(h) - \lambda_i(h)|} \quad \textit{if} \min_{i \neq j} |\lambda_j(h) - \lambda_i(h)| > 0, \tag{4.34}$$

where θ_i denotes the acute angle between ψ_i and $\psi_i(h)$.

4.2.3 Error Estimate for POD Approximations

Consider the solution $u(t, \mathbf{x})$ of the weak formulation (2.1) of the Navier-Stokes problem (2.1)-(2.5) with homogeneous boundary conditions and set $H = \mathbf{H}$, $V = \mathbf{V}$, where $\mathbf{H}, \mathbf{V}$ denote the divergence-free spaces of Subsection 1.3.3. Using the POD basis of Section 4.1 for the spatial approximation and the backward Euler method for time stepping, we wish to get a feel for the error we can expect from the Galerkin POD method of projecting the Navier-Stokes equations onto the POD basis and solving the resulting system of ordinary differential equations for $\mathbf{u}$. Following the discussion of Kunisch/Volkwein [92], assume we have used a snapshot set $\mathcal{S} = \{\mathbf{u}(\tau_0), \ldots, \mathbf{u}(\tau_n)\} \subset V \subset H$ taken at the time points $\{\tau_0, \ldots, \tau_n\}$ given in Section 4.2.1 to generate a POD basis $\{\psi_1, \ldots, \psi_l\}$, $l \leq p = \dim \mathcal{S}$, spanning some subspace $H_l = \text{span}\{\psi_1, \ldots, \psi_l\}$ of the space spanned by the snapshots. For some $m \in \mathbb{N}$ we introduce the time grid

$$0 = t_0 < t_1 < \cdots < t_m = T \tag{4.35}$$

with intervals $\triangle t_j = t_j - t_{j-1}$, $j = 1, \ldots, m$, and set $t_{\max} = \max\{\triangle t_1, \ldots, \triangle t_m\}$ and $t_{\min} = \min\{\triangle t_1, \ldots, \triangle t_m\}$. We assume that $t_{\max}/t_{\min}$ is uniformly bounded with respect to m and relate the time discretizations $\{\tau_j\}_{j=0}^n$ and $\{t_j\}_{j=0}^m$ by defining $\sigma_n = \arg\max\{\#\tau_{\hat{k}} \mid 1 \leq k \leq m\}$, where the symbol $\#$ denotes the frequency of occurrence and $\tau_{\hat{k}}$ is chosen according to $\hat{k}(t_k) = \arg\min\{|t_k - \tau_j| \mid 0 \leq j \leq n\}$ for each $k = 1, \ldots, m$.

The Galerkin POD problem consists of finding a sequence $\{\tilde{\mathbf{u}}_k\}_{k=0}^m$ in H_l, which satisfies

$$(\tilde{\mathbf{u}}_0, \psi)_H = (\mathbf{u}_0, \psi)_H \quad \forall \, \psi \in H_l \tag{4.36}$$

and

$$(D_t^m \tilde{\mathbf{u}}_k, \psi)_H + \nu a(\tilde{\mathbf{u}}_k, \psi) + n(\tilde{\mathbf{u}}_k, \tilde{\mathbf{u}}_k, \psi) = (\mathbf{f}, \psi)_H \tag{4.37}$$

for all $\psi \in H_l$ and $k = 1, \dots, m$, where the operator D_t^m is defined by

$$D_t^m \tilde{\mathbf{u}}_k = \frac{\tilde{\mathbf{u}}_k - \tilde{\mathbf{u}}_{k-1}}{\triangle t_k}. \tag{4.38}$$

The following theorem, which is proved in [92], establishes existence results for the sequence $\{\tilde{\mathbf{u}}_k\}_{k=0}^m$.

Theorem 4.10 (Kunisch/Volkwein). *For every $k = 1, \dots, m$ there exists at least one solution $\tilde{\mathbf{u}}_k$ of (4.37), and if t_{max} is sufficiently small, the sequence $\{\tilde{\mathbf{u}}_k\}_{k=0}^m$ is uniquely determined.*

We now turn to the derivation of an error estimate for

$$\sum_{k=0}^{m} \beta_k \left\| \tilde{\mathbf{u}}_k - \mathbf{u}(t_k) \right\|_H^2, \tag{4.39}$$

where $\mathbf{u}(t_k)$ is the solution of (2.1) at the time instances $t_0, \dots, t_m$ and the weights β_k are chosen according to

$$\beta_0 = \frac{\triangle t_1}{2}, \quad \beta_k = \frac{\triangle t_{k+1} + \triangle t_k}{2}, \ k = 1, \dots, m-1, \text{ and } \beta_m = \frac{\triangle t_m}{2}. \tag{4.40}$$

We state the following error estimate for (4.39), which was proved in [92].

Theorem 4.11. *Assume that $\mathbf{u}_t \in L^2(0,T;V)$ and $\mathbf{u}_{tt} \in L^2(0,T;H)$ hold, and that t_{max} is sufficiently small. Then there exist constants*[2] *c_α and C, with C depending on T but independent of the grids $\{\tau_i\}_{i=0}^n$ and $\{t_i\}_{i=0}^m$, such that*

$$\begin{aligned}
&\sum_{k=0}^{m} \beta_k \left\| \tilde{\mathbf{u}}_k - \mathbf{u}(t_k) \right\|_H^2 \\
&\quad \le C \sum_{i=l+1}^{p} \left(|(\psi_i, \mathbf{u}_0)|_V^2 + \frac{\sigma_n}{\tau_{min}} \big(\frac{1}{t_{min}} + t_{max} \big) \lambda_i \right) + C \sigma_n t_{max} \tau_{max} \left\| \mathbf{u}_t \right\|_{L^2(0,T;V)}^2 \\
&\qquad + C \sigma_n (1 + c_\alpha^2) t_{max} \big(\tau_{max} \left\| \mathbf{u}_t \right\|_{L^2(0,T;H)}^2 + (t_{max} + \tau_{max}) \left\| \mathbf{u}_{tt} \right\|_{L^2(0,T;H)}^2 \big).
\end{aligned} \tag{4.41}$$

For the special case that the time grids coincide, that is, $n = m$ and $t_j = \tau_j$, $j = 1, \dots, m$, we have

$$\begin{aligned}
&\sum_{k=0}^{m} \beta_k \left\| \tilde{\mathbf{u}}_k - \mathbf{u}(t_k) \right\|_H^2 \\
&\quad \le C (1 + c_\alpha^2) t_{max}^2 \left\| \mathbf{u}_{tt} \right\|_{L^2(0,T;H)}^2 \\
&\qquad + C \left(\sum_{i=l+1}^{p} \big(|(\psi_i, u_0)_V|^2 + \big(\frac{1}{t_{min}^2} + 1 \big) \lambda_i \big) + t_{max}^2 \left\| \mathbf{u}_t \right\|_{L^2(0,T;V)}^2 \right).
\end{aligned} \tag{4.42}$$

[2] The constant c_α results essentially from the fact that $H_0^1(\Omega)$ is continuously embedded into $L^2(\Omega)$ (cf. [92]).

Remark 4.12. Note that if the number of POD elements used coincides with the dimension of $\mathcal{S}$, then (4.42) takes the form

$$\sum_{k=0}^{m} \beta_k \left\| \tilde{\mathbf{u}}_k - \mathbf{u}(t_k) \right\|_H^2 \leq C t_{\max}^2 \big((1 + c_\alpha^2) \left\| \mathbf{u}_{tt} \right\|_{L^2(0,T;H)}^2 + \left\| \mathbf{u}_t \right\|_{L^2(0,T;V)}^2 \big). \tag{4.43}$$

This eliminates the troublesome term $1/t_{\min}^2$, but does not allow truncation of the POD basis, which is one of the primary benefits of using the POD approach. Keeping this in mind, we will truncate the POD basis as described in Section 4.1.3, but for optimization purposes we will need some mechanism to ensure the fidelity of our model. This will be the subject of Chapter 6.

We note that Kunisch/Volkwein [92, Corollary 4.11] suggest that better convergence properties can be achieved for (4.39) by adding difference quotients to the snapshot set prior to calculating the POD basis (see also [73]). We have not yet experimented with this approach, but may do so in future work.

Remark 4.13. We note that it is also interesting to examine the effect the Reynolds number has on the error estimates of Theorem 4.11. The estimates (4.41) and (4.42) are derived in [93] for more general nonlinear evolution equations (see also [130]). Careful examination of this derivation shows that for Navier-Stokes problems the system viscosity ν gets "hidden" in the constant c_α, such that $c_\alpha = c_\beta/\nu$, where c_β is now independent of ν. Looking at (4.42), for instance, we see that any reduction in ν must be accompanied by a proportional reduction in $t_{\max}$ in order to maintain the order of the error estimate.

4.3 Numerical Analysis of POD Basis and Model Behavior

The theoretical results of Section 4.2 provide some indication that the Galerkin POD method based on finite element approximations at different discretization levels will converge satisfactorily for finer discretizations. Before proceeding, we wish to do some numerical testing of this assertion. For this purpose, we solved the driven cavity problem of Section 2.2.3 at Reynolds numbers of 100, 400, $10,000$ and $20,000$ for various temporal and spatial discretizations. The range of the spatial discretizations was the same for all Reynolds numbers, with simulations carried out on uniform meshes with fineness (coarseness) ranging from a very coarse 4×4 ($h = 0.3334$) mesh to a 97×97 ($h = 0.0104$) mesh. Since the dynamics of the problem vary considerably with changes in Reynolds numbers, we used a simulation time of $T = 5$ seconds at $Re = 100$ and $Re = 400$, and $T = 20$ seconds at $Re = 10,000$ and $Re = 20,000$. The temporal velocity profile used in all cases was $\gamma(t) \equiv 1.0$ (cf. (2.28)). The temporal discretizations used for generating the snapshot ensembles also varied according to Reynolds number and are described in more detail below.

It may be noted that a POD basis can be computed for either the entire velocity field spanned by the snapshots $\mathbf{u}(\tau_1), \ldots, \mathbf{u}(\tau_n)$, or for its fluctuating part

$$\mathbf{u}(\tau_1) - \mathbf{u}_n, \ldots, \mathbf{u}(\tau_n) - \mathbf{u}_n, \tag{4.44}$$

where $\mathbf{u}_n = (1/n) \sum_{i=1}^{n} \mathbf{u}(\tau_i)$. In fact, it is common in the literature to use only the fluctuating portion of the velocity field in the construction of the POD basis, and we will follow this tradition in later sections; however, since the average velocity field $\mathbf{u}_n$ depends on the discretization level, using the set (4.44) would make the POD bases computed from different discretization levels incomparable. For this reason, we have based the POD computations in this section on the entire velocity field.

4.3.1 Dependence of the Eigenvalues on the Snapshot Density

We begin by studying the dependence of the eigenvalues on the snapshot density. For each value of the Reynolds number we fix the spatial discretization (using the 97×97 (h=0.0104) mesh) and compare the behavior of the first five eigenvalues of $\mathcal{K}_n$ for uniformly distributed snapshot ensembles. Simulation time at $Re = 100$ and $Re = 400$ was 5 seconds, with snapshot density ranging from $n = 4$ to $n = 536$ for $Re = 100$, and $n = 6$ to $n = 815$ for $Re = 400$. Simulation time at $Re = 10,000$ and $Re = 20,000$ was 20 seconds, with snapshot density ranging from $n = 4$ to $n = 600$.

Table 4.1 on Page 62 displays the values of the first five eigenvalues $\lambda_1, \ldots, \lambda_5$ for $Re = 100$. Convergence is rapid; indeed, assuming that the values for $n = 536$ are very close to the correct values, $n = 4$ snapshots already appears to be sufficient for good approximation of the first eigenvalue λ_1, and $n = 134$ yields a decently small relative error across the table. The situation is displayed graphically in Figure 4.1 for the first three (left) and five (right) eigenvalues with λ_1 scaled to improve the visual display.

Figures 4.2 to 4.4 and Tables 4.3, 4.5 and 4.7 on Pages 63 to 65 contain similar analysis for Reynolds numbers 400, 10,000 and 20,000, respectively, with results similar to those for $Re = 100$. It is interesting to note that the number of snapshots needed for eigenvalue convergence remains on the order of about 100, even for higher Reynolds numbers with longer simulation times. This is no doubt due in part to the well-behaved boundary control used here, but it is simple enough to add additional snapshots should it become necessary to do so. In any event, from a numerical point of view, the method of snapshots gives us considerable flexibility as to the choice of n.

4.3.2 Dependence of the Eigenvalues on the Spatial Discretization

Moving to the effect of the spacial discretization, the theoretical results of Section 4.2.2 are confirmed by Figures 4.5, 4.6, 4.7 and 4.8 on Pages 66 to 69, where the values of $\|\mathcal{K}(h)\|$ and $\|\mathcal{K}(h) - \mathcal{K}(0)\|$ are plotted against h for $Re = 100$ to $Re = 20,000$. Note that we have assumed $\mathcal{K}(h = 0.0104)$ for $\mathcal{K}(0)$. We have also included some additional curves generated at $Re = 100$ and $Re = 400$ with streamline diffusion used in the numerical solution process. The addition of streamline diffusion appears to cause an increase in $\|\mathcal{K}(h)\|$ with the effect more pronounced at $Re = 400$.

Tables 4.9, 4.11, 4.13 and 4.15 display the dependence of the eigenvalues and eigenvectors on the spatial discretization. The results, which were generated using the finest temporal discretization for each Reynolds number, are encouraging. The largest eigenvalues converge rapidly in all cases, though λ_4 and λ_5 are still somewhat unsettled on the 97×97 mesh at $Re = 20,000$.

Eigenvector Condition

Tables 4.2, 4.4, 4.6 and 4.8 on Pages 62 to 65, respectively, display the dependence on the snapshot density of the condition (calculated according to (4.17)) of the eigenvectors $\psi_1, \ldots, \psi_5$ corresponding to $\lambda_1, \ldots, \lambda_5$. As one would expect, the dependence of the eigenvector condition on snapshot density mirrors the behavior of the corresponding eigenvalues. As the eigenvalues become smaller from left to right, the condition of the eigenvectors deteriorates. As seen in Tables 4.1, 4.3, 4.5 and 4.7, the eigenvalues generally become somewhat larger with increasing snapshot density; this is reflected in the condition of the corresponding eigenvectors, where the condition improves with increasing snapshot density.

The results of Tables 4.10, 4.12, 4.14 and 4.16 are slightly more interesting. Since the eigenvalues generally become smaller with increasing mesh refinement (Tables 4.9, 4.11, 4.13 and 4.15), the condition of the corresponding eigenvectors deteriorates with finer meshes; that is, better finite element approximations.

4.3.3 Convergence of the POD Basis

It is common in the literature to use quiver diagrams for the display of POD basis vectors, and we follow this custom to some extent; however, it would be difficult to judge the convergence of the POD bases using such displays. For this reason, we concentrate on the velocity norms along a diagonal running from the lower left corner ($x_1 = x_2 = 0.0$) of the driven cavity to the upper right corner ($x_1 = x_2 = 1.0$). Consider for example Figure 4.9 on Page 70 reporting results for $Re = 100$. The quiver diagrams for the first three POD basis vectors – generated on the 49×49 mesh – are provided on the left for reference, while the velocity norms on the diagonal are displayed on the right for the 13×13, 25×25, 49×49 and 97×97 meshes. The nice behavior of the POD basis is obvious at $Re = 100$, the results for $Re = 400$ being nearly as good in Figure 4.10.

We see considerably slower convergence at $Re = 10,000$ in Figure 4.11 on Page 72, though the curves for the 49×49 and 97×97 meshes bear decent resemblance. Similar results hold in Figure 4.12 at $Re = 20,000$.

The slower convergence of the POD basis functions at higher Reynolds numbers is no cause for alarm, as even the high-order solver is bound to require more degrees-of-freedom to capture the essential system character at higher Reynolds numbers. Nevertheless, if we wish to use rougher discretizations in a recursive multilevel optimization scheme using POD-based models, then these results underscore the need for some type of mechanism to guide the process and ensure that we can "trust" our model.

4.3.4 Influence of Streamline Diffusion on the POD Basis

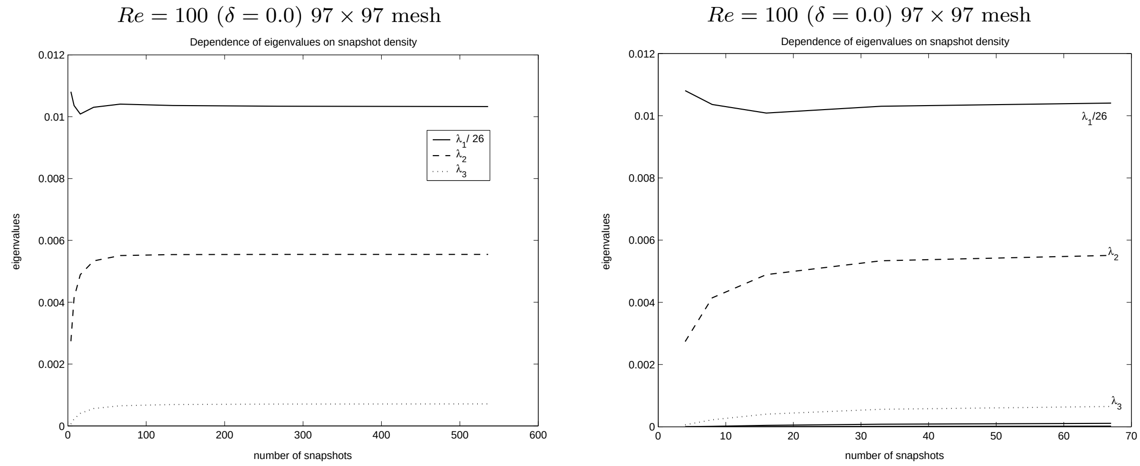
In Tables 4.17 and 4.18 on Page 74, we evaluate the behavior of the eigenvalues in the presence of streamline diffusion at $Re = 100$ and $Re = 400$ using the 97×97 mesh. For each Reynolds number we generated three numerical solutions of the driven cavity problem with $\gamma(t) \equiv 1.0$ and the streamline diffusion parameter δ set to 0.0, 0.5 and 1.0. Note that it was not necessary to use streamline diffusion to obtain a solution at these Reynolds numbers, but we hope that by adding stabilization at lower Reynolds numbers we can gain some insight into the effect of streamline diffusion on the POD data at higher Reynolds numbers.

As can be seen in Table 4.17, at $Re = 100$ the addition of streamline diffusion to the high-order solution process has little effect on the eigenvalues of the correlation matrix. The first three eigenvalues capture 99.9% of the system energy at all three values of δ.

The effect is somewhat more noticeable at $Re = 400$ in Table 4.18. Five eigenvalues are now needed to capture 99.9% of the system energy, and we see clearly that the largest eigenvalues decrease in value with increases in δ.

Table 4.19 on Page 75 lists some eigenvalues for $Re = 10,000$, with snapshots generated from simulations using various amounts of streamline diffusion: $\delta \in \{0.2, 0.5, 1.0, 2.0\}$. The value $\delta = 0.0$ is not included because the corresponding high-order solution process did not converge without stabilization.

The character of the eigenvalue distribution in the columns of Table 4.19 is considerably different from that of Table 4.18. We now need about 20 eigenvalues to capture 99.9% of the system energy. Across the table, however, we see that fewer eigenvalues are required to capture a given percentage of the system energy at higher values of δ – as was the case for the lower Reynolds numbers. The results for $Re = 20,000$ in Table 4.20 are similar to the results for $Re = 10,000$.



$Re = 100$ ($\delta = 0.0$) 97×97 mesh

$Re = 100$ ($\delta = 0.0$) 97×97 mesh

Figure 4.1: First three eigenvalues plotted against the snapshot density (left). The first eigenvalue was scaled to enhance visual comparability. Closeup for the first five eigenvalues (right).

Eigenvalues: $Re = 100$ on a 97×97 mesh						
n	$\triangle\tau$	λ_1	λ_2	λ_3	λ_4	λ_5
4	1.19e-0	2.8097e-1	2.7371e-3	6.0085e-5	4.6816e-7	0
8	5.97e-1	2.6939e-1	4.1447e-3	2.2225e-4	1.3276e-5	3.5281e-7
16	2.98e-1	2.6221e-1	4.8895e-3	4.0386e-4	4.3825e-5	3.2535e-6
33	1.49e-1	2.6786e-1	5.3366e-3	5.6121e-4	8.0612e-5	1.0276e-5
67	7.46e-2	2.7052e-1	5.5101e-3	6.5248e-4	1.1070e-4	1.9273e-5
134	3.73e-2	2.6937e-1	5.5392e-3	6.9067e-4	1.2923e-4	2.7157e-5
268	1.86e-2	2.6877e-1	5.5449e-3	7.0527e-4	1.3903e-4	3.2724e-5
536	9.32e-3	2.6847e-1	5.5447e-3	7.0999e-4	1.4323e-4	3.5888e-5

Table 4.1: Dependence of the first five eigenvalues on the number of snapshots.

Eigenvector condition: $Re = 100$ on a 97×97 mesh						
n	$\triangle\tau$	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5
4	1.19e-0	3.5940	3.7353e+2	1.6773e+4	2.1360e+6	∞
8	5.97e-1	3.7700	2.5494e+2	4.7851e+3	7.7378e+4	2.8926e+6
16	2.98e-1	3.8860	2.2292e+2	2.7774e+3	2.4647e+4	3.3119e+5
33	1.49e-1	3.8091	2.0940e+2	2.0807e+3	1.4217e+4	1.0923e+5
67	7.46e-2	3.7733	2.0585e+2	1.8457e+3	1.0936e+4	6.0715e+4
134	3.73e-2	3.7903	2.0624e+2	1.7811e+3	9.7963e+3	4.5062e+4
268	1.86e-2	3.7988	2.0662e+2	1.7660e+3	9.4066e+3	3.9019e+4
536	9.32e-3	3.8032	2.0683e+2	1.7644e+3	9.3154e+3	3.6855e+4

Table 4.2: Dependence of eigenvector condition on snapshot density.

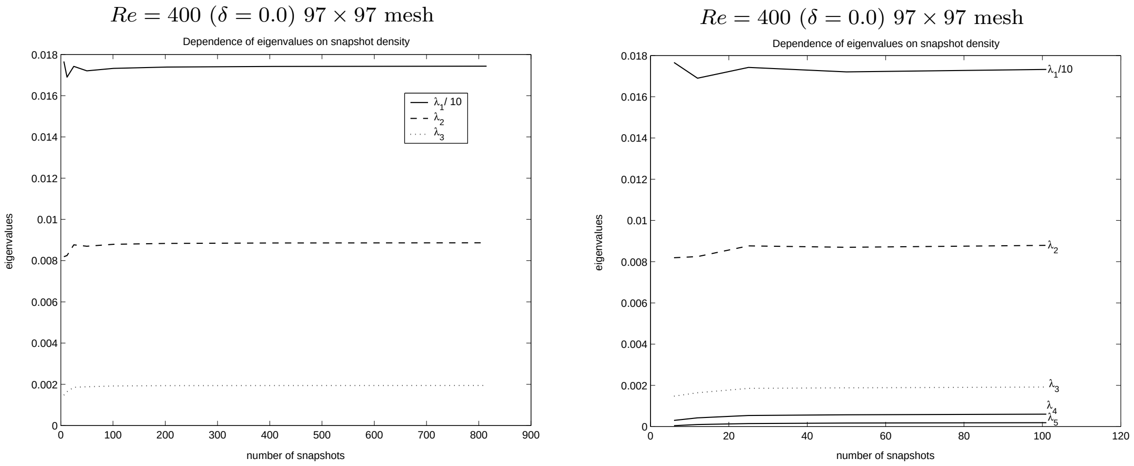


Figure 4.2: First three eigenvalues plotted against the snapshot density (left). The first eigenvalue was scaled to enhance visual comparability. Closeup for the first five eigenvalues (right).

Eigenvalues: $Re = 400$ on a 97 × 97 mesh						
n	$\triangle\tau$	λ_1	λ_2	λ_3	λ_4	λ_5
6	7.85e-1	1.7663e-1	8.1930e-3	1.4760e-3	3.0136e-4	4.3156e-5
12	3.92e-1	1.6898e-1	8.2472e-3	1.6426e-3	4.2448e-4	9.9954e-5
25	1.96e-1	1.7424e-1	8.7656e-3	1.8527e-3	5.3831e-4	1.4824e-4
50	9.82e-2	1.7204e-1	8.6970e-3	1.8799e-3	5.7487e-4	1.7188e-4
101	4.91e-2	1.7326e-1	8.7923e-3	1.9186e-3	6.0201e-4	1.8909e-4
203	2.45e-2	1.7387e-1	8.8352e-3	1.9342e-3	6.1314e-4	1.9729e-4
407	1.23e-2	1.7416e-1	8.8553e-3	1.9406e-3	6.1749e-4	2.0077e-4
815	1.80e-3	1.7431e-1	8.8650e-3	1.9435e-3	6.1927e-4	2.0224e-4

Table 4.3: The dependence of the first five eigenvalues on the number of snapshots.

Eigenvector condition: $Re = 400$ on a 97 × 97 mesh						
n	$\triangle\tau$	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5
6	7.85e-1	5.9368	1.4887e+2	8.5129e+2	3.8728e+3	2.4949e+4
12	3.92e-1	6.2212	1.5141e+2	8.2088e+2	3.0813e+3	1.3214e+4
25	1.96e-1	6.0431	1.4465e+2	7.6076e+2	2.5635e+3	9.5022e+3
50	9.82e-2	6.1218	1.4668e+2	7.6625e+2	2.4814e+3	8.4810e+3
101	4.91e-2	6.0799	1.4548e+2	7.5952e+2	2.4217e+3	7.9299e+3
203	2.45e-2	6.0593	1.4490e+2	7.5695e+2	2.4047e+3	7.7464e+3
407	1.23e-2	6.0491	1.4461e+2	7.5575e+2	2.3996e+3	7.6956e+3
815	1.80e-3	6.0440	1.4447e+2	7.5513e+2	2.3978e+3	7.6841e+3

Table 4.4: Dependence of eigenvector condition on snapshot density.

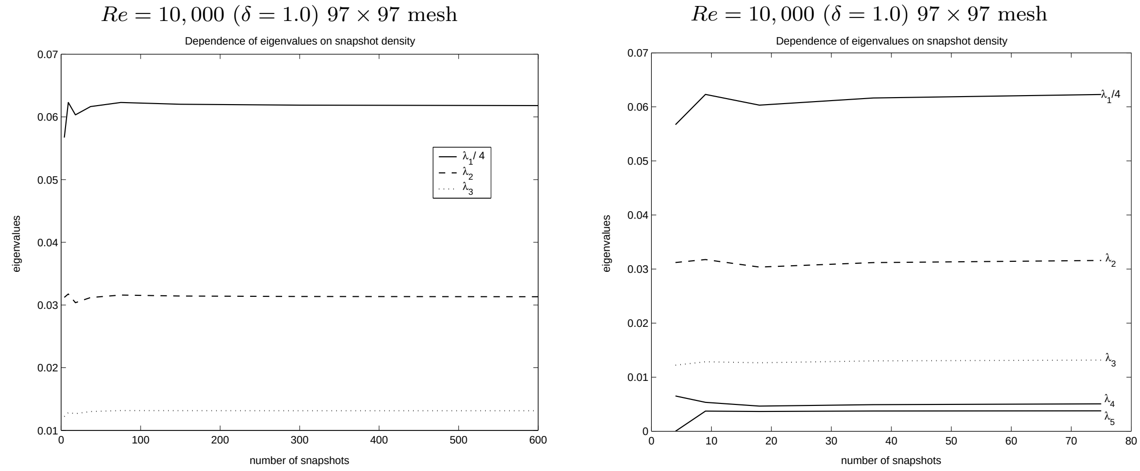


Figure 4.3: First three eigenvalues plotted against the snapshot density (left). The first eigenvalue was scaled to enhance visual comparability. Closeup for the first five eigenvalues (right).

Eigenvalues: $Re = 10,000$ on a 97×97 mesh						
n	$\triangle\tau$	λ_1	λ_2	λ_3	λ_4	λ_5
4	4.26e-0	2.2683e-1	3.1206e-2	1.2204e-2	6.5189e-3	0
9	2.13e-0	2.4920e-1	3.1751e-2	1.2835e-2	5.3429e-3	3.7146e-3
18	1.07e-0	2.4129e-1	3.0357e-2	1.2655e-2	4.6461e-3	3.6597e-3
37	5.33e-1	2.4657e-1	3.1188e-2	1.3002e-2	4.9124e-3	3.7334e-3
75	2.67e-1	2.4917e-1	3.1585e-2	1.3157e-2	5.0561e-3	3.7696e-3
150	1.33e-1	2.4806e-1	3.1428e-2	1.3142e-2	4.9929e-3	3.7862e-3
300	6.67e-2	2.4751e-1	3.1348e-2	1.3133e-2	4.9615e-3	3.7941e-3
600	3.33e-2	2.4723e-1	3.1308e-2	1.3128e-2	4.9458e-3	3.7980e-3

Table 4.5: The dependence of the first five eigenvalues on the number of snapshots.

Eigenvector condition: $Re = 10,000$ on a 97×97 mesh						
n	$\triangle\tau$	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5
4	4.26e-0	5.1117	5.2626e+1	1.7587e+2	1.7587e+2	∞
9	2.13e-0	4.5986	5.2865e+1	1.3347e+2	6.1416e+2	1.0139e+3
18	1.07e-0	4.7408	5.6490e+1	1.2485e+2	1.0138e+3	1.0138e+3
37	5.33e-1	4.6428	5.4987e+1	1.2361e+2	8.4815e+2	8.4815e+2
75	2.67e-1	4.5957	5.4263e+1	1.2343e+2	7.7734e+2	8.3301e+2
150	1.33e-1	4.6159	5.4685e+1	1.2270e+2	8.2870e+2	8.2870e+2
300	6.67e-2	4.6261	5.4899e+1	1.2236e+2	8.5662e+2	8.5662e+2
600	3.33e-2	4.6312	5.5006e+1	1.2220e+2	8.7120e+2	8.7120e+2

Table 4.6: Dependence of eigenvector condition on snapshot density.

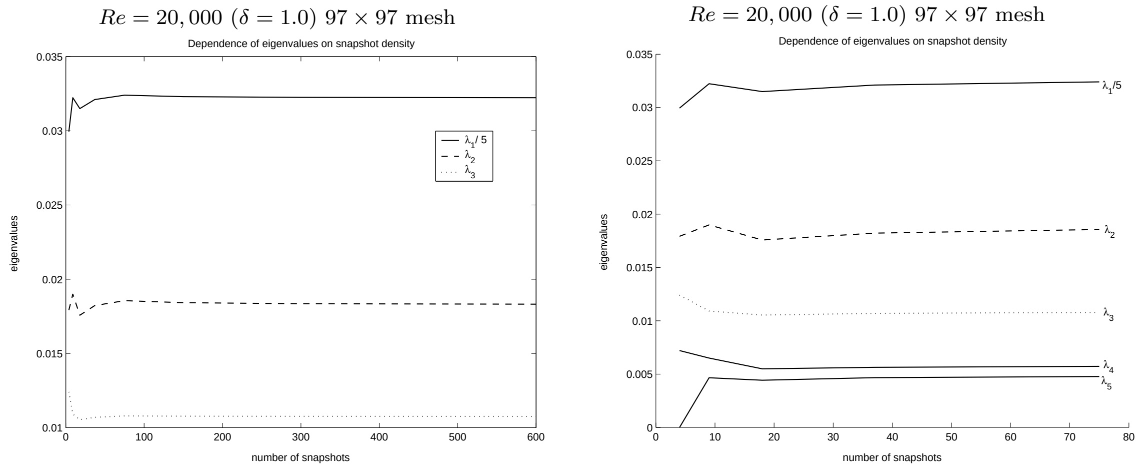


Figure 4.4: First three eigenvalues plotted against the snapshot density (left). The first eigenvalue was scaled to enhance visual comparability. Closeup for the first five eigenvalues (right).

Eigenvalues: $Re = 20,000$ on a 97×97 mesh						
n	$\triangle\tau$	λ_1	λ_2	λ_3	λ_4	λ_5
4	4.26e-0	1.4972e-1	1.7922e-2	1.2409e-2	7.2100e-3	0
9	2.13e-0	1.6113e-1	1.8992e-2	1.0914e-2	6.5076e-3	4.6565e-3
18	1.07e-0	1.5747e-1	1.7572e-2	1.0533e-2	5.4926e-3	4.4281e-3
37	5.33e-1	1.6051e-1	1.8222e-2	1.0696e-2	5.6329e-3	4.6682e-3
75	2.67e-1	1.6200e-1	1.8556e-2	1.0787e-2	5.7275e-3	4.7744e-3
150	1.33e-1	1.6151e-1	1.8419e-2	1.0769e-2	5.6776e-3	4.7667e-3
300	6.67e-2	1.6126e-1	1.8350e-2	1.0760e-2	5.6532e-3	4.7625e-3
600	3.33e-2	1.6113e-1	1.8316e-2	1.0755e-2	5.6412e-3	4.7603e-3

Table 4.7: The dependence of the first five eigenvalues on the number of snapshots.

Eigenvector condition: $Re = 20,000$ on a 97×97 mesh						
n	$\triangle\tau$	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5
4	4.26e-0	7.5872	1.8137e+2	1.9233e+2	1.9233e+2	∞
9	2.13e-0	7.0350	1.2379e+2	2.2693e+2	5.4022e+2	5.4022e+2
18	1.07e-0	7.1479	1.4206e+2	1.9838e+2	9.3937e+2	9.3937e+2
37	5.33e-1	7.0276	1.3288e+2	1.9748e+2	1.0366e+3	1.0366e+3
75	2.67e-1	6.9713	1.2871e+2	1.9761e+2	1.0491e+3	1.0491e+3
150	1.33e-1	6.9885	1.3072e+2	1.9638e+2	1.0978e+3	1.0978e+3
300	6.67e-2	6.9973	1.3174e+2	1.9580e+2	1.1226e+3	1.1226e+3
600	3.33e-2	7.0017	1.3225e+2	1.9552e+2	1.1350e+3	1.1350e+3

Table 4.8: Dependence of eigenvector condition on snapshot density.

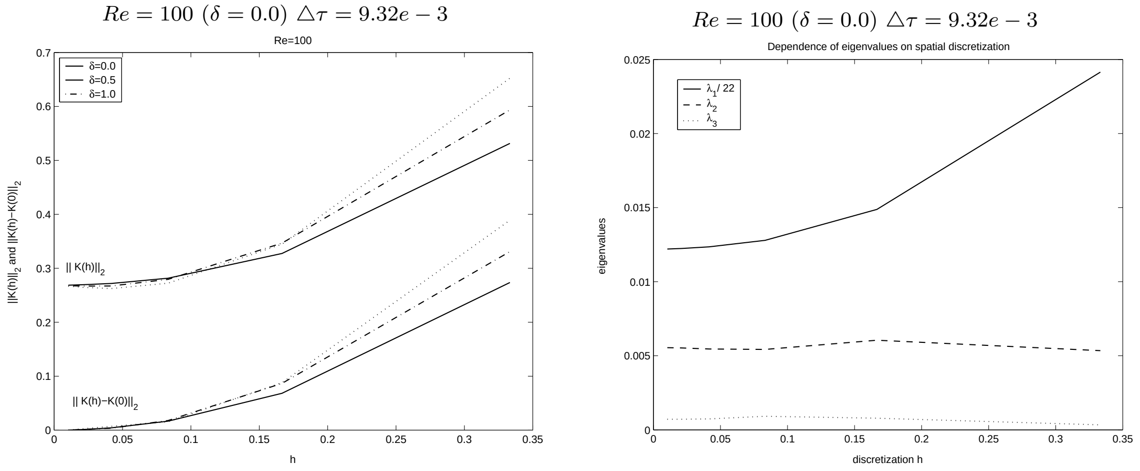


Figure 4.5: The dependence of $\|\mathcal{K}(h)\|_2$ on the spatial discretization (left). The dependence of the first three eigenvalues on the spatial discretization (right). The first eigenvalue was scaled to enhance visual comparability.

Eigenvalues: $Re = 100$ with $\triangle\tau = 9.32e - 3$						
Discr.	h	λ_1	λ_2	λ_3	λ_4	λ_5
4×4	0.3333	5.3120e-1	5.3446e-3	3.3574e-4	6.2351e-6	2.6919e-7
7×7	0.1667	3.2724e-1	6.0473e-3	7.8542e-4	6.1517e-5	4.9349e-6
13×13	0.0833	2.8141e-1	5.4279e-3	9.1214e-4	1.6466e-4	2.8621e-5
25×25	0.0417	2.7173e-1	5.4549e-3	7.3788e-4	1.6146e-4	4.4240e-5
49×49	0.0208	2.6933e-1	5.5273e-3	7.1332e-4	1.4476e-4	3.6911e-5
97×97	0.0104	2.6847e-1	5.5447e-3	7.0999e-4	1.4323e-4	3.5888e-5

Table 4.9: The dependence of the eigenvalues on the underlying spatial discretization ($Re = 100$; $n = 536$; $\triangle\tau = 9.32e - 3$; $T = 5$).

Eigenvector condition: $Re = 100$ with $\triangle\tau = 9.32e - 3$						
Discr.	h	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5
4×4	0.3333	1.9016	1.9964e+2	3.0348e+3	1.6761e+5	3.8954e+6
7×7	0.1667	3.1133	1.9004e+2	1.3813e+3	1.7673e+4	2.2029e+5
13×13	0.0833	3.6233	2.2144e+2	1.3378e+3	7.3505e+3	4.0403e+4
25×25	0.0417	3.7555	2.1199e+2	1.7348e+3	8.5304e+3	2.9348e+4
49×49	0.0208	3.7906	2.0772e+2	1.7588e+3	9.2716e+3	3.6184e+4
97×97	0.0104	3.8032	2.0683e+2	1.7644e+3	9.3154e+3	3.6855e+4

Table 4.10: The dependence of eigenvector condition on the spatial discretization.

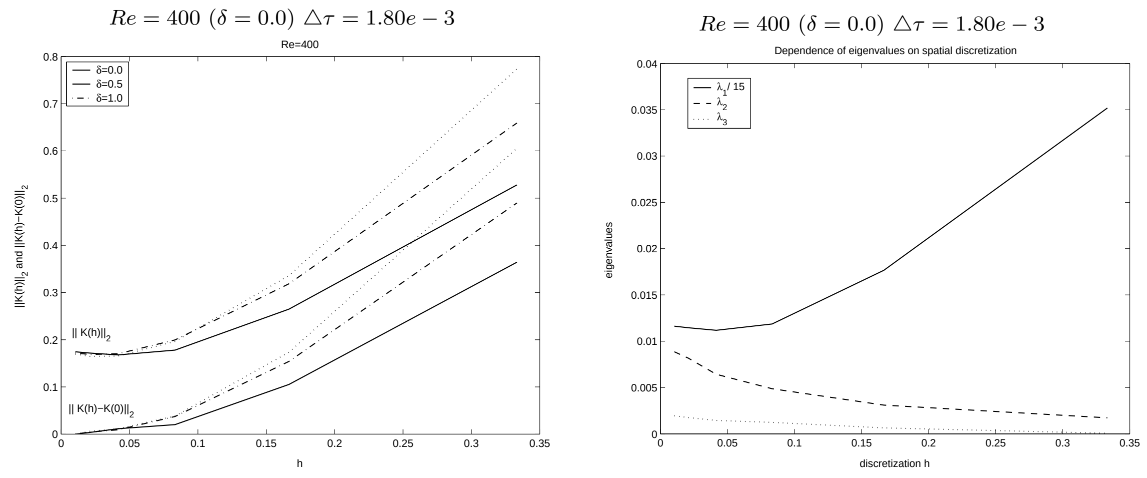


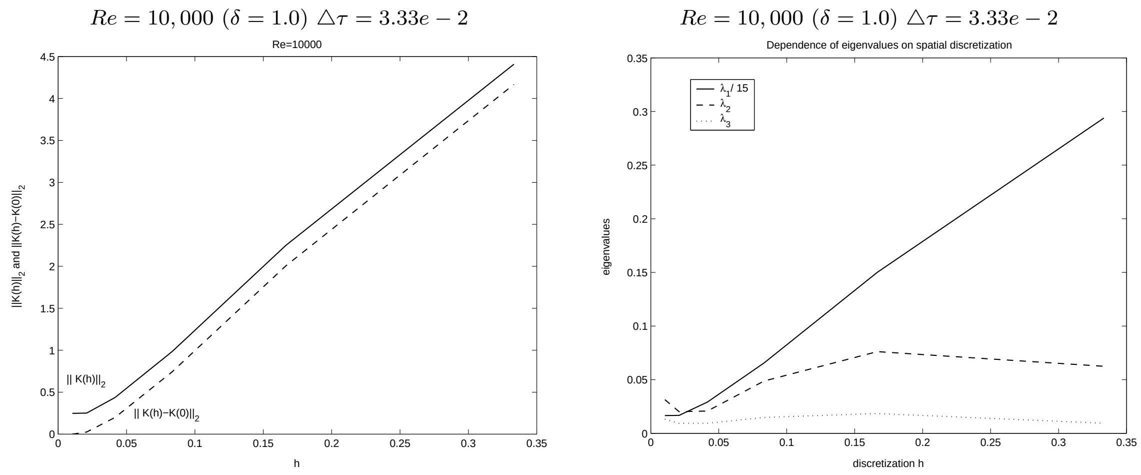
Figure 4.6: The dependence of $\|\mathcal{K}(h)\|_2$ on the spatial discretization (left). The dependence of the first three eigenvalues on the spatial discretization (right). The first eigenvalue was scaled to enhance visual comparability.

Eigenvalues: $Re = 400$ with $\triangle\tau = 1.80e - 3$						
Discr.	h	λ_1	λ_2	λ_3	λ_4	λ_5
4×4	0.3333	5.2783e-1	1.7237e-3	6.3999e-5	1.0067e-6	3.4037e-8
7×7	0.1667	2.6487e-1	3.0994e-3	6.3881e-4	2.9402e-5	2.0058e-6
13×13	0.0833	1.7799e-1	4.8720e-3	1.2378e-3	2.9308e-4	5.9545e-5
25×25	0.0417	1.6771e-1	6.4319e-3	1.4393e-3	4.7813e-4	1.5707e-4
49×49	0.0208	1.7178e-1	8.1731e-3	1.7578e-3	5.6094e-4	1.8632e-4
97×97	0.0104	1.7431e-1	8.8650e-3	1.9435e-3	6.1927e-4	2.0224e-4

Table 4.11: The dependence of the eigenvalues on the underlying spatial discretization ($Re = 400$; $n = 815$; $\triangle\tau = 1.80e - 3$; $T = 5$).

Eigenvector condition: $Re = 400$ with $\triangle\tau = 1.80e - 3$						
Discr.	h	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5
4×4	0.3333	1.9007	60250e+2	1.5874e+4	1.0280e+6	2.9805e+7
7×7	0.1667	3.8200	40639e+2	1.6409e+3	3.6500e+4	5.4229e+5
13×13	0.0833	5.7763	27516e+2	1.0584e+3	4.2820e+3	2.0418e+4
25×25	0.0417	6.2003	20029e+2	1.0403e+3	3.1147e+3	8.9542e+3
49×49	0.0208	6.1118	15587e+2	8.3548e+2	2.6693e+3	8.5078e+3
97×97	0.0104	6.0440	14447e+2	7.5513e+2	2.3978e+3	7.6841e+3

Table 4.12: The dependence of eigenvector condition on the spatial discretization.


$Re = 10,000$ ($\delta = 1.0$) $\triangle\tau = 3.33e-2$

$Re = 10,000$ ($\delta = 1.0$) $\triangle\tau = 3.33e-2$

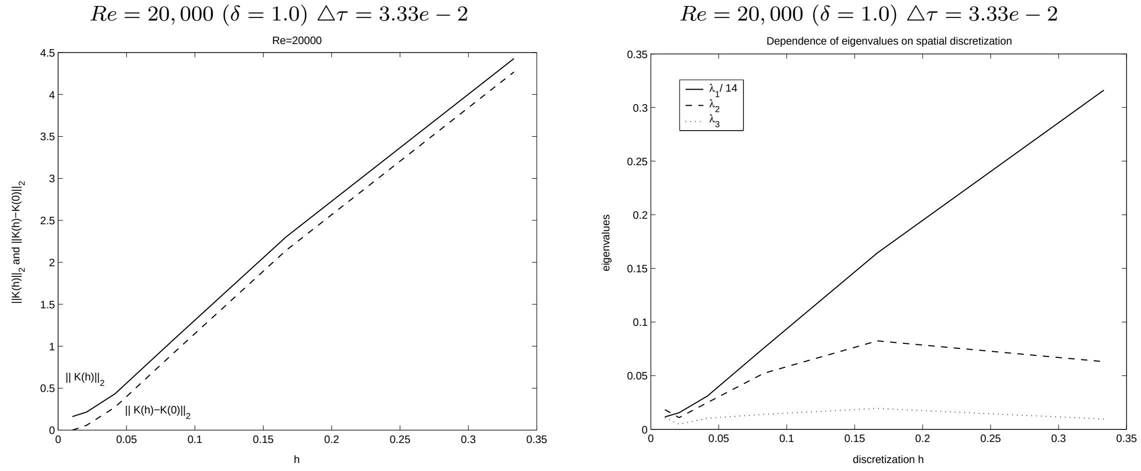
Figure 4.7: The dependence of $\|\mathcal{K}(h)\|_2$ on the spatial discretization (left). The dependence of the first three eigenvalues on the spatial discretization (right). The first eigenvalue was scaled to enhance visual comparability.

Eigenvalues: $Re = 10,000$ with $\triangle\tau = 3.33e-2$						
Discr.	h	λ_1	λ_2	λ_3	λ_4	λ_5
4×4	0.3333	4.4084e-0	6.2470e-2	9.4315e-3	2.6997e-3	3.6433e-4
7×7	0.1667	2.2509e-0	7.6103e-2	1.8461e-2	6.9158e-3	2.0123e-3
13×13	0.0833	9.8323e-1	4.8692e-2	1.4811e-2	4.6239e-3	1.7416e-3
25×25	0.0417	4.3699e-1	2.0706e-2	9.3454e-3	3.8202e-3	2.1111e-3
49×49	0.0208	2.5047e-1	2.0159e-2	9.3025e-3	3.9830e-3	2.7625e-3
97×97	0.0104	2.4723e-1	3.1308e-2	1.3128e-2	4.9458e-3	3.7980e-3

Table 4.13: The dependence of the eigenvalues on the underlying spatial discretization ($Re = 10,000$; $n = 600$; $\triangle\tau = 3.33e-2$; $T = 20$).

Eigenvector condition: $Re = 10,000$ with $\triangle\tau = 3.33e-2$						
Discr.	h	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5
4×4	0.3333	0.2301	1.8853e+1	1.4854e+2	4.2819e+2	3.3027e+3
7×7	0.1667	0.4597	1.7348e+1	8.6614e+1	2.0393e+2	1.4216e+3
13×13	0.0833	1.0700	2.9515e+1	9.8158e+1	3.4695e+2	1.1494e+3
25×25	0.0417	2.4021	8.8020e+1	1.8098e+2	5.8509e+2	1.0764e+3
49×49	0.0208	4.3418	9.2111e+1	1.8798e+2	8.1934e+2	8.1934e+2
97×97	0.0104	4.6312	5.5006e+1	1.2220e+2	8.7120e+2	8.7120e+2

Table 4.14: The dependence of eigenvector condition on the spatial discretization.



$Re = 20,000$ $(\delta = 1.0)$ $\triangle\tau = 3.33e - 2$ $\qquad$ $Re = 20,000$ $(\delta = 1.0)$ $\triangle\tau = 3.33e - 2$

Figure 4.8: The dependence of $\|\mathcal{K}(h)\|_2$ on the spatial discretization (left). The dependence of the first three eigenvalues on the spatial discretization (right). The first eigenvalue was scaled to enhance visual comparability.

Eigenvalues: $Re = 20,000$ with $\triangle\tau = 3.33e - 2$						
Discr.	h	λ_1	λ_2	λ_3	λ_4	λ_5
4×4	0.3333	4.4267e-0	6.3045e-2	9.5293e-3	2.7142e-3	3.6614e-4
7×7	0.1667	2.3021e-0	8.2415e-2	1.9453e-2	7.4972e-3	2.1164e-3
13×13	0.0833	1.0620e-0	5.2310e-2	1.3947e-2	5.2133e-3	1.4197e-3
25×25	0.0417	4.3365e-1	2.4901e-2	1.0272e-2	4.8256e-3	2.7500e-3
49×49	0.0208	2.1597e-1	1.0904e-2	4.7713e-3	1.8123e-3	1.0199e-3
97×97	0.0104	1.6113e-1	1.8316e-2	1.0755e-2	5.6412e-3	4.7603e-3

Table 4.15: The dependence of the eigenvalues on the underlying spatial discretization ($Re = 20,000$; $n = 600$; $\triangle\tau = 3.33e - 2$; $T = 20$).

Eigenvector condition: $Re = 20,000$ with $\triangle\tau = 3.33e - 2$						
Discr.	h	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5
4×4	0.3333	0.2301	1.8853e+1	1.4854e+2	4.2819e+2	3.3027e+3
7×7	0.1667	0.4597	1.7348e+1	8.6614e+1	2.0393e+2	1.4216e+3
13×13	0.0833	1.0700	2.9515e+1	9.8158e+1	3.4695e+2	1.1494e+3
25×25	0.0417	2.4021	8.8020e+1	1.8098e+2	5.8509e+2	1.0764e+3
49×49	0.0208	4.3418	9.2111e+1	1.8798e+2	8.1934e+2	8.1934e+2
97×97	0.0104	4.6312	5.5006e+1	1.2220e+2	8.7120e+2	8.7120e+2

Table 4.16: The dependence of eigenvector condition on the spatial discretization.

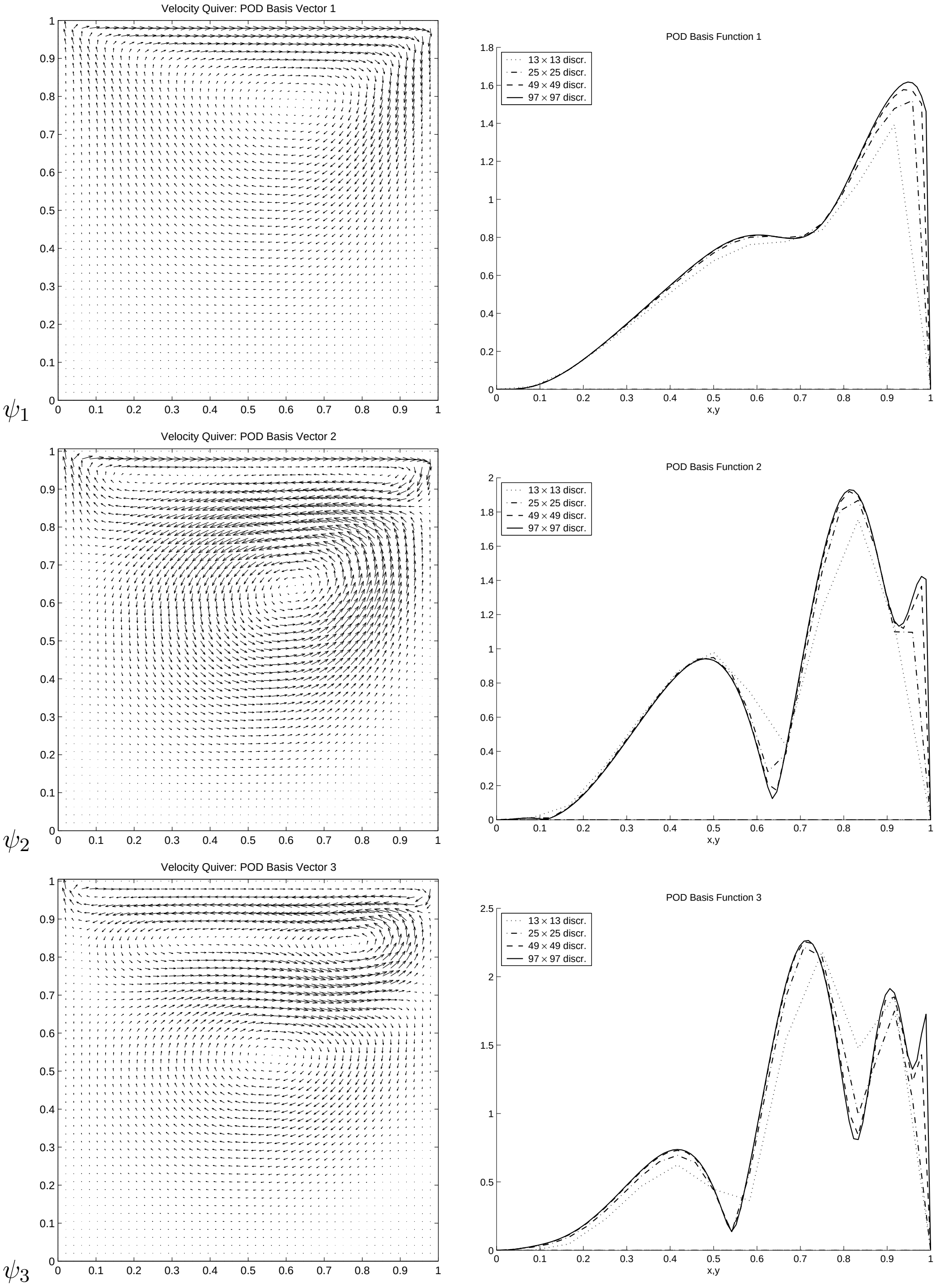


Figure 4.9: The dependence of the first three POD basis functions on the underlying spatial discretization at $Re = 100$. On the left, the velocity quiver diagram for each POD basis function taken on the 49×49 mesh. On the right, the velocity norms at various discretization levels on the mesh diagonal running from the lower left to the upper right of the cavity.

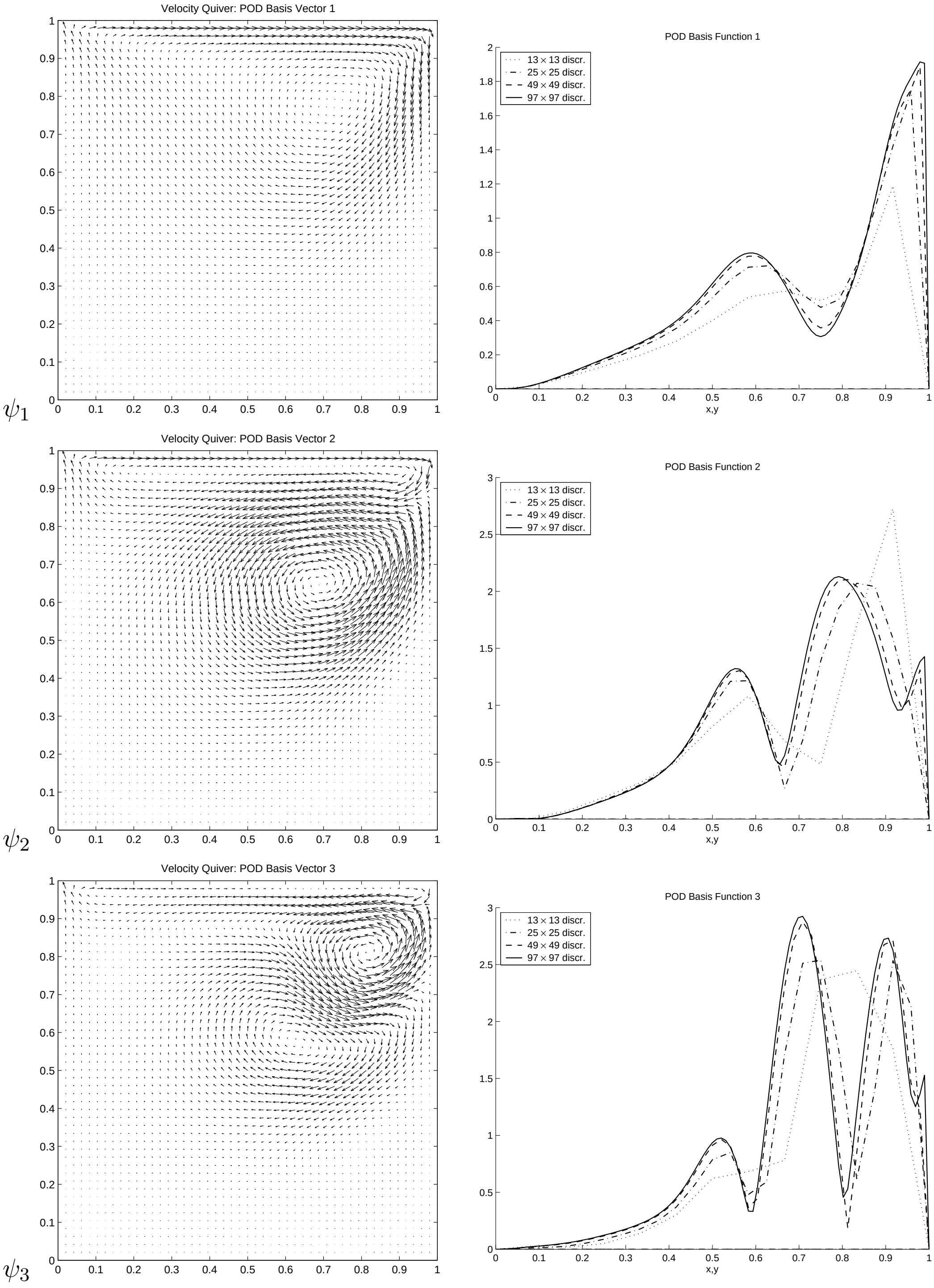


Figure 4.10: The dependence of the first three POD basis functions on the underlying spatial discretization at $Re = 400$. On the left, the velocity quiver diagram for each POD basis function taken on the 49×49 mesh. On the right, the velocity norms at various discretization levels on the mesh diagonal running from the lower left to the upper right of the cavity.

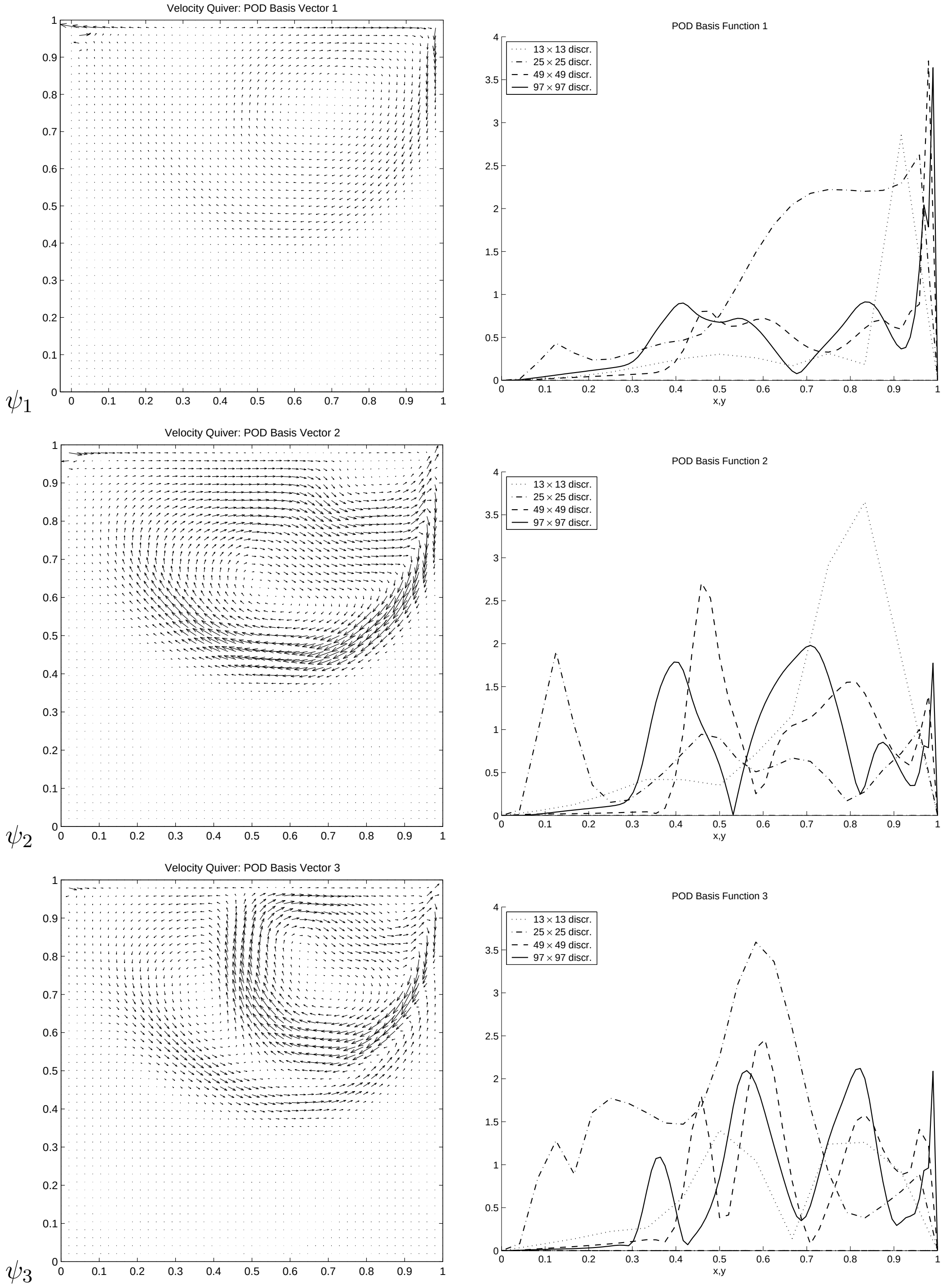


Figure 4.11: The dependence of the first three POD basis functions on the underlying spatial discretization at $Re = 10,000$. On the left, the velocity quiver diagram for each POD basis function taken on the 49×49 mesh. On the right, the velocity norms at various discretization levels on the mesh diagonal running from the lower left to the upper right of the cavity.

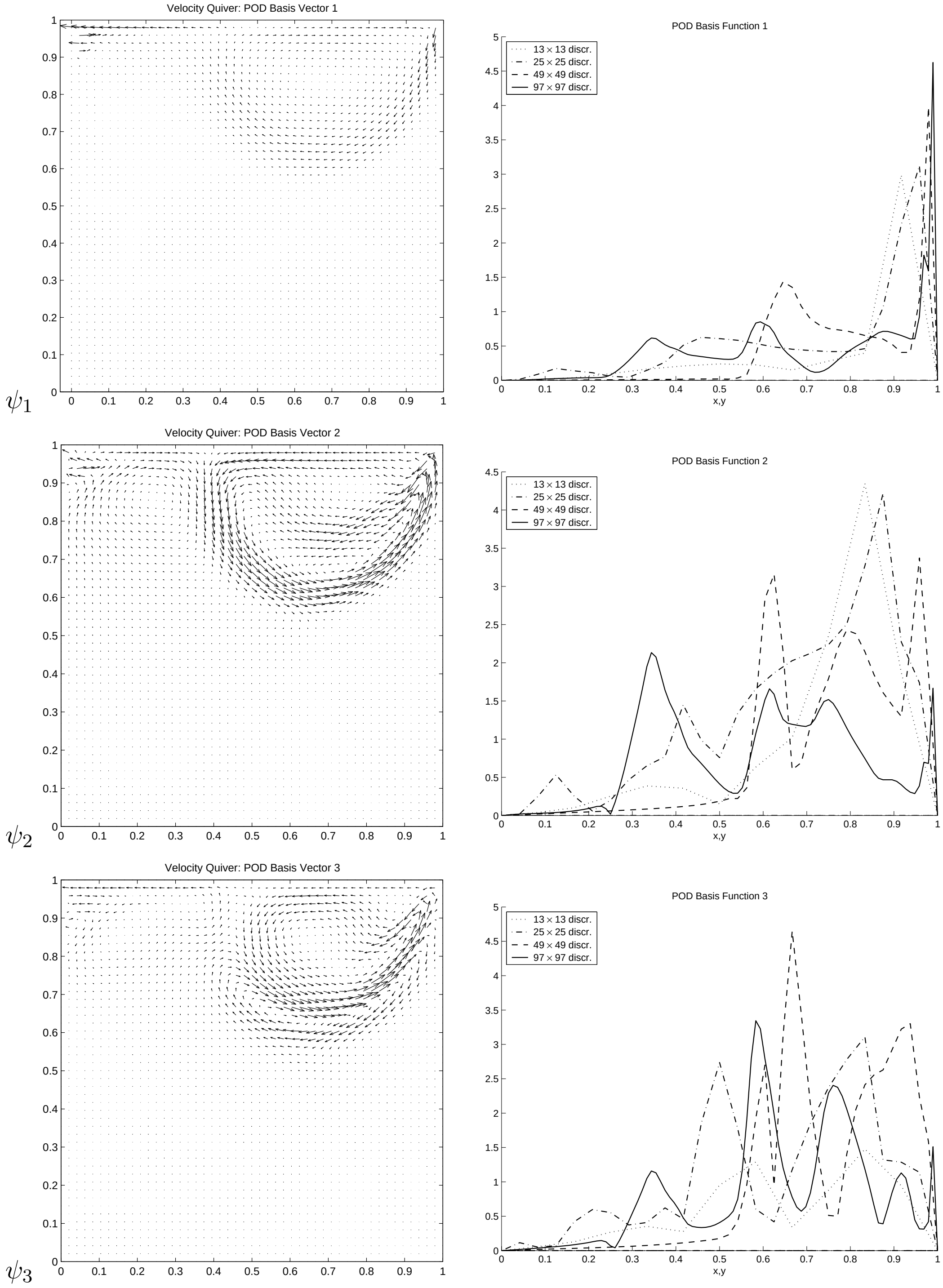


Figure 4.12: The dependence of the first three POD basis functions on the underlying spatial discretization at $Re = 20,000$. On the left, the velocity quiver diagram for each POD basis function taken on the 49×49 mesh. On the right, the velocity norms at various discretization levels on the mesh diagonal running from the lower left to the upper right of the cavity.

	$\delta = 0.0$		$\delta = 0.5$		$\delta = 1.0$	
i	λ_i	% energy	λ_i	% energy	λ_i	% energy
1	2.6848e-1	97.65551	2.6760e-1	97.66589	2.6675e-1	97.67584
2	5.5447e-3	99.67233	5.4990e-3	99.67281	5.4549e-3	99.67328
3	7.0999e-4	99.93058	7.0621e-4	99.93055	7.0255e-4	99.93053
4	1.4324e-4	99.98268	1.4274e-4	99.98264	1.4224e-4	99.98261
5	3.5888e-5	99.99573	3.5800e-5	99.99571	3.5710e-5	99.99569
6	8.7553e-6	99.99892	8.7617e-6	99.99891	8.7659e-6	99.99890
7	2.1752e-6	99.99971	2.1864e-6	99.99971	2.1966e-6	99.99970
8	5.7641e-7	99.99992	5.8131e-7	99.99992	5.8580e-7	99.99992
9	1.5228e-7	99.99997	1.5382e-7	99.99997	1.5524e-7	99.99997
10	3.9219e-8	99.99999	3.9691e-8	99.99999	4.0133e-8	99.99999
15	1.3507e-10	99.99999	1.3902e-10	99.99999	1.4285e-10	99.99999
20	1.6886e-14	99.99999	1.7626e-14	99.99999	1.8371e-14	99.99999

Table 4.17: Effect of streamline diffusion on POD basis at $Re = 100$ using a 97×97 mesh. Some characteristic eigenvalues of the POD for a velocity field in a two-dimensional driven cavity with different levels of streamline diffusion. The right column for each value of the streamline diffusion parameter δ shows the relative energy projected onto the first i eigenfunctions.

	$\delta = 0.0$		$\delta = 0.5$		$\delta = 1.0$	
i	λ_i	% energy	λ_i	% energy	λ_i	% energy
1	1.7431e-1	93.68490	1.7184e-1	93.97388	1.6945e-1	94.21630
2	8.8650e-3	98.44931	8.3859e-3	98.55980	7.9717e-3	98.64843
3	1.9435e-3	99.49384	1.7908e-3	99.53915	1.6658e-3	99.57459
4	6.1927e-4	99.82666	5.6010e-4	99.84545	5.1250e-4	99.85953
5	2.0224e-4	99.93536	1.7943e-4	99.94358	1.6183e-4	99.94951
6	7.2105e-5	99.97411	6.2656e-5	99.97784	5.5664e-5	99.98046
7	2.8412e-5	99.98938	2.4219e-5	99.99108	2.1221e-5	99.99226
8	1.1502e-5	99.99556	9.6008e-6	99.99633	8.2931e-6	99.99687
9	4.7844e-6	99.99813	3.8904e-6	99.99846	3.2932e-6	99.99870
10	2.0343e-6	99.99923	1.6284e-6	99.99935	1.3514e-6	99.99945
15	2.1952e-8	99.99999	1.9760e-8	99.99999	1.7549e-8	99.99999
20	1.9428e-10	99.99999	1.9755e-10	99.99999	1.9455e-10	99.99999

Table 4.18: Effect of streamline diffusion on POD basis at $Re = 400$ using a 97×97 mesh. Some characteristic eigenvalues of the POD for a velocity field in a two-dimensional driven cavity with different levels of streamline diffusion. The right column for each value of the streamline diffusion parameter δ shows the relative energy projected onto the first i eigenfunctions.

	$\delta = 0.5$		$\delta = 1.0$		$\delta = 2.0$	
i	λ_i	% energy	λ_i	% energy	λ_i	% energy
1	2.6235e-1	80.03557	2.4723e-1	79.98697	2.2717e-1	84.20407
2	3.1856e-2	89.75400	3.1308e-2	90.11608	2.1233e-2	92.07430
3	1.2788e-2	93.65549	1.3128e-2	94.36356	7.9807e-3	95.03242
4	6.7911e-3	95.72727	4.9458e-3	95.96369	5.0936e-3	96.92044
5	3.4055e-3	96.76620	3.7980e-3	97.19246	2.6443e-3	97.90059
6	3.1998e-3	97.74238	2.5314e-3	98.01146	1.3961e-3	98.41806
7	1.7054e-3	98.26267	1.2340e-3	98.41072	1.1502e-3	98.84442
8	1.2146e-3	98.63322	1.0937e-3	98.76458	8.9123e-4	99.17476
9	9.4429e-4	98.92129	7.9792e-4	99.02273	5.8731e-4	99.39245
10	7.0587e-4	99.13663	6.6877e-4	99.23909	3.9792e-4	99.53995
15	2.4443e-4	99.69824	1.8868e-4	99.76622	9.4337e-5	99.88854
20	7.8879e-5	99.89547	5.8986e-5	99.92398	2.3552e-5	99.97226

Table 4.19: Effect of streamline diffusion on POD basis at $Re = 10,000$ using a 97×97 mesh. Some characteristic eigenvalues of the POD for a velocity field in a two-dimensional driven cavity with different levels of streamline diffusion. The right column for each value of the streamline diffusion parameter δ shows the relative energy projected onto the first i eigenfunctions.

	$\delta = 0.5$		$\delta = 1.0$		$\delta = 2.0$	
i	λ_i	% energy	λ_i	% energy	λ_i	% energy
1	1.6451e-1	73.85283	1.6113e-1	76.80181	1.4958e-1	80.43133
2	2.6146e-2	85.59026	1.8316e-2	85.53209	1.6924e-2	89.53111
3	1.1505e-2	90.75515	1.0755e-2	90.65847	7.6280e-3	93.63255
4	4.9892e-3	92.99488	5.6412e-3	93.34724	3.9044e-3	95.73191
5	3.1183e-3	94.39473	4.7603e-3	95.61611	2.0691e-3	96.84447
6	2.5374e-3	95.53380	2.1188e-3	96.62599	1.6454e-3	97.72922
7	2.1729e-3	96.50925	1.4895e-3	97.33596	1.2085e-3	98.37901
8	1.6613e-3	97.25502	1.0290e-3	97.82641	7.1080e-4	98.76120
9	1.0501e-3	97.72646	9.1137e-4	98.26079	4.4922e-4	99.00274
10	7.5787e-4	98.06667	7.4397e-4	98.61539	3.9553e-4	99.21542
15	3.3340e-4	99.06608	2.2144e-4	99.40427	1.2568e-4	99.75685
20	1.6381e-4	99.54175	8.8454e-5	99.74533	3.7295e-5	99.91673

Table 4.20: Effect of streamline diffusion on POD basis at $Re = 20,000$ using a 97×97 mesh. Some characteristic eigenvalues of the POD for a velocity field in a two-dimensional driven cavity with different levels of streamline diffusion. The right column for each value of the streamline diffusion parameter δ shows the relative energy projected onto the first i eigenfunctions.

Chapter 5

The Streamline Diffusion POD Model

As was discussed in the introduction and outline, the fidelity of POD-based models is dependent on the problem data used to generate the model. If the problem data (e.g., boundary conditions, Reynolds number or boundary geometry) change, then the accuracy of the POD-based model is compromised. This is obviously problematic in applications to optimal control using boundary controls, where the boundary conditions evolve during the optimization process. The need for periodic updating of the POD-based model in such applications is well-established in the POD literature, and a variety of problem-dependent methods have been proposed to ameliorate this difficulty. In order to guarantee convergence to an optimal solution, Fahl [41] used a trust-region framework for a driven cavity problem, replacing the entire snapshot set when necessary based on information from the trust-region procedure. Afanasiev/Hinze [3] used distributed controls to study the optimal control of wake flow in a channel with a circular cylinder. As the control profile generated by the POD-based model evolved, the model was periodically updated by solving the full high-order problem with the updated control, adding the new snapshots to the snapshot set, and subsequently generating a new POD-based model. Graham et al. [55] studied the use of POD methods to generate optimal control strategies for incompressible unsteady wake flow behind a circular cylinder at a Reynolds number of 100, with the goal of controlling vortex shedding behind the cylinder via cylinder rotation. The applicable range of the POD basis functions generated with the cylinder driven at a frequency corresponding to that of natural vortex separation was extended by the addition of generalized basis functions from an enhanced snapshot set, generated by exciting the flow in the high-order model with representative control action and different initial conditions. Nevertheless, the model had to be periodically reset during the optimization process.

Common to all of these basis augmentation methods is the need to repeatedly solve the full Navier-Stokes equations, which is precisely what reduced-order methods are supposed to avoid. In this sense, it would be advantageous if one could acquire the information for POD basis generation and augmentation with less computational effort. For data generated by DNS this might be accomplished by using coarser grids to compute approximate solutions, which can then be used as starting points for optimization on finer grids (cf. [10, 11, 15, 58]). Since the coarser grids may still require considerable computational effort, it makes sense to extend the POD approach to the coarser grids as well.

The intuitive appeal of this idea notwithstanding, before this procedure can be evaluated in the fluid flow context, another complication must be dealt with. It is well-known that mixed convection-diffusion problems with dominant convection may suffer from numerical instability problems (cf. Chapter 3) that can lead to oscillations in the solution and failure of the numerical procedure to converge. This problem can be eliminated either by resorting to finer grids, which we wish to avoid, or by adding some sort of stabilization, e.g., upwinding or streamline diffusion. Though both techniques are sufficient to stabilize the solution, we are especially interested in the streamline diffusion finite element (SD-FEM), because of its better theoretical convergence properties and because it is naturally formulated as a Petrov-Galerkin method, allowing easy incorporation into the POD-based model.

In Section 5.1, a POD-based reduced-order model is formulated for the driven cavity problem described in Section 2.2.3 by projecting the Navier-Stokes equations onto the POD basis of Section 4.1. Numerical testing reveals that the streamline diffusion needed to stabilize the high-order Navier-Stokes solution procedure used for the generation of the snapshot ensemble results in POD basis functions that are incompatible with the standard POD-based reduced-order model. As a remedy, in Section 5.2 we suggest and experiment with approaches for incorporating the stabilization action into the reduced-order model. We show that the resulting procedure leads to a POD-based model that is tuned to the high-order Navier-Stokes solver, so that models derived from rougher discretizations can be used with confidence.

5.1 The POD-Based Model

Before we can project the Navier-Stokes equations onto our POD basis, we must resolve some issues concerning the form of the reduced-order model. First, if the model is to be useful, it must have some way of simulating the boundary conditions of the Navier-Stokes problem. Second, in order to maintain consistency with the Navier-Stokes equations, the reduced-order model must reflect the incompressible nature of the flow. Finally, it is advantageous during the construction of the model if the POD basis functions satisfy homogeneous boundary conditions.

Assuming that the boundary conditions can be written in the form

$$\mathbf{g}(t, \mathbf{x}) = \gamma(t)\mathbf{h}(\mathbf{x}), \tag{5.1}$$

we can accomplish all objectives by means of the *control function method* (cf. Burns/Ou [24] or Graham et al. [55]), in which the snapshots are modified by subtracting a suitable reference field from each snapshot prior to the generation of the POD basis. The reference field must be divergence-free and satisfy both the homogeneous and inhomogeneous boundary conditions of the Navier-Stokes problem, but is otherwise arbitrary. An admissible reference field $\mathbf{u}_c$ can easily be generated by solving the Navier-Stokes problem with $\gamma(t) \equiv 1$ in (5.1). The snapshot set then takes the form

$$\mathbf{v}_i = \mathbf{u}_i - \gamma(t)\mathbf{u}_c \quad i = 1, \ldots, n. \tag{5.2}$$

Furthermore, it is also convenient to subtract the resulting mean flow field $\mathbf{u}_n = \frac{1}{n}\sum_{i=1}^{n} \mathbf{v}_i$ from the snapshots prior to generating the POD basis. This modified snapshot set[1]

$$\mathbf{w}_i = \mathbf{u}_i - \gamma(t)\mathbf{u}_c - \mathbf{u}_n \quad i = 1, \ldots, n \tag{5.3}$$

is then used to generate a POD basis according to one of the methods of Section 4.1.1. One can easily see that the POD basis elements generated from the modified snapshots are divergence-free and homogeneous, and the velocity expansion

$$\mathbf{u}(t, \mathbf{x}) = \mathbf{u}_n(\mathbf{x}) + \gamma(t)\mathbf{u}_c(\mathbf{x}) + \sum_{i=1}^{m} y_i(t)\mathbf{\Psi}_i(\mathbf{x}) \tag{5.4}$$

satisfies the boundary conditions of the problem.

5.1.1 The Galerkin POD Projection

After extracting the homogeneous and divergence-free POD basis from the snapshot data and selecting the number m of POD basis elements desired for the reduced-order model (5.4), we determine the coefficients $y_j(t)$, $j = 1, \ldots, m$ by projecting the momentum equation

$$\frac{\partial \mathbf{u}}{\partial t} - \nu\Delta\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla p = 0 \tag{5.5}$$

onto the POD basis and solving the resulting system of ordinary differential equations. In order to simplify the following discussion, and because our model problem uses only boundary controls, we have set $\mathbf{f} = 0$ in (5.5), though a nonzero value can easily be accommodated by the model.

The Galerkin projection

$$\left(\frac{\partial \mathbf{u}}{\partial t}, \mathbf{\Psi}_j\right) = -\left(\mathbf{u}\cdot\nabla\mathbf{u}, \mathbf{\Psi}_j\right) - \left(\nabla p, \mathbf{\Psi}_j\right) + \nu\left(\Delta\mathbf{u}, \mathbf{\Psi}_j\right) \tag{5.6}$$

of (5.5) onto the POD basis leads to a system

$$\begin{aligned}\dot{\mathbf{y}}(t) = {} & \mathcal{M}_0 + \gamma(t)\mathcal{M}_1 + \gamma^2(t)\mathcal{M}_2 + \dot{\gamma}(t)\mathcal{M}_c \\ & + \mathcal{M}_3\mathbf{y} + \gamma(t)\mathcal{M}_4\mathbf{y} + \mathcal{M}_5(\mathbf{y}, \mathbf{y})\end{aligned} \tag{5.7}$$

of ordinary differential equations, with vectors $\mathcal{M}_0$, $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_c \in \mathbb{R}^m$, matrices $\mathcal{M}_3$, $\mathcal{M}_4 \in \mathbb{R}^{m,m}$ and a bilinear term $\mathcal{M}_5(\mathbf{y}, \mathbf{y}) : \mathbb{R}^{m,m} \to \mathbb{R}^m$. The details of this derivation, which are straightforward but rather laborious, can be found in Appendix A.

[1] We ask the reader's forgiveness for the abuse of notation regarding $\mathbf{u}_n$, which has been used to denote the average flow field, as well as the n-th snapshot.

5.1.2 Error Measures for the POD-Based Model

In order to obtain a numerical measure for the ability of the reduced-order model to accurately simulate the Navier-Stokes equations we need two measures: One for the ability of the truncated POD basis to represent the snapshot data, and a similar measure for the reduced-order model. To this end, we essentially follow Graham et al. [55] and consider three velocity fields: The field $\mathbf{u}(t, \mathbf{x})$ generated by the original snapshots, the predicted field $\tilde{\mathbf{u}}(t, \mathbf{x})$ generated by the reduced-order model (5.4), and the projected field $\hat{\mathbf{u}}(t, \mathbf{x})$, defined by direct projection of the snapshots onto the POD basis.

According to Theorem 4.3 the truncated POD basis is optimal in the sense of (4.7) for modeling the snapshot ensemble though there will be some error due to the truncation; thus, the projected field

$$\hat{\mathbf{u}}(t, \mathbf{x}) = \mathbf{u}_n(\mathbf{x}) + \gamma(t)\mathbf{u}_c(\mathbf{x}) + \sum_{i=1}^{m} \hat{y}_i(t)\mathbf{\Psi}_i(\mathbf{x}) \tag{5.8}$$

with the modes

$$\hat{y}_i(t) = (\mathbf{u} - \mathbf{u}_n(\mathbf{x}) - \gamma(t)\mathbf{u}_c(\mathbf{x}), \mathbf{\Psi}_i), \tag{5.9}$$

differs from the field $\mathbf{u}(t, \mathbf{x})$ depending only on the degree of the truncation. To measure this error, we define the *absolute projection error*

$$\hat{E}_{\text{abs}} = (\mathbf{u} - \hat{\mathbf{u}}, \mathbf{u} - \hat{\mathbf{u}}) \tag{5.10}$$

and the *relative projection error*

$$\hat{E}_{\text{rel}} = \frac{(\mathbf{u} - \hat{\mathbf{u}}, \mathbf{u} - \hat{\mathbf{u}})}{\|\mathbf{u} - \mathbf{u}_n\|^2}. \tag{5.11}$$

In contrast to the direct projection, the error for reduced-order model

$$\tilde{\mathbf{u}}(t, \mathbf{x}) = \mathbf{u}_n(\mathbf{x}) + \gamma(t)\mathbf{u}_c(\mathbf{x}) + \sum_{i=1}^{m} y_i(t)\mathbf{\Psi}_i(\mathbf{x}) \tag{5.12}$$

with the POD modes $y_i(t)$ will be greater since the POD modes must satisfy the system (5.7). To measure the error for the reduced-order model, we define the *absolute prediction error*

$$\tilde{E}_{\text{abs}} = (\mathbf{u} - \tilde{\mathbf{u}}, \mathbf{u} - \tilde{\mathbf{u}}) \tag{5.13}$$

and the *relative prediction error*

$$\tilde{E}_{\text{rel}} = \frac{(\mathbf{u} - \tilde{\mathbf{u}}, \mathbf{u} - \tilde{\mathbf{u}})}{\|\mathbf{u} - \mathbf{u}_n\|^2}. \tag{5.14}$$

Remark 5.1. The value of $\hat{\mathbf{u}}$ will always be smaller than $\tilde{\mathbf{u}}$; however, if the reduced-order model is optimal for the given snapshots, the projected modes should match the predicted modes, and the projected and the predicted errors should be equal. We note that similar error measures can be defined for the individual POD basis functions by adjusting the range of the indices in the sums of (5.8) and (5.12).

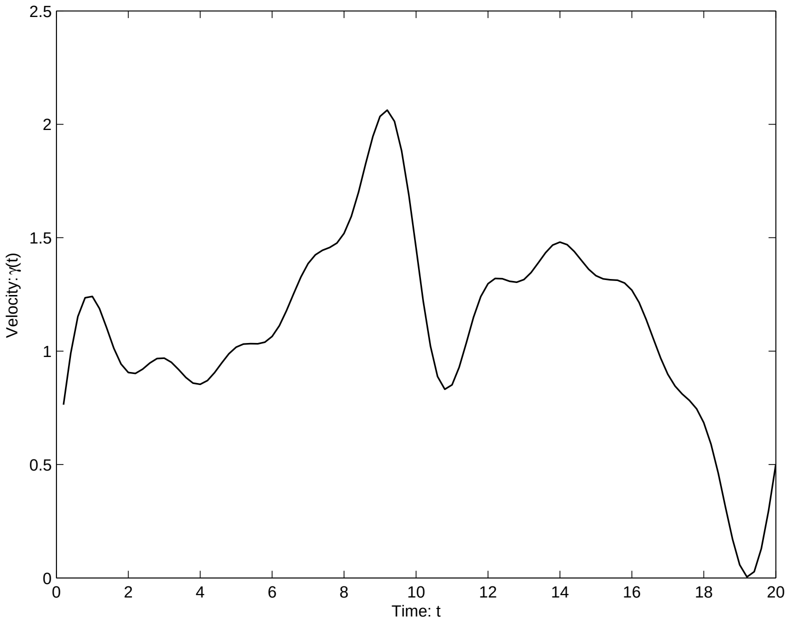


Figure 5.1: Velocity profile for model problem.

In addition to the temporally local errors described above, we define two global measures of error in the sense of (4.8); the *average reconstruction error* for the POD basis,

$$E_{\mathrm{PROJ}} = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{u}_i - \hat{\mathbf{u}}_i\|\,, \tag{5.15}$$

and the *average simulation error* for the POD-based reduced-order model,

$$E_{\mathrm{POD}} = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{u}_i - \tilde{\mathbf{u}}_i\|\,. \tag{5.16}$$

Remark 5.1 also applies to these global measures.

5.1.3 Numerical Analysis of POD-Based Model

To test the basic POD-based model we used the driven cavity problem described in Section 2.2.3 with simulation time $T = 20$ seconds to generate 100 snapshots at uniform time intervals of 0.2 seconds at Reynolds number $Re = 400$. For the time-dependent boundary profile (see (2.28) and Figure 5.1) we used

$$\begin{aligned}\gamma(t) = \frac{1}{2} + \frac{3}{\pi}\Bigg[&\sin(\pi t/20) + \frac{1}{3}\sin(3\pi t/10) + \\ &\frac{1}{5}\sin(5\pi t/10) + \frac{1}{7}\sin(7\pi t/10) + \frac{1}{9}\sin(9\pi t/10)\Bigg].\end{aligned} \tag{5.17}$$

Nine basis functions were sufficient to capture 99.9% of the flow energy. The first three projected and POD (predicted) modes are plotted in Figure 5.2, where there appears to be

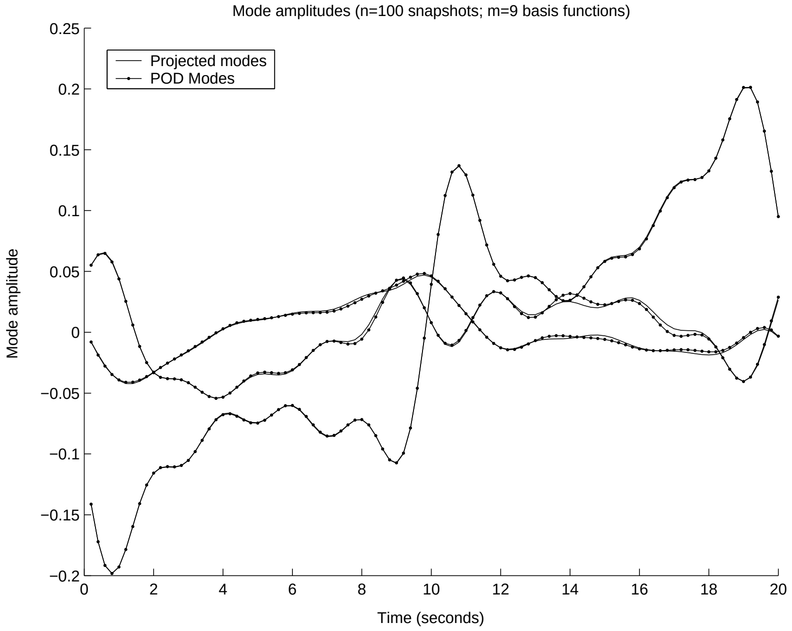


Figure 5.2: First three predicted and projected modes at $Re = 400$.

very good agreement. A more exact analysis is provided by absolute and relative errors, which are depicted in Figures 5.3 and 5.4 on Page 83. The time profile of the predicted error mirrors that of the projected error, with the predicted error as expected slightly larger. Both the absolute and relative errors are of the same order for the predicted and projected errors, indicating that the POD-based reduced-order model is nearly optimal for representing the space spanned by the snapshot ensemble.

5.1.4 Effect of Stabilization on the POD-Based Model

To test the effect of streamline diffusion on the POD-based model, we set $\delta = 1.0$ in (3.64) and again solved the model problem at Reynolds number $Re = 400$, generating 100 snapshots at intervals of 0.2 seconds. Note that it was not necessary to add streamline diffusion to solve the Navier-Stokes equations at $Re = 400$, and we do so only for illustrative purposes. In Figure 5.5 on Page 84 the first projected and POD (predicted) modes are compared. We see from the obvious difference between them that the resulting basis functions are no longer optimal for low-order simulation of the Navier-Stokes problem.

Table 5.1 presents the results for POD-based models generated from numerical simulations with various levels of streamline diffusion added, the streamline diffusion parameter ranging from $\delta = 0.0$ to $\delta = 2.0$. In every case, nine basis functions were sufficient to capture about 99.9% of the system energy. It is noteworthy that the percentage of system energy captured by the first nine basis functions increases steadily with increasing streamline diffusion, reflecting the action of the added stabilization on the system dynamics. As expected, the average reconstruction error E_{PROJ} declines as the fraction of system energy

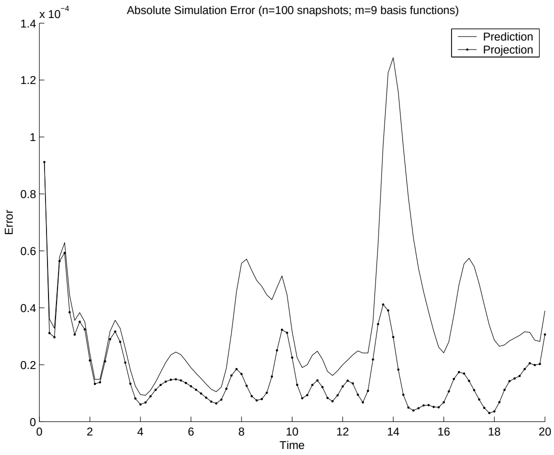


Figure 5.3: Absolute error at $Re = 400$.

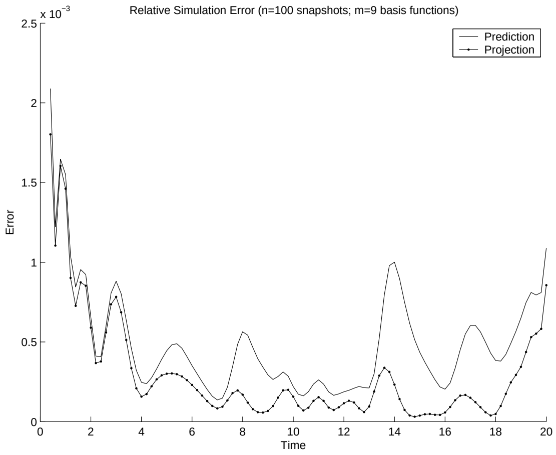


Figure 5.4: Relative error at $Re = 400$.

captured by the basis functions increases; however, as the streamline diffusion parameter increases, the average simulation error E_{POD} increases rapidly. While E_{PROJ} and E_{POD} are of the same order with no streamline diffusion added, E_{POD} is about two orders of ten greater than E_{PROJ} for $\delta \geq 1.0$. The compatibility between the POD basis functions and the standard POD reduced-order model deteriorates with increasing streamline diffusion, such that POD-based model and the high-order numerical solver are no longer "in tune" with each other.

The effect becomes even more pronounced at higher Reynolds numbers for which some sort of stabilization is necessary if the numerical solution procedure is to converge. In Figure 5.6 on Page 85 we present the results for $\delta = 1.0$ and $Re = 10,000$, again on a 49×49 mesh.

This incongruence between the POD basis and the dynamical system can become catastrophal as is illustrated in Figure 5.7, where the projected and predicted modes for a POD basis generated using $\gamma(t) \equiv 1.0$ with $\delta = 1.0$ and $Re = 10,000$ on a 13×13 mesh are presented. The ODE solver for the predicted modes failed to converge for this simple example, with the solver terminating at about 19 sec. POD-based models are clearly useless if one cannot solve (5.7) for the coefficients $y_i(t)$.

5.2 The Streamline Diffusion POD Model

The numerical results of Section 5.1.3 show that the POD basis generated by the standard POD-based model may not be optimal for deriving a reduced-order model for the Navier-Stokes problem when stabilization, such as streamline diffusion, is required during the numerical solution of the Navier-Stokes equations. The solution produced by the solver including the stabilization term converges to the true solution at finer discretizations, but if we are interested in using rougher discretizations, we must somehow modify the POD-based model to account for the stabilization. In this section we modify the Galerkin

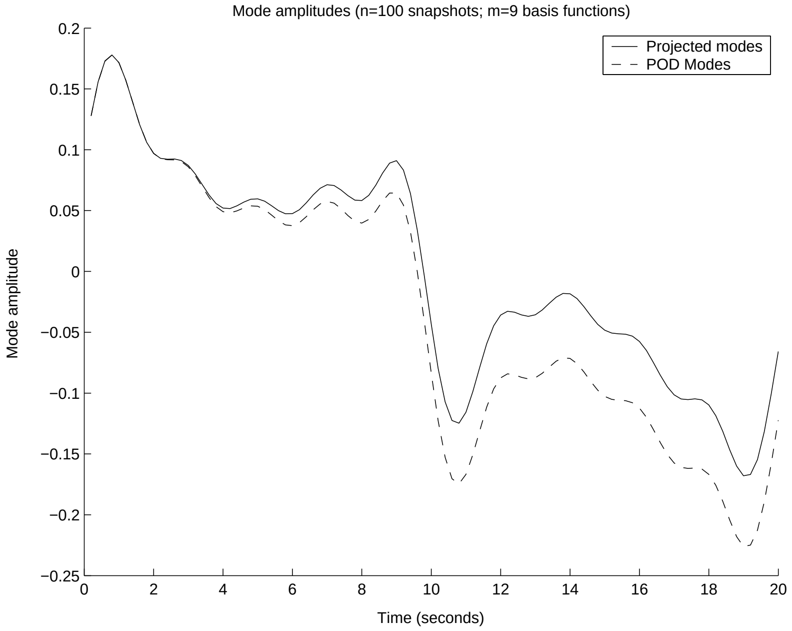


Figure 5.5: Projected and predicted modes at $Re = 400$ with streamline diffusion added in the Navier-Stokes solver ($\delta = 1.0$).

δ	m	%Energy	E_{PROJ}	E_{POD}
0.00	9	99.86	1.69833 e-5	3.63561 e-5
0.05	9	99.87	1.62830 e-5	4.61308 e-5
0.10	9	99.87	1.56084 e-5	6.84643 e-5
0.20	9	99.88	1.43345 e-5	1.54284 e-4
0.50	9	99.90	1.11760 e-5	7.21537 e-4
1.00	9	99.92	7.80152 e-6	2.07090 e-3
2.00	9	99.94	4.58066 e-6	4.78961 e-3

Table 5.1: Deterioration of model accuracy with increasing streamline diffusion parameter at $Re = 400$ on a 49×49 mesh.

projection so that our model more accurately reflects the action of the Navier-Stokes solver, thus improving compatibility between the solver and the model.

5.2.1 Motivation and Formulation of the Model

The essential difference between the usual discrete formulation (2.35)-(2.36) of the Navier-Stokes problem and the problem solved by our solver consists of the stabilization term in (3.64). After extracting snapshots for a given discretization of the Navier-Stokes equations, we seek to harmonize the solver and the POD-based model by adding an approximation of the streamline diffusion term to the Galerkin projection. This is accomplished by replacing (5.6) with

$$\left(\frac{\partial \mathbf{u}}{\partial t}, \boldsymbol{\Psi}_j\right) = -\left(\mathbf{u}\cdot\nabla\mathbf{u}, \boldsymbol{\Psi}_j\right) + \nu\left(\Delta\mathbf{u}, \boldsymbol{\Psi}_j\right) - \delta_T\left(\mathbf{u}\cdot\nabla\mathbf{u}, \mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j\right), \tag{5.18}$$

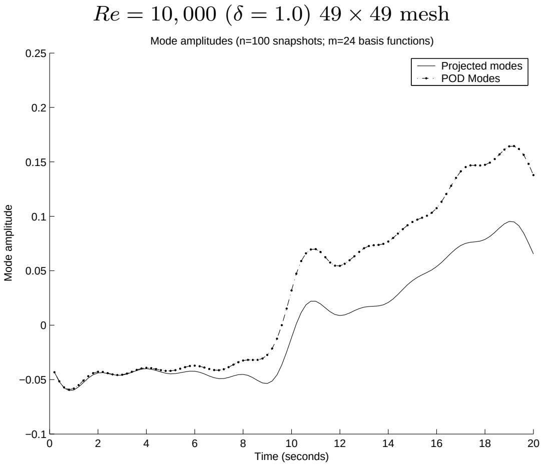


Figure 5.6: First POD modes diverge in presence of streamline diffusion.

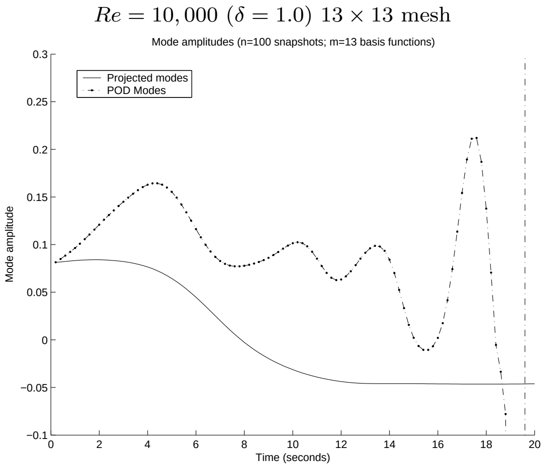


Figure 5.7: ODE system fails to converge.

where $\delta_T = \delta_T(\mathbf{u}, \mathcal{T}^h)$ is a local weighting parameter corresponding to (3.64) that depends on the local velocity and discretization used to solve the Navier-Stokes equations. We call (5.18) the *streamline diffusion POD (SDPOD) model.*

This leads, after considerations analogous to those of Section 5.1.1, to the system

$$\begin{aligned} \dot{y}_j(t) = -\nu\,(\nabla\mathbf{u}, \nabla\mathbf{\Psi}_j) - \dot{\gamma}(t)\,(\mathbf{u}_c, \mathbf{\Psi}_j) - (\mathbf{u}\cdot\nabla\mathbf{u}, \mathbf{\Psi}_j) \\ -\delta_T\,(\mathbf{u}\cdot\nabla\mathbf{u}, \mathbf{u}\cdot\nabla\mathbf{\Psi}_j)\,, \quad j = 1, \dots, m. \end{aligned} \tag{5.19}$$

for the coefficients $y_j(t)$ of (5.4).

The term δ_T as used by FEATFLOW (see (3.64)) presents difficulties for the SDPOD model because it depends on the local Reynolds number, that is, it depends nonlinearly and locally on the velocity $\mathbf{u}(t, \mathbf{x}) = \mathbf{u}_n(\mathbf{x}) + \gamma(t)\mathbf{u}_c(\mathbf{x}) + \sum_{i=1}^m y_i(t)\mathbf{\Psi}_i(\mathbf{x})$, making it impossible to separate the temporal variables from the spatial variables, which we must do if we are to achieve a true reduced-order model. However, an easy reformulation of δ_T gives

$$\begin{aligned} \delta_T &= \delta \cdot \frac{h_T}{\parallel \mathbf{u} \parallel_\Omega} \cdot \frac{2Re_T}{1 + Re_T} \\ &= \delta \cdot \frac{h_T}{\parallel \mathbf{u} \parallel_\Omega} \cdot \frac{2\frac{\|\mathbf{u}\|_T \cdot h_T}{\nu}}{1 + \frac{\|\mathbf{u}\|_T \cdot h_T}{\nu}} \\ &= \delta \cdot \frac{2h_T}{\parallel \mathbf{u} \parallel_\Omega} \cdot \frac{\parallel \mathbf{u} \parallel_T \cdot h_T}{\nu + \parallel \mathbf{u} \parallel_T \cdot h_T} \quad \underset{\nu\to 0}{\longrightarrow} \quad \delta \cdot \frac{2h_T}{\parallel \mathbf{u} \parallel_\Omega}, \end{aligned}$$

so that for reasonably large Reynolds numbers we can replace δ_T with

$$\delta_T^m = \delta \cdot \frac{2h_T}{\parallel \mathbf{u} \parallel_\Omega}, \tag{5.20}$$

and expect to get a reasonable approximation to the streamline diffusion term in our model.

Replacing δ_T with δ_T^m and substituting the velocity expansion (5.4) into the modified Galerkin projection (5.20) we get the system

$$\begin{aligned}\dot{\mathbf{y}}(t) = {} & \mathcal{M}_0 + \gamma(t)\mathcal{M}_1 + \gamma^2(t)\mathcal{M}_2 + \dot{\gamma}(t)\mathcal{M}_c + \mathcal{M}_3\mathbf{y} + \gamma(t)\mathcal{M}_4\mathbf{y} + \mathcal{M}_5(\mathbf{y},\mathbf{y}) \\ & + \delta_T^m \left[\mathcal{M}_0^\delta + \gamma(t)\mathcal{M}_1^\delta + \gamma^2(t)\mathcal{M}_2^\delta + \mathcal{M}_3^\delta\mathbf{y} + \gamma(t)\mathcal{M}_4^\delta\mathbf{y} \right.\\ & \left. + \mathcal{M}_5^\delta(\mathbf{y},\mathbf{y}) + \gamma^3(t)\mathcal{M}_6^\delta + \gamma^2\mathcal{M}_7^\delta\mathbf{y} + \gamma(t)\mathcal{M}_8^\delta(\mathbf{y},\mathbf{y}) + \mathcal{M}_9^\delta(\mathbf{y},\mathbf{y},\mathbf{y})\right], \end{aligned} \tag{5.21}$$

where the new terms, including the vectors $\mathcal{M}_0^\delta$, $\mathcal{M}_1^\delta$, $\mathcal{M}_2^\delta$, $\mathcal{M}_c^\delta$, $\mathcal{M}_6^\delta \in \mathbb{R}^m$, the matrices $\mathcal{M}_3^\delta$, $\mathcal{M}_4^\delta$, $\mathcal{M}_7^\delta \in \mathbb{R}^{m,m}$, the bilinear terms $\mathcal{M}_5^\delta(\mathbf{y},\mathbf{y})$, $\mathcal{M}_8^\delta(\mathbf{y},\mathbf{y}) : \mathbb{R}^{m,m} \to \mathbb{R}^m$ and the trilinear term $\mathcal{M}_9^\delta(\mathbf{y},\mathbf{y},\mathbf{y}) : \mathbb{R}^{m,m,m} \to \mathbb{R}^m$ are derived in Appendix A.

Remark 5.2. To understand why the inclusion of the high-order stabilization term to the Galerkin POD projection is necessary, let us consider the difference between the systems (5.7) and (5.21), which are used to determine the modes of the POD-based and SDPOD-based models, respectively. Subtracting the right-hand side of (5.7) from the right hand side of (5.21), we obtain the system

$$\begin{aligned}\dot{\mathbf{y}}(t) = {} & \delta_T^m \left[\mathcal{M}_0^\delta + \gamma(t)\mathcal{M}_1^\delta + \gamma^2(t)\mathcal{M}_2^\delta + \mathcal{M}_3^\delta\mathbf{y} + \gamma(t)\mathcal{M}_4^\delta\mathbf{y} \right.\\ & \left. + \mathcal{M}_5^\delta(\mathbf{y},\mathbf{y}) + \gamma^3(t)\mathcal{M}_6^\delta + \gamma^2\mathcal{M}_7^\delta\mathbf{y} + \gamma(t)\mathcal{M}_8^\delta(\mathbf{y},\mathbf{y}) + \mathcal{M}_9^\delta(\mathbf{y},\mathbf{y},\mathbf{y})\right] \end{aligned} \tag{5.22}$$

for determining the difference between the models. Note that (5.22) is itself a nonlinear system of ordinary differential equations (cf. [59, 89, 85]), so that even for finer meshes (small h) we have no guarantee that the modes of the standard POD-based model will be compatible with the POD basis generated from the snapshot ensemble, especially for longer simulation periods (large T). Moreover, even if the difference between the standard POD modes and the Fourier coefficients of the direct projection of the snapshots onto the POD basis is small, it might still be sensible to go ahead and use the SDPOD method; if one is willing to tolerate a certain error, then this might be better used for further truncation of the POD basis, which reduces the size of the ODE system.

Remark 5.3. Use of (5.20) in an optimization context is complicated by the fact that $\| \mathbf{u} \|_\Omega$ is not differentiable. For this reason we have further simplified (5.20) by recording the values of $\| \mathbf{u} \|_\Omega$ during the high-order solution process along with the snapshots and using the recorded values in the SDPOD model. This simplification might result in some deterioration of model fidelity if used for optimization, but for Reynolds numbers large enough to require stabilization, we do not expect the deterioration to be significant.

Remark 5.4. It should be noted that the difficulties presented by the local dependence of the parameter δ_T on the velocity are not all that restrictive in general. To begin with, in many cases of interest the streamline diffusion parameter is not, in fact, dependent on the local Reynolds number as was the case for the SDFEM approach of Section (3.3) (see also [135]). Moreover, where the parameter δ_T does depend locally on the velocity, it may be possible to modify the underlying SDFEM approach without significantly impairing convergence. Finally, if all else fails, we can always resort to an approximation of the sort used above.

5.2.2 Numerical Analysis of the Streamline Diffusion POD Model

To assess the ability of the enhanced basis functions to represent the space spanned by the snapshots, we ran simulations of the Navier-Stokes problem at $Re = 10,000$ over a range of discretizations with the streamline diffusion parameter set to $\delta = 1.0$. One hundred snapshots were generated at each level, and enough basis functions were chosen at each level to ensure that 99.9% of the system energy was captured. As the discretization becomes finer, both the SDFEM and classical FEM methods converge in theory to the true solution so that one would expect the absolute error of the standard POD-based model to continue declining with finer discretizations.

Figure 5.8 on Page 89 plots the absolute error for the standard POD Model with POD bases generated at the various levels of discretization. As expected, the error falls rapidly as the discretization becomes finer. Nevertheless, even on a 97 × 97 mesh the absolute error reaches values of about $\tilde{E}_{\text{abs}} \approx 2 \times 10^{-2}$. In contrast, the corresponding error for the SDPOD model is less than 10^{-5} at all times and for all discretizations. The average error is illustrated in Table 5.2 for all levels of discretization. The projection error ranges between 10^{-6} and 10^{-5} for all levels, with no discernable pattern, which is not surprising since we would expect this error to be dependent on the amount of system energy captured by the basis functions, which is constant here. The simulation errors for the SDPOD models are also as expected; since the SDPOD model has been tuned to the high-order system used to generate the snapshots and basis functions, the values for E_{SDPOD} mirror those of E_{PROJ} quite closely, being only slightly higher. The values for E_{POD} appear to improve somewhat on average with finer discretizations, but the pattern is not as clear as one might have expected. This reflects our general experience; in experiments with various settings of mesh refinement level, streamline diffusion parameter δ, energy captured and number of snapshots, we found that the results for the standard POD method were fairly unpredictable.

Comparison with a Reference Solution

When the high-order numerical solver requires stabilization to guarantee a solution, the results above confirm that the SDPOD method is more compatible with the resulting POD basis than is the standard POD-based model. We now go a step further and test the ability of the SDPOD model generated from rough discretizations to match the numerical solution generated at finer discretizations. To this end, we used very fine time stepping to generate a reference solution on a 193 × 193 grid at $Re = 10,000$ using no streamline diffusion ($\delta = 0.0$). We used the resulting data to generate a standard POD-based model that could be used to test the convergence of the SDPOD modes. Figure 5.10 gives a graphical comparison of the first modes corresponding to the 7 × 7 and 97 × 97 discretizations. As was the case for the POD model in Section 5.1.3, the projected and predicted modes for the SDPOD model cannot be distinguished. Further, it is apparent that the agreement between the SDPOD modes and the standard POD modes improves for the finer discretization. This again reflects the convergence of the streamline-diffusion finite element method. For the very rough 7 × 7 discretization, the SDPOD mode does not match the first reference mode well, though it is significantly better than the first mode from the standard POD-based model. For the 97 × 97 discretization, the SDPOD mode agrees very well with the reference mode, while the standard POD mode, though improved, nevertheless is still far from accurate.

By replacing $\mathbf{u}$ in the error measures defined in Section 5.1.2 with the restriction of the solution derived from the 193×193 mesh to the rougher grids, we can measure the ability of the SDPOD and standard POD-based models to model the reference solution. Figure 5.11 on Page 90 plots the corresponding errors for the 7 × 7 and 97 × 97 discretizations, with the solid black line depicting the reference mode generated from the 193×193 mesh. Even for the rougher discretization, the SDPOD model appears to be nearly as good as the direct projection of the restricted reference snapshots onto the rougher POD basis. The same is true for the 97 × 97 discretization, with the standard POD-based model also much improved, as expected. The results for all discretization levels are reported in Table 5.3, where the superscript indicates that the error measure is now with respect to the reference solution.

5.3 Gradient Information for the Model Control Problem

We begin this section with some notational conventions. Since any element $\mathbf{\Psi}$ from the space H_n spanned by the snapshot ensemble can be written as a linear combination

$$\begin{pmatrix} \sum_{i=1}^{N} \Psi_i \Phi_i(\mathbf{x}) \\ \sum_{i=1}^{N} \Psi_{i+N} \Phi_i(\mathbf{x}) \end{pmatrix}$$

of the finite element basis functions Φ_i, $i = 1, \ldots, N$, it is natural to identify $\mathbf{\Psi}$ with the coefficient vector of its representation in the finite element basis ($\mathbf{\Psi} \in H_n \leftrightarrow \mathbf{\Psi} \in \mathbb{R}^{2N}$), such that for $\mathbf{\Psi}, \mathbf{\Theta} \in H_n$ we can write

$$(\mathbf{\Psi}, \mathbf{\Theta})_H = \mathbf{\Psi} \mathrm{M} \mathbf{\Theta} = (\mathbf{\Psi}, \mathbf{\Theta})_{\mathrm{M}},$$

where $\mathrm{M} \in \mathbb{R}^{2N,2N}$ is the positive definite finite element mass matrix. Using this convention we set $\|\cdot\|_{\mathrm{M}} = (\mathbf{\Psi}, \mathbf{\Theta})_{\mathrm{M}}^{1/2}$.

Now, replacing the state equations in the velocity-tracking problem with either the standard POD-based model of Section 5.1 or the SDPOD model leads to the model control problem

$$\begin{aligned} \min \mathcal{J}(\gamma) = \int_0^T L(\mathbf{y}(\gamma), \gamma, t)\, dt \\ \text{s.t. } \dot{\mathbf{y}} = \phi(\mathbf{y}, \gamma, t), \end{aligned} \tag{5.23}$$

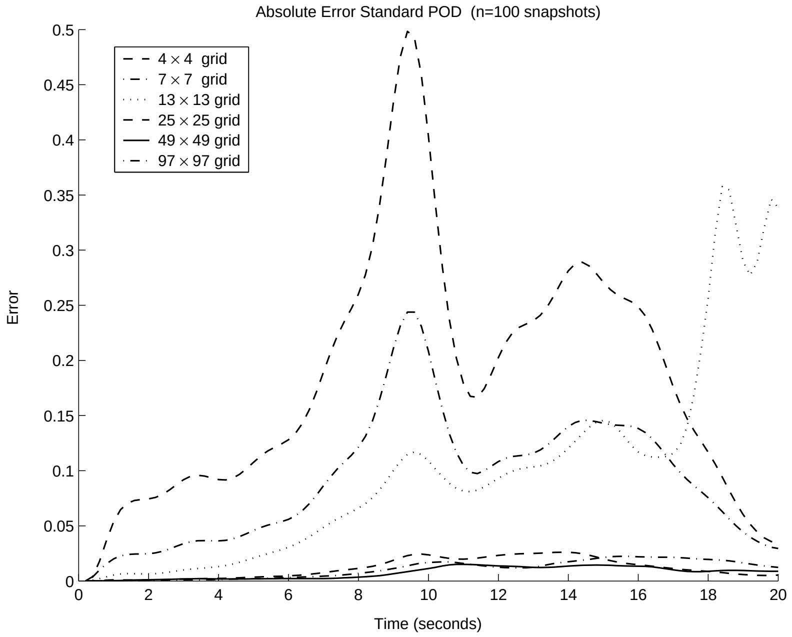


Figure 5.8: Absolute error for standard POD method.

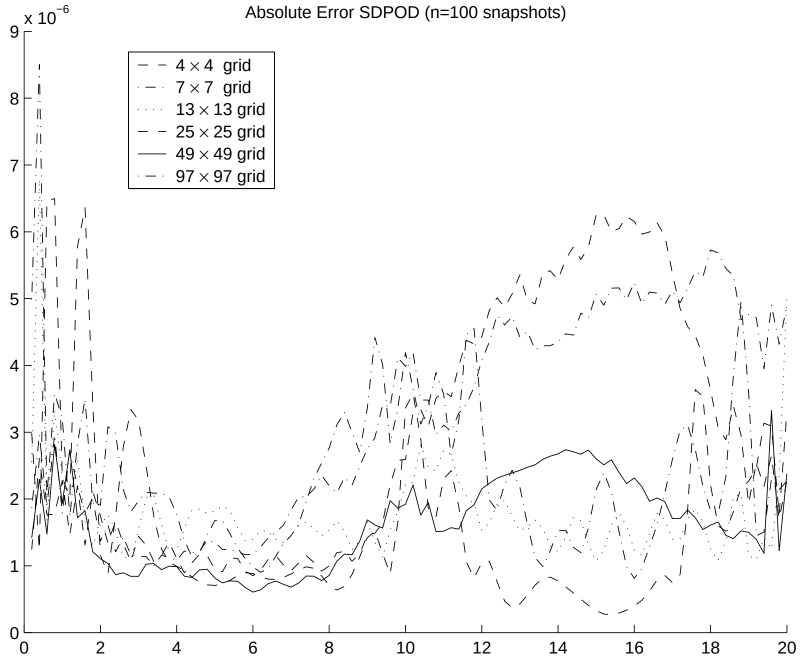
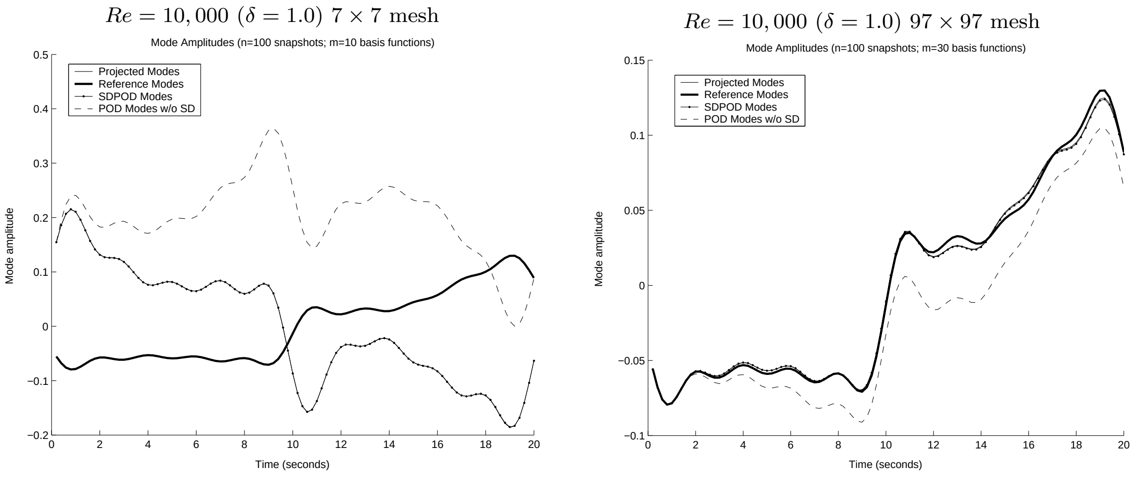


Figure 5.9: Absolute error for the SDPOD method.

Discr.	m	%Energy	E_{PROJ}	E_{SDPOD}	E_{POD}
4×4	5	99.9	8.4037 e-6	9.7677 e-6	1.8121 e-1
7×7	10	99.9	1.1408 e-5	1.3565 e-5	9.1854 e-2
13×13	15	99.9	6.6566 e-6	9.1258 e-6	9.8077 e-2
25×25	20	99.9	4.5393 e-6	7.4863 e-6	1.1682 e-2
49×49	24	99.9	3.6355 e-6	5.0878 e-6	7.4593 e-3
97×97	30	99.9	6.4540 e-6	9.2616 e-6	1.0731 e-2

Table 5.2: Comparison of model accuracy at various discretizations with streamline diffusion parameter $\delta = 1.0$ at $Re = 10,000$.


$Re = 10,000$ ($\delta = 1.0$) 7×7 mesh

$Re = 10,000$ ($\delta = 1.0$) 97×97 mesh

Figure 5.10: Comparison to a reference solution.

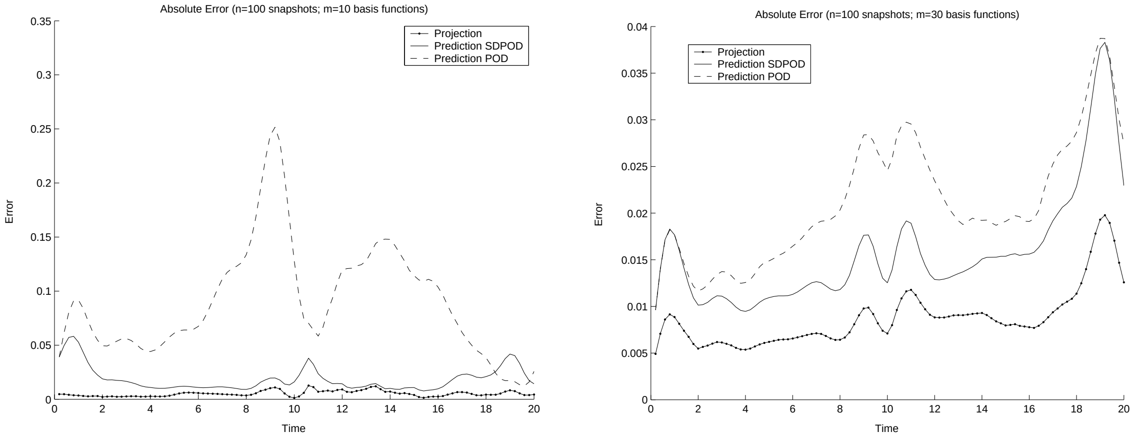


Figure 5.11: Absolute error for standard POD and SDPOD methods for the 7×7 (left) and 97×97 (right) meshes.

Discr.	m	%Energy	E^{ref}_{PROJ}	E^{ref}_{SDPOD}	E^{ref}_{POD}
4×4	5	99.9	4.0061 e-3	1.5083 e-2	1.8089 e-1
7×7	10	99.9	5.1876 e-3	1.8422 e-2	9.0022 e-2
13×13	15	99.9	8.6752 e-3	1.6988 e-2	1.0369 e-1
25×25	20	99.9	8.5888 e-3	1.3514 e-2	2.0268 e-2
49×49	24	99.9	9.2043 e-3	1.3718 e-2	2.3441 e-2
97×97	30	99.9	8.7334 e-3	1.5802 e-2	2.1182 e-2

Table 5.3: Comparison of model to reference solution at various discretizations with streamline diffusion parameter $\delta = 1.0$ at $Re = 10,000$.

where

$$L(\mathbf{y}, \gamma, t) = \frac{1}{2} \|\mathbf{u}_n + \gamma(t)(\mathbf{u}_c + \mathbf{\Phi}\mathcal{M}_c) + \mathbf{\Phi}\mathbf{y}(t) - \mathbf{u}_d\|^2_{\text{M}},$$

$\phi(\mathbf{y}, \gamma, t)$ is given by the right-hand side of (5.7) or (5.19) and $\mathbf{\Phi}$ is the matrix with columns consisting of the finite element coefficient vectors of the POD basis functions. We have suppressed the regularization term for the boundary control here in order to simplify notation and discussion.

5.3.1 Gradient Information via the Adjoint Method

The degrees of freedom in the discretized version of (5.23) are equal to the number of snapshots, making it advantageous to acquire gradient information using the adjoint method.

Since L and ϕ are continuously differentiable in $\mathbf{y}$ and γ, the gradient of $\mathcal{J}(\gamma)$ in $L^2([0, T])$ is given by

$$(\nabla\mathcal{J}(\gamma)(t)) = p(t)\phi_\gamma(\mathbf{y}(t), \gamma(t), t) + L_\gamma(\mathbf{y}(t), \gamma(t), t)$$

where the adjoint variable p satisfies the terminal-value problem on $[0, T]$

$$\begin{aligned} -\dot{p}(t) &= p(t)\phi_{\mathbf{y}}(\mathbf{y}(t), \gamma(t), t) + L_{\mathbf{y}}(\mathbf{y}(t), \gamma(t), t) \\ p(T) &= 0. \end{aligned}$$

The derivation of $L_{\mathbf{y}}$, L_γ, $\phi_{\mathbf{y}}$ and ϕ_γ in terms of the POD basis is given in Appendix A.3.

5.3.2 Numerical Example for the Adjoint Derivatives

We generated target solutions $\mathbf{u}_d$ of the driven cavity problem at Reynolds numbers of $Re = 400$ and $Re = 10,000$ on a 97×97 mesh using the velocity profile $\gamma(t)$ of (5.17), setting $\delta = 0.0$ at $Re = 400$ and $\delta = 1.0$ at $Re = 10,000$, and taking 100 snapshots at $Re = 400$ and 200 snapshots at $Re = 10,000$. We used the velocity profile $\gamma(t) \equiv 1.0$ to generate an initial iterate for both problems and calculated gradient information for (5.23) using both the adjoint method and finite differences. The resulting gradients are compared in Figure 5.12 on Page 92. The analytical (adjoint) derivatives show excellent agreement with the finite differences, but required much less computation time. The adjoint derivatives also appear to be more stable for $Re = 10,000$.

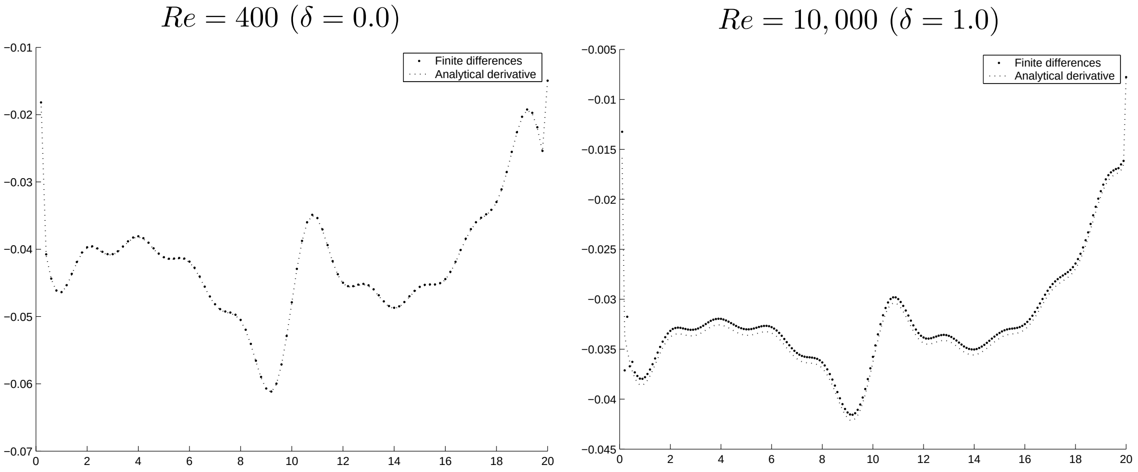


Figure 5.12: Comparison of gradient computed using the adjoint method with the gradient computed using finite differences.

5.4 Some Remarks on the SDPOD Model

We have seen that stabilization methods for the convective term in the Navier-Stokes equations, which are necessary for obtaining a numerical solution at higher Reynolds numbers, can lead to POD basis functions that are incompatible with the standard POD-based model. The SDPOD method circumvents this difficulty by incorporating the stabilization into the reduced-order model, resulting in increased compatibility between the model and the high-order Navier-Stokes solver. Having overcome the model compatibility difficulties we now have a reduced-order method, the SDPOD method, that is suitable for recursive multilevel reduced-order optimization.

We also wish to emphasize that the idea behind SDPOD is not limited to the situation we examine in our model problem. Note for instance that the model could easily be extended to include the jump terms of (3.50). The same is true of the modified test function (3.30), which yields a Galerkin least squares method (cf. [101]).

Chapter 6

The Trust-Region SDPOD Method

In this chapter we seek to apply the methods developed in the earlier chapters to some concrete optimal control problems at Reynolds numbers of $Re = 400$ and $Re = 10,000$. As was discussed at the beginning of Chapter 5, the fidelity of POD-based models deteriorates during the iterative optimization process, making it necessary to update the model periodically. In order to avoid the computational expense of unnecessary updates, it is desirable to have some sort of systematic procedure for determining when the current POD-based model should be rejected and a new model computed. Furthermore, using POD-based models generated from coarse finite element meshes, such as those described in Chapter 5, amplifies the need for such a procedure, since – as indicated by the numerical results of Chapters 4 and 5 – the stability properties of the POD basis may vary from one mesh refinement level to the next, especially at higher Reynolds numbers.

The method of choice in [41] was the trust-region approach, where the method is initiated with a high-order solution of the state equations using some initial control $\mathbf{g}_0$. A POD-based model derived from the high-order solution is then used to solve the optimal control problem, generating a potential optimal control $\mathbf{g}_{\text{new}}$, with the set of admissible controls limited to some "trusted" neighborhood $\|\mathbf{g}_{\text{new}} - \mathbf{g}_0\| \leq \triangle_0$ of $\mathbf{g}_0$, where $\triangle_0$ is known as the *trust-region radius*. A new high-order solution is subsequently generated using the updated control, and the actual decease in the cost functional $\mathcal{J}(\mathbf{g}_0) - \mathcal{J}(\mathbf{g}_{\text{new}})$ achieved using the high-order solver is compared to the decrease $\hat{\mathcal{J}}(\mathbf{g}_0) - \hat{\mathcal{J}}(\mathbf{g}_{\text{new}})$ predicted by the model. If the ratio

$$\rho = \frac{\mathcal{J}(\mathbf{g}_0) - \mathcal{J}(\mathbf{g}_{\text{new}})}{\hat{\mathcal{J}}(\mathbf{g}_0) - \hat{\mathcal{J}}(\mathbf{g}_{\text{new}})}$$

is sufficiently large, the trust-region radius is decreased, left constant or increased depending on the size of ρ, a new POD-based model is generated using the updated high-order solution of the state equation, and the control problem is resolved using the new POD-based model. If ρ is too small, the new high-order solution is rejected, the trust-region radius is decreased and optimal control problem is resolved using the original POD-based model. The process continues until convergence to a local stationary point is achieved.

This approach is attractive; however, it still requires repeated high-order solution of the state equations. In this spirit, we propose the use of recursive trust-region methods in combination with the SDPOD method introduced in Chapter 5, with the objective of significantly reducing the computational effort required during the solution of the optimal control problem while ensuring convergence of the method. Note that the use of the SDPOD method is required to ensure the POD method can be used at all at lower mesh refinement levels (see Chapter 5).

In Section 6.1 we review a recursive multilevel trust-region method recently suggested and analyzed by Gratton et al. [57]. The method is constructed using quadratic model functions, so the method and the accompanying theoretical analysis are not directly applicable to POD-based model functions. Nevertheless, since nonrecursive trust-region methods have successfully been adapted to more general model functions, including POD-based models (cf. [26, 35, 41, 138]), we are hopeful that the theoretical results from the recursive procedure can be applied to the SDPOD-based models as well (see also [19, 25, 27, 87, 88]).

In Section 6.2 we solve the velocity tracking problem at Reynolds numbers of $Re = 400$ and $Re = 10,000$ using SDPOD-based models and a modified version of trust-region methodology of Section 6.1.

6.1 A Recursive Trust-Region Method for Multilevel Optimization

In the following discussion we borrow heavily from the first part of the paper by Gratton et al. [57]. Consider the solution of the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} f(x), \tag{6.1}$$

where f is a twice-continuously differentiable objective function which maps $\mathbb{R}^n$ into $\mathbb{R}$ and is bounded below. Classical trust-region methods are iterative and, given an initial point x_0, produce a sequence $\{x_k\}$ of iterates (hopefully) converging to a local stationary point for the problem; that is, to a point where $g(x) := \nabla_x f(x) = 0$. At each iterate x_k, classical trust-region methods build a model $m_k(x_k + s)$ of $f(x_k + s)$, which is assumed to be an adequate approximation of $f(x)$ in a "trust-region," defined as a sphere of radius $\triangle_k > 0$ centered at x_k. A step s_k is then computed that sufficiently reduces this model within the trust-region. The objective function is computed at the trial point $x_k + s_k$ and this trial point is accepted as the next iterate if and only if the ratio

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$$

is larger than a small positive constant η_1. The value of the radius is then deceased if the trial point was rejected, and increased or left unchanged if ρ_k is sufficiently large. In many practical trust-region algorithms, the model $m_k(x_k+s)$ is quadratic and takes the form

$$m_k(x_k+s) = f(x_k) + (g_k, s) + \frac{1}{2}(s, H_k s), \tag{6.2}$$

where $g_k := \nabla_x f(x_k)$, H_k is a symmetric $n \times n$ approximation of $\nabla_{xx} f(x_k)$ and $(\cdot,\cdot)$ is the Euclidean inner product. Obtaining a sufficient decrease on this model then amounts to (approximately) solving

$$\min_{\|s\| \leq \triangle_k} m_k(x_k+s) = \min_{\|s\| \leq \triangle_k} f(x_k) + (g_k, s) + \frac{1}{2}(s, H_k s), \tag{6.3}$$

where $\|\cdot\|$ is the Euclidean norm. Such methods provably converge to first-order critical points whenever the sequence $\{\|H_k\|\}$ is uniformly bounded above, i.e., when there is a constant $\kappa_H \geq 1$ such that $1+\|H_k\| < \kappa_H$ for all k. A comprehensive discussion of classical trust-region methods with quadratic model functions can be found in Conn et al. [35].

The impetus for the recursive trust-region approach is the desire to exploit alternative simplified expressions of the objective function that may exist. Specifically, assume knowledge of a collection of functions $\{f_i\}_{i=0}^{r}$, where each f_i is a twice continuously differentiable function from $\mathbb{R}^{n_i}$ to $\mathbb{R}$ ($n_i \geq n_{i-1}$), such that $n_r = n$ and $f_r(x) = f(x)$ for all $x \in \mathbb{R}^n$. Assume also that for each $i = 1, \ldots, r$, f_i is "more costly" to minimize than f_{i-1}. Note that in our case this is due to differences in the underlying mesh refinement levels; correspondingly, a particular i will be referred to as a *level*, and the first subscript i in all subsequent subscripted symbols will denote a quantity corresponding to the i-th level.

In order to establish an exploitable relationship between the functions f_{i-1} and f_i, assume that for each $i = 1, \ldots, r$, there exists a full-rank linear operator R_i from $\mathbb{R}^{n_i}$ into $\mathbb{R}^{n_{i-1}}$ (the restriction) and another full-rank operator P_i from $\mathbb{R}^{n_{i-1}}$ into R^{n_i} (the prolongation) such that

$$P_i = R_i^T. \tag{6.4}$$

The idea is then to use f_{r-1} to construct an alternative model h_{r-1} for $f_r = f$ in the neighborhood of the current iterate that is cheaper to solve than (6.2), and to use this alternative model to define the step in the trust-region algorithm whenever possible. If more than two levels are available ($r > 1$), this can be done recursively, the approximation process starting at level 0, where (6.2) is always used. For notational purposes, variables are indexed with a double subscript i, k. The first, i, is the level index ($0 \leq i \leq r$) and the second, k, the index of the current iteration *within level i, and is reset to* 0 *each time level i is entered.*

Consider now some iteration k at level i (with current iterate $x_{i,k}$) and suppose that one decides to use the lower level model h_{i-1} based on f_{i-1} to compute a step. The first task is to restrict $x_{i,k}$ to create the starting iterate $x_{i-1,0}$ at level $i-1$, that is

$$x_{i-1,0} = R_i x_{i,k}. \tag{6.5}$$

The lower level model is defined as the function

$$h_{i-1}(x_{i-1,0}+s_{i-1}) := f_{i-1}(x_{i-1,0}+s_{i-1}) + (v_{i-1}, s_{i-1}), \tag{6.6}$$

where

$$v_{i-1} = R_i g_{i,k} - \nabla_{x_{i-1}} f_{i-1}(x_{i-1,0}) \tag{6.7}$$

with $g_{i,k} := \nabla_{x_i} h_i(x_{i,k})$. By convention, set $v_r := 0$, so that

$$h_r(x_{r,0}+s_r) = f_r(x_{r,0}+s_r) = f(x_0+s) \text{ and } g_{r,k} = \nabla_{x_r} h_r(x_{r,k}) = \nabla_x f(x_k) = g_k.$$

The function h_i therefore corresponds to a modification of f_i by a linear term that enforces the relation

$$g_{i-1,0} = \nabla_{x_{i-1}} h_{i-1}(x_{i-1,0}) = R_i g_{i,k}. \tag{6.8}$$

Because the technique involves minimizing h_i at level i, this function must be bounded below. This assumption is therefore made for every $i = 0, \dots, r$. The first order modification (6.6) serves to ensure that the first-order behaviors of h_i and h_{i-1} are coherent in a neighborhood of $x_{i,k}$ and $x_{i-1,0}$ respectively.

When entering some level i then, one wishes to (locally) minimize h_i starting from $x_{i,0}$. At iteration k of this minimization, one first chooses between the models $h_{i-1}(x_{i-1,0}+s_{i-1})$ and

$$m_{i,k}(x_{i,k}+s_i) = h_i(x_{i,k}) + (g_{i,k}, s_i) + \frac{1}{2}(s_i, H_{i,k} s_i), \tag{6.9}$$

where the latter is a truncated Taylor series, in which $H_{i,k}$ is a symmetric $n_i \times n_i$ approximation to the second derivatives of h_i at $x_{i,k}$, such that, for some $\kappa_H \geq 1$

$$1 + \|H_{i,k}\| \leq \kappa_H \tag{6.10}$$

for all k and all $i = 0, \dots, r$. Once the model is chosen, a step $s_{i,k}$ is computed that generates a "sufficient decrease" on this model within a trust-region defined by

$$\mathcal{B}_{i,k} := \{s_i \mid \|s_i\|_i \leq \triangle_{i,k}\}, \tag{6.11}$$

for some trust-region radius $\triangle_{i,k} > 0$, where the norm $\|\cdot\|_i$ is level-dependent.

The "sufficient decrease" of the model $m_{i,k}$ is understood here in its usual meaning for trust-region methods, which is to say that $s_{i,k}$ is such that

$$m_{i,k}(x_{i,k}) - m_{i,k}(x_{i,k}+s_{i,k}) \geq \kappa_{red} \|g_{i,k}\| \min\left[\frac{\|g_{i,k}\|}{1+\|H_{i,k}\|}, \triangle_{i,k}\right] \tag{6.12}$$

for some constant $\kappa_{red} \in (0,1)$. This condition is known as the "Cauchy point" condition (see Conn et al. [35, Chapter 7]).

Finally, it may happen that $g_{i,k}$ lies in the nullspace of R_i, so that the current iterate appears to be first-order critical for h_{i-1} in $\mathbb{R}^{n_{i-1}}$ while it is not for h_i in $\mathbb{R}^{n_i}$. This can be avoided by requiring

$$\|R_i g_{i,k}\| \geq \kappa_g \|g_{i,k}\| \text{ and } \|R_i g_{i,k}\| \geq \epsilon^g_{i-1} \tag{6.13}$$

for some constant $\kappa_g \in (0,1)$, where $\epsilon_{i-1}^g \in (0,1)$ is a measure of the first-order criticality that is judged sufficient at level $i-1$.

To begin the algorithm, an initial trust-region radius for each level $\triangle_i^s > 0$ must be defined, as well as level-dependent gradient norm tolerances $\epsilon_i^g \in (0,1)$ and trust-region tolerances $\epsilon_i^{\triangle} \in (0,1)$ for $i = 0, \dots r$. The algorithm's initial data consists of the level index i $(0 < i < r)$, a starting point $x_{i,0}$, the gradient $g_{i,0}$ at this point, the radius $\triangle_{i+1}$ of the level $i+1$ trust-region (by convention, $\triangle_{r+1} := \infty$), the tolerances ϵ_i^g and $\epsilon_i^{\triangle}$, and constants η_1, η_2, γ_1 and γ_2 satisfying the conditions

$$0 < \eta_1 \leq \eta_2 < 1, \text{ and } 0 < \gamma_1 \leq \gamma_2 < 1.$$

We can now state the following recursive multilevel trust-region algorithm from Gratton et al. [57, Algorithm 2.1].

Algorithm 6.1. ***RMTR****(i, $x_{i,0}$, $g_{i,0}$, $\triangle_{i+1}$, ϵ_i^g, $\epsilon_i^{\triangle}$)*

Step 0: Initialization.
Compute $v_i = g_{i,0} - \nabla_{x_i} f_i(x_{i,0})$ and $h_i(x_{i,0})$. Set $\triangle_{i,0} = \min[\triangle_i^s, \triangle_{i+1}]$ and $k = 0$.

Step 1: Model choice.
If $i = 0$ or if (6.13) fails, go to Step 3. Otherwise, choose to go to Step 2 (recursive step) or to Step 3 (Taylor step).

Step 2: Recursive step computation.
Call Algorithm RMTR($i-1$, $R_i x_{i,k}$, $R_i g_{i,k}$, $\triangle_{i,k}$, ϵ_{i-1}^g, $\epsilon_{i-1}^{\triangle}$) yielding an approximate solution $x_{i-1,*}$. Then define $s_{i,k} = P_i(x_{i-1,*} - R_i x_{i,k})$, set $\delta_{i,k} = h_{i-1}(R_i x_{i,k}) - h_{i-1}(x_{i-1,*})$ and go to Step 4.

Step3: Taylor step computation.
Choose $H_{i,k}$ in view of (6.10) and compute a step $s_{i,k} \in \mathbb{R}^{n_i}$ that sufficiently reduces the model $m_{i,k}$ (given by (6.9)) in the sense of (6.12) and such that $\|s_{i,k}\|_i \leq \triangle_{i,k}$. Set $\delta_{i,k} = m_{i,k}(x_{i,k}) - m_{i,k}(x_{i,k} + s_{i,k})$ and go to Step 4.

Step 4: Acceptance of the trial point.
Compute $h_i(x_{i,k} + s_{i,k})$ and define

$$\rho_{i,k} = \frac{h_i(x_{i,k}) - h_i(x_{i,k} + s_{i,k})}{\delta_{i,k}}. \tag{6.14}$$

If $\rho_{i,k} \geq \eta_1$ then define $x_{i,k+1} = x_{i,k} + s_{i,k}$; otherwise define $x_{i,k+1} = x_{i,k}$.

Step 5: Termination.
Compute $g_{i,k+1}$. If $\|g_{i,k+1}\| < \epsilon_i^g$ or $\|x_{i,k+1} - x_{i,0}\|_i > (1 - \epsilon_i^{\triangle})\triangle_{i+1}$, then return with the approximate solution $x_{i,*} = x_{i,k+1}$.

Step 6: Trust-region radius update.

Set

$$\triangle_{i,k}^{+} = \begin{cases} [\triangle_{i,k}, +\infty) & \textit{if } \rho_{i,k} \geq \eta_2, \\ [\gamma_2 \triangle_{i,k}, \triangle_{i,k}] & \textit{if } \rho_{i,k} \in [\eta_1, \eta_2), \\ [\gamma_1 \triangle_{i,k}, \gamma_2 \triangle_{i,k}] & \textit{if } \rho_{i,k} < \eta_1. \end{cases} \tag{6.15}$$

and

$$\triangle_{i,k+1} = \min \left[\triangle_{i,k}^{+}, \triangle_{i+1} - \|x_{i,k+1} - x_{i,0}\|_i \right]. \tag{6.16}$$

Increment k by one and go to Step 1.

Application to SDPOD-Based Models

We wish to consider the application of the above methodology to SDPOD-based models. The application of nonrecursive trust-region methods to non-quadratic model functions is well-established in the literature (cf. [26, 41, 138]), so we will concentrate on issues arising from the recursion in Algorithm 6.1.

To begin with, we note that the restriction and prolongation operators (6.5) take on a different context in the SDPOD approach. The restriction and prolongation of iterates and gradients in (6.5), (6.7) and (6.8) are not directly useful since we are optimizing with respect to the boundary velocity γ along the top of the driven cavity using constant temporal discretization. Nevertheless, the gradients of the SDPOD-based model functions will differ at different mesh refinement levels, so that some mechanism, similar to that of (6.7), will be required to ensure coherence of the gradient information between models at different levels. In addition, this difficulty is complicated by the fact that the conditions $f(x_k) = m(x_k)$ and $\nabla_x f(x_k) = \nabla_x m(x_k)$, which are trivially fulfilled for quadratic model functions, no longer hold for general model functions. This is usually handled by requiring some sort of asymptotic consistency condition (cf. [19, 26, 41, 138]). We note that if gradient information is available, gradient accuracy can sometimes be maintained using so-called sensitivity-based scaling ([4, 29, 41]).

The usual approximation methods used for quadratic model functions in Step 3 of Algorithm 6.1 are not applicable to general model functions. Instead, this step must be replaced with a more general approximation that guarantees a so-called ”sufficient decrease.” For nonrecursive trust-region methods, this can be accomplished by the step determination algorithm proposed by Toint [138].

The rest of the algorithm should be applicable as is. Nevertheless, for our numeric investigations in the next section we have chosen to allow the trust-region radius $\triangle_i$ at level i to exceed $\triangle_{i+1}$, resetting the radius when moving from level i to $i+1$. This makes the second termination condition of Step 5 superfluous.

For the choice of the trust-region parameters, Gratton et al. [57] suggest

$$\eta_1 = 0.01, \quad \eta_2 = 0.95, \quad \gamma_1 = 0.05 \text{ and } \gamma_2 = 0.25,$$

and these are the choices we use in Section 6.2 (see also Gould et al. [54]). For the trust-region radius update (6.15), we choose

$$\triangle_{i,k}^{+} = \begin{cases} 2\triangle_{i,k} & \text{if } \rho_{i,k} \geq \eta_2, \\ \triangle_{i,k} & \text{if } \rho_{i,k} \in [\eta_n, \eta_2), \\ \frac{\gamma_1+\gamma_2}{2}\triangle_{i,k} & \text{if } \rho_{i,k} < \eta_1. \end{cases} \tag{6.17}$$

Remark 6.2. We are aware that it would also be possible to apply here a variation of the well-known continuation method of computational fluid dynamics (cf. [36]); that is, solving the system on coarser meshes at a Reynolds number that doesn't require stabilization, then repeatedly increasing the Reynolds number in tandem with the mesh refinement, thus circumventing the need for stabilization altogether. This idea holds promise for future research and should be pursued; however, one must keep in mind that the method adds an additional source of error, since one must now contend with both the coarseness of the mesh and the use of the incorrect Reynolds number. Moreover, for problems at interesting Reynolds numbers one will eventually need some sort of stabilization if unmanageably large algebraic systems are to be avoided. In this case, Remark 5.2 applies.

6.2 Multilevel Optimization with SDPOD-Based Models

We performed numerical testing of the recursive trust-region method as applied to the SDPOD method using data generated from the driven cavity problem at $Re = 400$ and $Re = 10,000$. No stabilization was used for the data generated at $Re = 400$, whereas the streamline diffusion parameter was set to $\delta = 1.0$ for $Re = 10,000$ (see (3.64)). The programm code of the Navier-Stokes solver FEATFLOW (see Section 2.3) was modified to generate the POD basis and the coefficients of ordinary differential equations systems (5.7) and (5.19) resulting from the Galerkin POD projection.

We once again used the discretizations of the previous chapters, mapping them now to the recursion levels as shown in Table 6.1, which also gives the gradient tolerances ϵ_i^g for each optimization level and problem. Note that we are somewhat more "tolerant" with the coarser meshes. The target velocity profile $\mathbf{u}_d$ in the interior of the cavity was generated at level 5 (97×97 mesh) using the target control velocity γ_d from (5.17) and Figure 5.1. The initial control was set to $\gamma_0 \equiv -1.0$, representing a steady force in the "wrong" direction along the top of the driven cavity.

Level (i)	Mesh	ϵ_i^g ($Re = 400$)	ϵ_i^g ($Re = 10,000$)
0	4×4	$3.0e-3$	$1.0e-1$
1	7×7	$3.0e-3$	$1.0e-1$
2	13×13	$3.0e-3$	$1.0e-1$
3	25×25	$3.0e-3$	$1.0e-1$
4	49×49	$1.0e-3$	$1.0e-3$
5	97×97	$1.0e-3$	$1.0e-3$

Table 6.1: Mesh levels and gradient tolerances for $Re = 400$ and $Re = 10,000$.

We used MATLAB for the optimization of the SDPOD-based models. The ODE systems were solved using the stiff ODE solver *ode15s* from the MATLAB ODE Suite [109]. Solution time varied depending on the number of basis functions (equivalently, the dimension of the ODE) and the character of the boundary control function. Some typical values are given in Table 6.2 on Page 101. The time in in seconds required to solve the ODE for the model coefficients and the associated adjoint problem for the derivative is given for each mesh level along with the number of POD basis functions used. Clearly, the solution time for the ODE systems grows rapidly with the number of equations; that is, with the number of POD basis functions needed to capture 99.9% of the system energy. Note that the solution time for the adjoint is more significant, not only because the solution time for the adjoint was generally greater, but because the adjoint is solved multiple times at each iteration of the optimization in order to approximate the Hessian. Note also that it is inappropriate to compare these values with the time required for the generation of solutions using the FEATFLOW solver, which is coded in the FORTRAN programming language. The solution time for the ODE systems using an ODE solver implemented in FORTRAN would certainly be significantly smaller than those produced by MATLAB. The optimization itself was accomplished using the medium-scale version of *fmincon* from the MATLAB Optimization Toolbox [34], which uses a sequential quadratic programming method (cf. [14, 112]).

Finally, in the tables and figures of the following discussion we have adapted the notation of Section 6.1 to our velocity tracking problem. The objective function is denoted by f instead of $\mathcal{J}$. At each level i and iteration k, the current solution iterate is denoted by $x_{i,k}$ ($x \leftrightarrow \gamma$), the gradient by $g_{i,k}$, the step by $s_{i,k}$. For convenience, we set

$$x^- = \begin{cases} x_{i,k-1}, & \text{if } k > 1, \\ x_{i-1,k^{max}}, & \text{otherwise,} \end{cases}$$

where k^{max} is the maximum iteration of level $i-1$. The number of POD basis functions used is denoted by m, while t_{NS} is the time in seconds required for the high-order Navier-Stokes solution.

Let us begin with the optimization at $Re = 400$. The results at level 0 are displayed in Figure 6.1 on Page 102 and the first 5 rows of Table 6.3 on Page 103. In the first iteration, the POD-based model is generated from numerical data corresponding to the initial control $\gamma \equiv -1.0$. Four POD basis functions are sufficient to capture 99.9% of the system energy. The first step $s_{i,k}$ is on the boundary of the trust-region; however, since the ratio $\rho_{0,1} \approx 3.42/3.1 \approx 1.1007 > \eta_2 = 0.95$, the trust-region is doubled for the next iteration. Since the boundary velocity in the second iteration is relatively small in the $L^2(0,T)$ norm, only two POD basis functions are required to capture 99.9% of the system energy (there is less energy to capture); however, the increased trust-region radius allows rapid movement toward the optimal solution. At the fifth iteration on level 0, the POD-based model satisfies the gradient constraint upon entering the optimization loop.

	$Re = 400$			$Re = 10,000$		
Level (i)	m	t_{ODE}	t_{ADJ}	m	t_{ODE}	t_{ADJ}
0	4	0.12	1.16	4	0.56	2.23
1	5	0.39	2.12	8	3.67	7.42
2	8	1.01	2.65	11	7.88	9.31
3	11	1.44	7.05	13	10.42	10.96
4	11	1.14	28.62	17	20.79	48.20
5	11	1.69	113.91	24	43.81	349.16

Table 6.2: Some typical values for the solution time of the ODE systems arising from the Galerkin POD projection.

The optimization is terminated, the trust-region radius is reset, and we proceed to level 1. The remainder of the procedure is displayed in Figure 6.2 and Table 6.3 on Page 103 and can certainly be interpreted without additional comment. The final solution on level 5 was superimposed on the optimal solution, so we did not bother displaying it in Figure 6.2.

We also performed the simulation at $Re = 400$ using the standard nonrecursive trust-region procedure. The results are depicted in Figure 6.3 and Table 6.4 on Page 104. Note that we again suppressed the display of the fifth iterate, as it was indistinguishable from the optimal solution. The nonrecursive procedure requires five high-order solutions of the Navier-Stokes system, resulting in a total high-order computation time of approximately 272 minutes. Contrast this to the recursive case with total high-order computation time of 152 minutes. Finally, as can surmised from the values in Table 6.1, the computational effort required for the optimization procedure was also considerably less for the recursive procedure. We have chosen not to report the full results here because the optimization was performed using MATLAB, so as mentioned earlier, the values are not directly comparable to the high-order solution times. By way of example, a typical optimization encompassing 18 iterations and requiring about 5 minutes at level 0 might take more than 80 minutes are level 5.

The results for $Re = 10,000$ are reported in Figure 6.5 and Table 6.5 on Page 105, and Figure 6.5 and Table 6.6 on Page 106. The results are similar in nature to those reported for $Re = 400$.

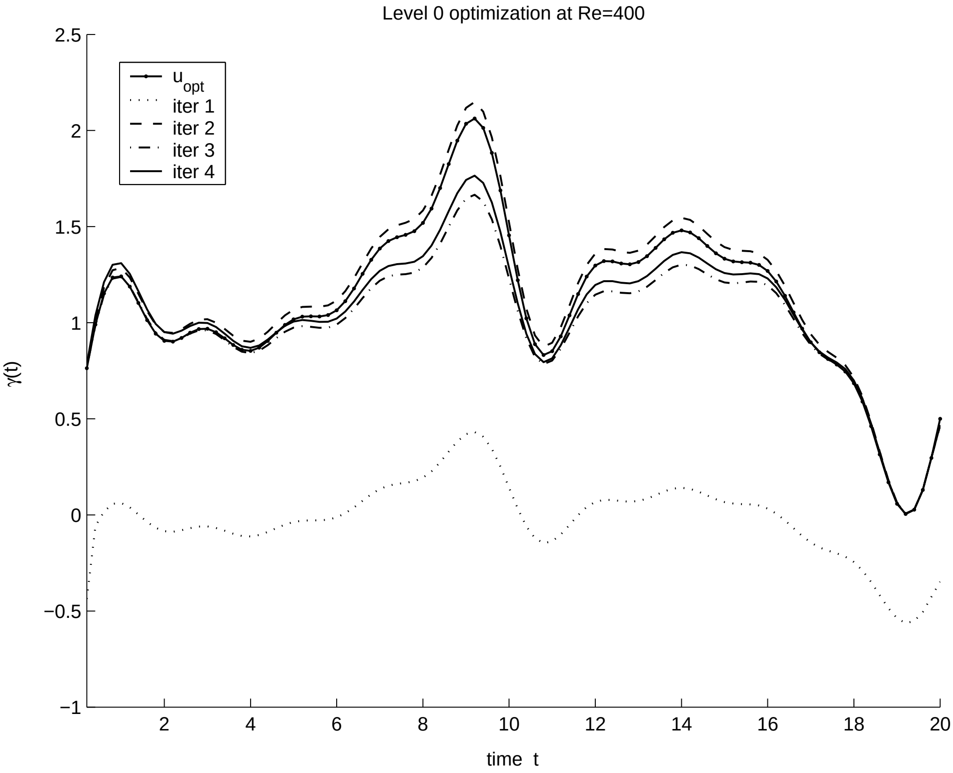


Figure 6.1: The optimization iterates at level 0 for $Re = 400$.

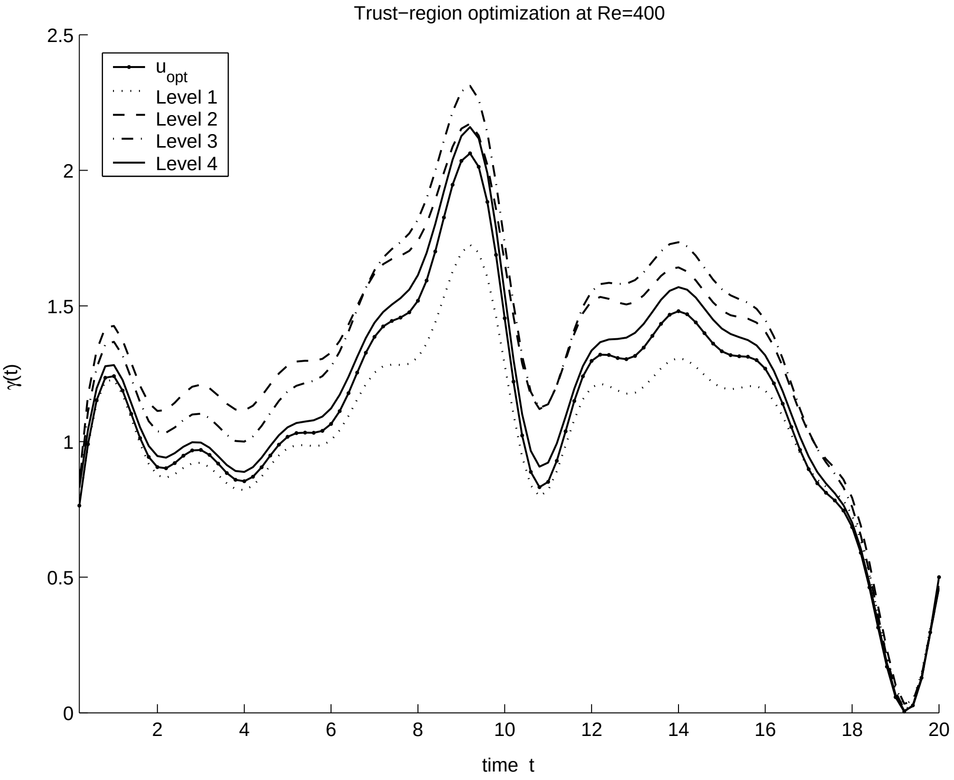


Figure 6.2: The optimization iterates at levels 1 to 4 for $Re = 400$.

i	k	$\triangle_{i,k}$	$\|s_{i,k}\|$	$\|g_{i,k}\|$	$f(x^-)$	$m(x^-)$	$m(x_{i,k})$	$\rho_{i,k}$	m	t_{NS}
0	1	1.00	1.00e-0	2.133e-1	5.23e-0	5.23e-0	2.13e-0	1.1007	4	6
0	2	2.00	1.19e-0	2.983e-4	1.81e-0	1.81e-0	2.45e-1	0.7763	2	6
0	3	2.00	1.86e-1	1.181e-3	5.96e-1	5.96e-1	5.54e-1	0.9642	4	6
0	4	4.00	4.64e-2	1.478e-3	5.57e-1	5.55e-1	5.51e-1	0.4854	3	6
0	5	4.00	—	1.281e-3	—	—	—	—	3	6
1	1	1.00	4.31e-1	3.507e-4	4.88e-1	4.98e-1	4.97e-1	1.9853	5	14
1	2	2.00	—	3.518e-4	—	—	—	—	5	14
2	1	1.00	2.92e-1	2.164e-3	2.50e-1	2.50e-1	2.18e-1	0.9989	8	48
2	2	2.00	—	2.374e-3	—	—	—	—	7	48
3	1	1.00	7.16e-2	1.635e-3	4.73e-2	4.74e-2	4.55e-2	0.7238	11	182
3	2	2.00	—	1.469e-3	—	—	—	—	11	182
4	1	1.00	1.38e-1	6.121e-4	1.84e-2	1.85e-2	3.17e-3	1.0104	11	733
4	2	2.00	8.59e-3	5.212e-4	2.87e-3	3.11e-3	3.06e-3	0.9465	11	730
4	3	2.00	—	5.960e-4	—	—	—	—	11	731
5	1	1.00	5.78e-2	3.907e-4	3.21e-3	3.52e-3	3.43e-4	0.9942	11	3233
5	2	2.00	—	4.563e-4	—	—	—	—	11	3191

Table 6.3: Tabular history of the recursive trust-region optimization process at $Re = 400$.

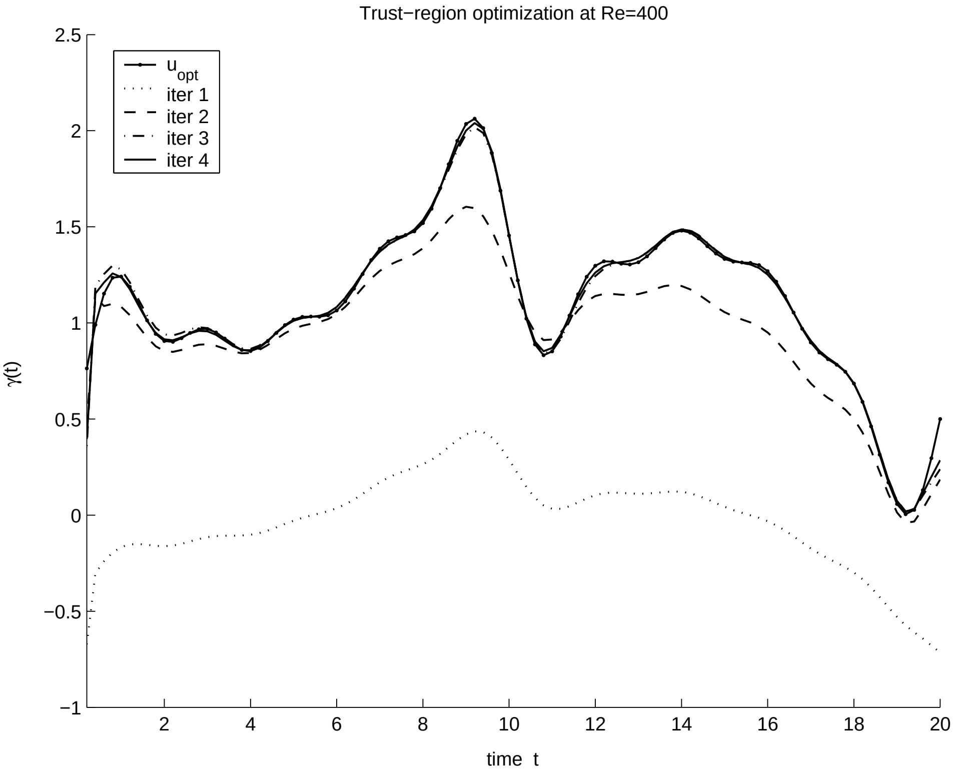


Figure 6.3: The optimization iterates at level 5 for $Re = 400$.

i	k	$\triangle_{i,k}$	$\|s_{i,k}\|$	$\|g_{i,k}\|$	$f(x^-)$	$m(x^-)$	$m(x_{i,k})$	$\rho_{i,k}$	m	t_{NS}
5	1	1.00	1.00e-0	1.146e-1	2.76e-0	2.76e-0	5.06e-1	0.8220	6	3387
5	2	1.00	1.00e-0	1.209e-2	9.07e-1	9.06e-1	2.76e-1	1.4009	7	3214
5	3	2.00	1.89e-1	1.348e-3	2.32e-2	2.30e-2	1.55e-3	1.0649	10	3228
5	4	4.00	1.56e-2	9.057e-4	4.02e-4	6.30e-4	4.98e-4	0.8396	11	3239
5	5	4.00	—	8.446e-4	—	—	—	—	11	3244

Table 6.4: Tabular history of the standard trust-region optimization process at $Re = 400$.

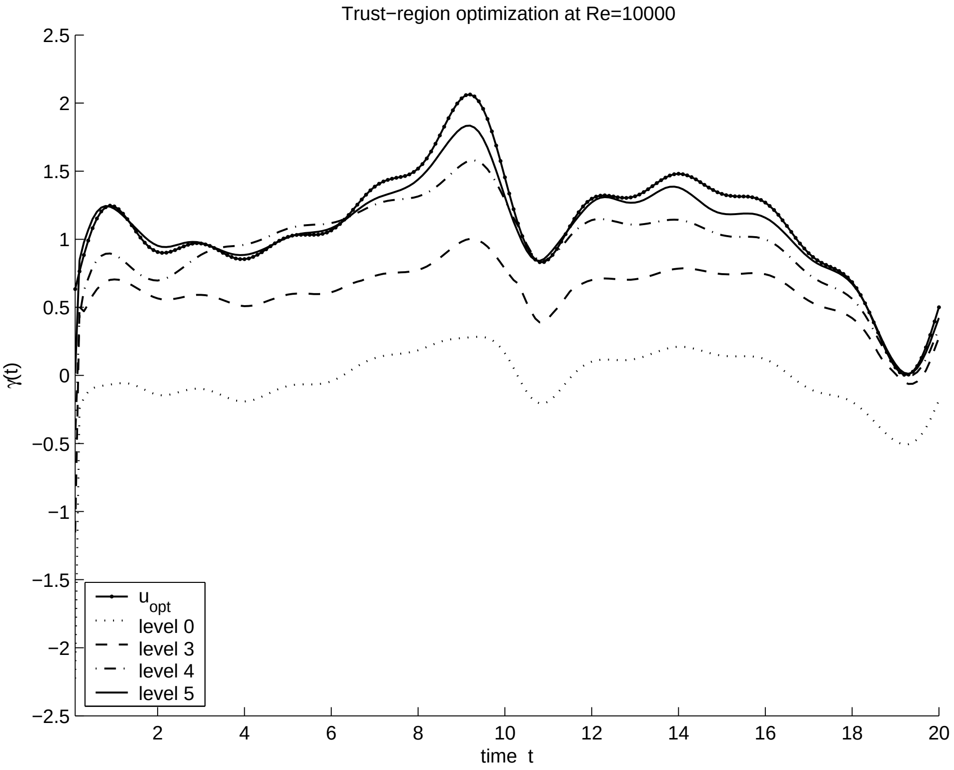


Figure 6.4: The optimization iterates at levels 0, 3, 4 and 5 for $Re = 10,000$.

i	k	$\triangle_{i,k}$	$\|s_{i,k}\|$	$\|g_{i,k}\|$	$f(x^-)$	$m(x^-)$	$m(x_{i,k})$	$\rho_{i,k}$	m	t_{NS}
0	1	1.00	1.00e-0	1.687e-1	5.69e-0	5.69e-0	1.52e-0	0.9990	4	6
0	2	2.00	3.91e-1	5.611e-2	1.52e-0	1.52e-0	1.17e-0	1.2248	3	6
0	3	4.00	3.61e-3	5.089e-2	1.09e-0	1.10e-0	1.09e-1	—	4	6
1	1	1.00	—	2.281e-2	—	—	—	—	8	15
2	1	1.00	—	1.094e-2	—	—	—	—	11	50
3	1	1.00	2.58e-1	3.572e-3	2.34e-1	2.34e-1	2.13e-1	0.3351	13	186
3	2	1.00	—	2.284e-3	—	—	—	—	17	186
4	1	1.00	3.80e-1	1.480e-3	1.25e-1	1.25e-1	1.01e-1	0.7223	17	748
4	2	1.00	—	3.001e-3	—	—	—	—	21	756
5	1	1.00	1.50e-1	5.135e-4	2.28e-2	2.27e-2	1.68e-2	2.4468	28	3193
5	2	2.00	6.32e-2	4.611e-4	8.32e-3	8.24e-3	7.58e-3	3.2435	26	3845
5	3	2.00	—	8.977e-4	—	—	—	—	31	3161

Table 6.5: Tabular history of the recursive trust-region optimization process at $Re = 10,000$.

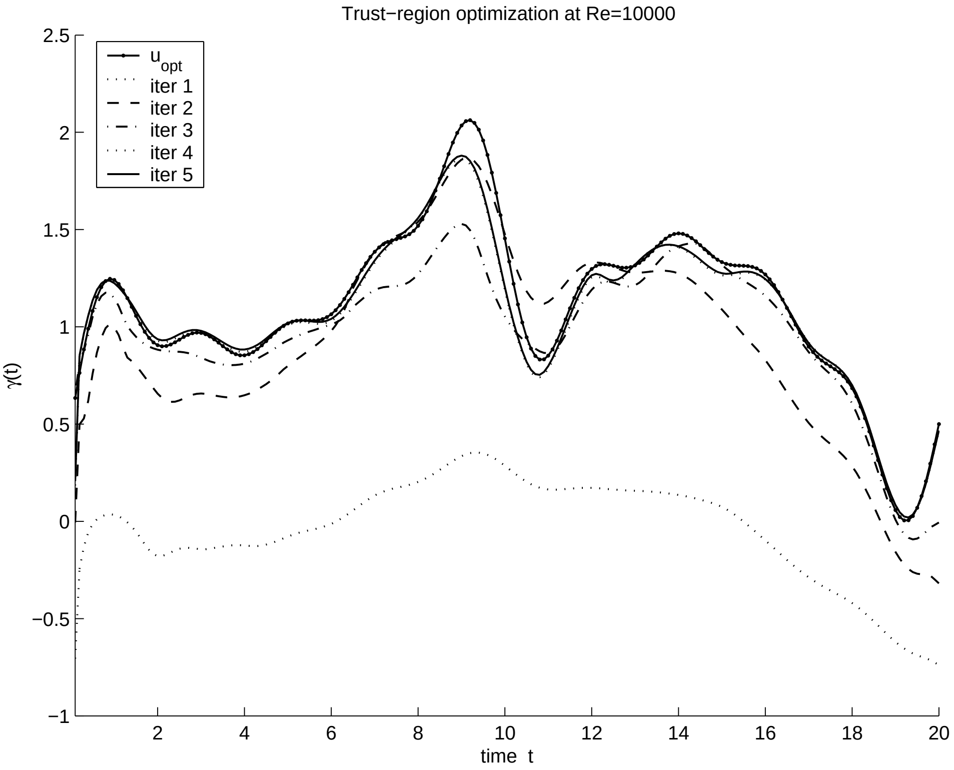


Figure 6.5: The optimization iterates at level 5 for $Re = 10,000$.

i	k	$\triangle_{i,k}$	$\|s_{i,k}\|$	$\|g_{i,k}\|$	$f(x^-)$	$m(x^-)$	$m(x_{i,k})$	$\rho_{i,k}$	m	$t_{\rm NS}$
5	1	1.00	9.97e-1	1.109e-2	5.75e-1	5.74e-1	1.33e-1	0.8141	24	3332
5	2	1.00	9.99e-1	5.841e-3	2.17e-1	2.17e-1	7.58e-2	1.2879	9	3147
5	3	2.00	2.48e-1	1.407e-3	3.54e-2	3.51e-2	8.43e-3	2.9592	29	3154
5	4	4.00	9.65e-3	7.025e-4	1.05e-2	1.04e-2	1.02e-2	27.507	31	3186
5	5	8.00	9.44e-3	6.920e-4	8.62e-3	8.56e-3	8.49e-3	3.1643	31	3142
5	6	16.0	—	6.550e-4	—	—	—	—	31	3171

Table 6.6: Tabular history of the standard trust-region optimization process at $Re = 10,000$.

Conclusion

The numerical solution of optimal control problems, in which the system state is described by one or more partial differential equations, presents enormous computational difficulties when the high-order (e.g., finite element methods) discretization of the system equations leads to large algebraic systems. To alleviate this problem researchers have developed low-order models, such as the proper orthogonal decomposition, that approximate the system equations well, but can be solved with much less computational effort. Since POD-based models are generated from data provided by the high-order numerical solution of the system in question as described in Chapter 4, the fidelity of the model depends on the problem data (boundary and initial conditions, Reynolds number, etc.), so that the model must be periodically reset, requiring renewed high-order solution of the system.

In order reduce the high-order computational effort needed to repeatedly reset the POD-based model, it would be advantageous if one could acquire the information for POD basis generation and augmentation with less computational effort. For data generated by DNS this might be accomplished by extending the POD approach to coarser grids to compute approximate solutions, which can then be used as starting points for optimization on finer grids. This idea is appealing; however, we saw in Chapter 5 that the high-order stabilization required for solving certain partial differential equations of interest can lead to the generation of POD basis functions that are incompatible with standard POD-based models. As a remedy, we introduced the streamline diffusion POD method, which in essence adds the high-order stabilization to the POD-based model, resulting in a model that is fully compatible with the POD basis.

In Chapter 6 we introduced the idea of embedding the SDPOD-based models into a multilevel recursive trust-region procedure. The initial numerical results indicate that this method has the potential to significantly reduce the computational effort required for solving optimal control problems, while maintaining the guaranteed convergence of trust-region methods.

Appendix A

Derivation of the Reduced-Order Models

This appendix provides a detailed derivation of the reduced-order models of Chapter 5. We begin with the standard POD-based model of Section 5.1, then extend the derivation to the SDPOD model of Section 5.2. We assume that the Navier-Stokes problem has been solved numerically and that a set of n snapshots of the flow field, $\mathbf{u}(t_i, \mathbf{x})$, at times t_i, $i = 1, \ldots, n$, were generated, from which a set of $p \leq n$ divergence-free and homogeneous POD basis functions, $\boldsymbol{\Psi}_i$, $i = 1, \ldots, p$, has been extracted as described in Chapter 4. In the case of the two-dimensional Navier-Stokes equations, each basis function takes the form $\boldsymbol{\Psi} = \left(\Psi^{(1)}, \Psi^{(2)}\right)^T$, with $\Psi^{(j)} : \mathbb{R}^2 \mapsto \mathbb{R}$, $j = 1, 2$.

A.1 Derivation of the Standard POD-Based Model

In accordance with (5.4), the POD-based model assumes that the system velocity takes the form of the linear expansion

$$\mathbf{u}(t, \mathbf{x}) = \mathbf{u}_n(\mathbf{x}) + \gamma(t)\mathbf{u}_c(\mathbf{x}) + \sum_{i=1}^{m} y_i(t)\boldsymbol{\Psi}_i(\mathbf{x}), \tag{A.1}$$

where $m \leq p$ is the number of POD basis functions desired for the POD-based model, $\gamma(t)$ is the temporal component of the boundary velocity, $\mathbf{u}_c$ is a reference flow field and $\mathbf{u}_n$ is the average of the n snapshots after subtraction of the reference field.

Our goal in this section is to determine the unknown coefficients $y_i(t)$ in (A.1) by projecting the momentum equation

$$\frac{\partial \mathbf{u}}{\partial t} - \nu \Delta \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p = \mathbf{f} \tag{A.2}$$

onto the POD basis and solving the resulting system of ordinary differential equations. In order to simplify the following discussion, and because our model problem uses only boundary controls, we shall assume $\mathbf{f} = 0$ in (A.2), though a nonzero value can easily be accommodated by the model.

The Galerkin projection of (A.2) onto the POD basis yields the identities

$$\left(\frac{\partial \mathbf{u}}{\partial t}, \boldsymbol{\Psi}_j\right) = -\left(\mathbf{u}\cdot\nabla\mathbf{u}, \boldsymbol{\Psi}_j\right) - \left(\nabla p, \boldsymbol{\Psi}_j\right) + \nu\left(\Delta\mathbf{u}, \boldsymbol{\Psi}_j\right), \tag{A.3}$$

where $\boldsymbol{\Psi}_j$ is the j-th POD basis function, $j = 1, \ldots, m$.

Setting (A.1) into the left-hand side of (A.3) and using the orthogonality of the POD basis, we have

$$\begin{aligned}\left(\frac{\partial \mathbf{u}}{\partial t}, \boldsymbol{\Psi}_j\right) &= \dot{\gamma}(t)\left(\mathbf{u}_c, \boldsymbol{\Psi}_j\right) + \sum_{i=1}^{m} \dot{y}_i(t)\left(\boldsymbol{\Psi}_i, \boldsymbol{\Psi}_j\right) \\ &= \dot{\gamma}(t)\left(\mathbf{u}_c, \boldsymbol{\Psi}_j\right) + \dot{y}_j(t).\end{aligned} \tag{A.4}$$

Moreover, it is easily seen using partial integration and the divergence-free nature of the POD basis that the pressure term $(\nabla p, \boldsymbol{\Psi}_j) = -(p, \nabla\cdot\boldsymbol{\Psi}_j)$ drops out of the right-hand side of (A.3). Finally, using Green's formula we have

$$\left(\Delta\mathbf{u}, \boldsymbol{\Psi}_j\right) = \int_\Gamma \boldsymbol{\Psi}_j \frac{\partial \mathbf{u}}{\partial \mathbf{n}}\, d\mathbf{x} - \left(\nabla\mathbf{u}, \nabla\boldsymbol{\Psi}_j\right). \tag{A.5}$$

Inserting (A.4) and (A.5) into (A.3), and using the homogeneity of POD basis, the Galerkin projection (A.3) can be written as

$$\dot{y}_j(t) = -\nu\left(\nabla\mathbf{u}, \nabla\boldsymbol{\Psi}_j\right) - \dot{\gamma}(t)\left(\mathbf{u}_c, \boldsymbol{\Psi}_j\right) - \left(\mathbf{u}\cdot\nabla\mathbf{u}, \boldsymbol{\Psi}_j\right), \quad j = 1, \ldots, m. \tag{A.6}$$

Expansion of the right-hand side. In order to obtain a true reduced-order model, we must separate the right-hand side of (A.6) into spacial components that can be determined a priori, and temporal components to be determined by solving the resulting system of ordinary differential equations. For the first term $(\nabla\mathbf{u}, \nabla\boldsymbol{\Psi}_j)$ in the right-hand side of (A.6) we obtain

$$\begin{aligned}(\nabla\mathbf{u}, \nabla\boldsymbol{\Psi}_j) &= (\nabla\mathbf{u}_n + \gamma(t)\nabla\mathbf{u}_c + \sum_{i=1}^{m} y_i(t)\nabla\boldsymbol{\Psi}_i, \nabla\boldsymbol{\Psi}_j) \\ &= (\nabla\mathbf{u}_n, \nabla\boldsymbol{\Psi}_j) + \gamma(t)(\nabla\mathbf{u}_c, \nabla\boldsymbol{\Psi}_j) + \sum_{i=1}^{m} y_i(t)(\nabla\boldsymbol{\Psi}_i, \nabla\boldsymbol{\Psi}_j).\end{aligned} \tag{A.7}$$

The term $(\mathbf{u}_c, \boldsymbol{\Psi}_j)$ is already in reduced form, while the term $(\mathbf{u}\cdot\nabla\mathbf{u}, \boldsymbol{\Psi}_j)$ can be reduced as follows:

$$\begin{aligned}&(\mathbf{u}\cdot\nabla\mathbf{u}, \boldsymbol{\Psi}_j) = \\ &\quad = ((\mathbf{u}_n + \gamma(t)\mathbf{u}_c + \sum_{i=1}^{m} y_i(t)\boldsymbol{\Psi}_i)\cdot(\nabla\mathbf{u}_n + \gamma(t)\nabla\mathbf{u}_c + \sum_{k=1}^{m} y_k(t)\nabla\boldsymbol{\Psi}_k), \boldsymbol{\Psi}_j) \\ &\quad = (\mathbf{u}_n\cdot\nabla\mathbf{u}_n, \boldsymbol{\Psi}_j) + \gamma(t)(\mathbf{u}_n\cdot\nabla\mathbf{u}_c, \boldsymbol{\Psi}_j) + \sum_{k=1}^{m} y_k(t)(\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_k, \boldsymbol{\Psi}_j)\end{aligned}$$

$$
\begin{aligned}
&+ \gamma(t)(\mathbf{u}_c \cdot \nabla \mathbf{u}_n, \boldsymbol{\Psi}_j) + \gamma^2(t)(\mathbf{u}_c \cdot \nabla \mathbf{u}_c, \boldsymbol{\Psi}_j) + \gamma(t) \sum_{k=1}^{m} y_k(t)(\mathbf{u}_c \cdot \nabla \boldsymbol{\Psi}_k, \boldsymbol{\Psi}_j) \\
&+ \sum_{i=1}^{m} y_i(t)(\boldsymbol{\Psi}_i \cdot \nabla \mathbf{u}_n, \boldsymbol{\Psi}_j) + \gamma(t) \sum_{i=1}^{m} y_i(t)(\boldsymbol{\Psi}_i \cdot \nabla \mathbf{u}_c, \boldsymbol{\Psi}_j) \\
&+ \sum_{i=1}^{m} \sum_{k=1}^{m} y_i(t) y_k(t)(\boldsymbol{\Psi}_i \cdot \nabla \boldsymbol{\Psi}_k, \boldsymbol{\Psi}_j). \qquad \text{(A.8)}
\end{aligned}
$$

Rearranging and collecting terms, and changing indices where convenient, we obtain

$$
\begin{aligned}
(\mathbf{u} \cdot \nabla \mathbf{u}, \boldsymbol{\Psi}_j) = (\mathbf{u}_n \cdot \nabla \mathbf{u}_n, \boldsymbol{\Psi}_j)& \\
&+ \gamma(t)[(\mathbf{u}_n \cdot \nabla \mathbf{u}_c, \boldsymbol{\Psi}_j) + (\mathbf{u}_c \cdot \nabla \mathbf{u}_n, \boldsymbol{\Psi}_j)] \\
&+ \gamma(t) \sum_{i=1}^{m} y_i(t)[(\mathbf{u}_c \cdot \nabla \boldsymbol{\Psi}_i, \boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla \mathbf{u}_c, \boldsymbol{\Psi}_j)] \\
&+ \gamma^2(t)(\mathbf{u}_c \cdot \nabla \mathbf{u}_c, \boldsymbol{\Psi}_j) \\
&+ \sum_{i=1}^{m} y_i(t)[(\mathbf{u}_n \cdot \nabla \boldsymbol{\Psi}_i, \boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla \mathbf{u}_n, \boldsymbol{\Psi}_j)] \\
&+ \sum_{i=1}^{m} \sum_{k=1}^{m} y_i(t) y_k(t)(\boldsymbol{\Psi}_i \cdot \nabla \boldsymbol{\Psi}_k, \boldsymbol{\Psi}_j). \qquad \text{(A.9)}
\end{aligned}
$$

Substitution of (A.7) and (A.9) into (A.6) now leads to the desired system

$$
\dot{\mathbf{y}}(t) = \mathcal{M}_0 + \gamma(t)\mathcal{M}_1 + \gamma^2(t)\mathcal{M}_2 + \dot{\gamma}(t)\mathcal{M}_c + \mathcal{M}_3\mathbf{y} + \gamma(t)\mathcal{M}_4\mathbf{y} + \mathcal{M}_5(\mathbf{y}, \mathbf{y}) \qquad \text{(A.10)}
$$

of ordinary differential equations, where the vectors $\mathcal{M}_0$, $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_c \in \mathbb{R}^m$, the matrices $\mathcal{M}_3$, $\mathcal{M}_4 \in \mathbb{R}^{m,m}$ and the bilinear term $\mathcal{M}_5(\mathbf{y}, \mathbf{y}) : \mathbb{R}^{m,m} \to \mathbb{R}^m$ are given by

$$
\begin{aligned}
(\mathcal{M}_0)_j &= -[\nu(\nabla \mathbf{u}_n, \nabla \boldsymbol{\Psi}_j) + (\mathbf{u}_n \cdot \nabla \mathbf{u}_n, \boldsymbol{\Psi}_j)], \\
(\mathcal{M}_1)_j &= -[\nu(\nabla \mathbf{u}_c, \nabla \boldsymbol{\Psi}_j) + (\mathbf{u}_n \cdot \nabla \mathbf{u}_c, \boldsymbol{\Psi}_j) + (\mathbf{u}_c \cdot \nabla \mathbf{u}_n, \boldsymbol{\Psi}_j)], \\
(\mathcal{M}_2)_j &= -(\mathbf{u}_c \cdot \nabla \mathbf{u}_c, \boldsymbol{\Psi}_j), \\
(\mathcal{M}_c)_j &= -(\mathbf{u}_c, \boldsymbol{\Psi}_j), \\
(\mathcal{M}_3)_{ji} &= -[\nu(\nabla \boldsymbol{\Psi}_i, \nabla \boldsymbol{\Psi}_j) + (\mathbf{u}_n \cdot \nabla \boldsymbol{\Psi}_i, \boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla \mathbf{u}_n, \boldsymbol{\Psi}_j)], \\
(\mathcal{M}_4)_{ji} &= -[(\mathbf{u}_c \cdot \nabla \boldsymbol{\Psi}_i, \boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla \mathbf{u}_c, \boldsymbol{\Psi}_j)], \\
(\mathcal{M}_5(\mathbf{y}, \mathbf{y}))_j &= -\mathbf{y}^T(t)\mathcal{Q}_j\mathbf{y}(t), \text{ with } (\mathcal{Q}_j)_{ik} = (\boldsymbol{\Psi}_i \cdot \nabla \boldsymbol{\Psi}_k, \boldsymbol{\Psi}_j). \qquad \text{(A.11)}
\end{aligned}
$$

Given initial conditions on the coefficient vector $\mathbf{y} = (y_1, \ldots, y_m)^T$, and a prescribed time history for γ, this system may be integrated forward in time to yield predicted values of $\mathbf{y}$; hence, predicted flow fields.

Initial Conditions for the ODE

It remains to derive initial conditions for the system (A.10). Setting $t = t_0$ in (A.1) and reformulating gives

$$\sum_{i=1}^{m} y_i(t_0)\mathbf{\Psi}_i(\mathbf{x}) = \underbrace{\mathbf{u}(t_0, \mathbf{x})}_{=:\mathbf{u}_{t_0}} - \mathbf{u}_n(\mathbf{x}) - \gamma(t_0)\mathbf{u}_c(\mathbf{x}). \tag{A.12}$$

The Galerkin projection

$$\sum_{i=1}^{m} y_i(t_0)\,(\mathbf{\Psi}_i, \mathbf{\Psi}_j) = (\mathbf{u}_{t_0}, \mathbf{\Psi}_j) - (\mathbf{u}_n, \mathbf{\Psi}_j) - \gamma(t_0)\,(\mathbf{u}_c, \mathbf{\Psi}_j) \qquad j = 1, \ldots, m \tag{A.13}$$

now yields

$$y_j(t_0) = (\mathbf{u}_{t_0}, \mathbf{\Psi}_j) - (\mathbf{u}_n, \mathbf{\Psi}_j) - \gamma(t_0)\,(\mathbf{u}_c, \mathbf{\Psi}_j) \qquad j = 1, \ldots, m. \tag{A.14}$$

Elimination of the first order term.

The term $\dot{\gamma}$ in (A.10) is difficult to handle during optimization with the reduced-order model. To eliminate this term we set

$$\tilde{\mathbf{y}}(t) := \mathbf{y}(t) - \gamma(t)\mathcal{M}_c \tag{A.15}$$

and substitute $\mathbf{y} = \tilde{\mathbf{y}} + \gamma\mathcal{M}_c$ in (A.1) and (A.10). This substitution changes the velocity expansion to

$$\mathbf{u}(t, \mathbf{x}) = \mathbf{u}_n(\mathbf{x}) + \gamma(t)\mathbf{u}_c(\mathbf{x}) + \sum_{i=1}^{m}(\tilde{y}_i(t) + \gamma(t)(\mathcal{M}_c)_i)\mathbf{\Psi}_i(\mathbf{x}). \tag{A.16}$$

Moreover, the substitution (A.15) eliminates the term $\dot{\gamma}(t)\mathcal{M}_c$ in (A.10) leading to the system

$$\begin{aligned}\dot{\tilde{\mathbf{y}}} = \mathcal{M}_0 + \gamma\mathcal{M}_1 + \gamma^2\mathcal{M}_2 + \mathcal{M}_3(\tilde{\mathbf{y}} + \gamma\mathcal{M}_c) \\ + \gamma\mathcal{M}_4(\tilde{\mathbf{y}} + \gamma\mathcal{M}_c) + \mathcal{M}_5((\tilde{\mathbf{y}} + \gamma\mathcal{M}_c), (\tilde{\mathbf{y}} + \gamma\mathcal{M}_c))\end{aligned} \tag{A.17}$$

where the coefficient vector $\tilde{\mathbf{y}}(t)$ can be determined by integrating (A.17) forward in time.

A.2 Derivation of the SDPOD Model

The SDPOD model is derived from the standard POD-based model by adding the term $\delta_T^m(\mathbf{u}\cdot\nabla\mathbf{u}, \mathbf{u}\cdot\nabla\mathbf{\Psi}_j)$ to (A.6). Reduction of this term after substitution of (A.1) is somewhat tedious, but concentrating first on the component $\mathbf{u}\cdot\nabla\mathbf{u}$ using calculations analogous to those used for (A.7) and (A.9) results in the following partial reduction:

$$(\mathbf{u}\cdot\nabla\mathbf{u}, \mathbf{u}\cdot\nabla\mathbf{\Psi}_j) = (\mathbf{u}_n\cdot\nabla\mathbf{u}_n, \mathbf{u}\cdot\nabla\mathbf{\Psi}_j)$$

$$
\begin{aligned}
&+\gamma(t)[(\mathbf{u}_n\cdot\nabla\mathbf{u}_c,\mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j)+(\mathbf{u}_c\cdot\nabla\mathbf{u}_n,\mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j)]\\
&+\gamma(t)\sum_{i=1}^{m}y_i(t)[(\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_i,\mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j)+(\boldsymbol{\Psi}_i\cdot\nabla\mathbf{u}_c,\mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j)]\\
&+\gamma^2(t)(\mathbf{u}_c\cdot\nabla\mathbf{u}_c,\mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j)\\
&+\sum_{i=1}^{m}y_i(t)[(\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_i,\mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j)+(\boldsymbol{\Psi}_i\cdot\nabla\mathbf{u}_n,\mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j)]\\
&+\sum_{i=1}^{m}\sum_{k=1}^{m}y_i(t)y_k(t)(\boldsymbol{\Psi}_i\cdot\nabla\boldsymbol{\Psi}_k,\mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j). \qquad \text{(A.18)}
\end{aligned}
$$

Repetition of the above procedure for $\mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j$ and some rather laborious rearrangement and collection of terms now completes the reduction:

$$
\begin{aligned}
&(\mathbf{u}\cdot\nabla\mathbf{u},\mathbf{u}\cdot\nabla\boldsymbol{\Psi}_j)=\\
&\quad(\mathbf{u}_n\cdot\nabla\mathbf{u}_n,\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_j)\\
&\qquad+\gamma(t)[(\mathbf{u}_n\cdot\nabla\mathbf{u}_n,\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_j)+(\mathbf{u}_n\cdot\nabla\mathbf{u}_c,\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_j)+(\mathbf{u}_c\cdot\nabla\mathbf{u}_n,\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_j)]\\
&\qquad+\gamma^2(t)[(\mathbf{u}_n\cdot\nabla\mathbf{u}_c,\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_j)+(\mathbf{u}_c\cdot\nabla\mathbf{u}_n,\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_j)+(\mathbf{u}_c\cdot\nabla\mathbf{u}_c,\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_j)]\\
&\qquad+\gamma^3(t)(\mathbf{u}_c\cdot\nabla\mathbf{u}_c,\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_j)\\
&\qquad+\sum_{i=1}^{m}y_i(t)[(\mathbf{u}_n\cdot\nabla\mathbf{u}_n,\boldsymbol{\Psi}_i\cdot\nabla\boldsymbol{\Psi}_j)+(\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_i,\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_j)\\
&\qquad\qquad+(\boldsymbol{\Psi}_i\cdot\nabla\mathbf{u}_n,\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_j)]\\
&\qquad+\gamma(t)\sum_{i=1}^{m}y_i(t)[(\mathbf{u}_n\cdot\nabla\mathbf{u}_c,\boldsymbol{\Psi}_i\cdot\nabla\boldsymbol{\Psi}_j)+(\mathbf{u}_c\cdot\nabla\mathbf{u}_n,\boldsymbol{\Psi}_i\cdot\nabla\boldsymbol{\Psi}_j)\\
&\qquad\qquad+(\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_i,\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_j)+(\boldsymbol{\Psi}_i\cdot\nabla\mathbf{u}_c,\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_j)\\
&\qquad\qquad+(\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_i,\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_j)+(\boldsymbol{\Psi}_i\cdot\nabla\mathbf{u}_n,\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_j)]\\
&\qquad+\gamma^2(t)\sum_{i=1}^{m}y_i(t)[(\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_i,\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_j)+(\boldsymbol{\Psi}_i\cdot\nabla\mathbf{u}_c,\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_j)\\
&\qquad\qquad+(\mathbf{u}_c\cdot\nabla\mathbf{u}_c,\boldsymbol{\Psi}_i\cdot\nabla\boldsymbol{\Psi}_j)]\\
&\qquad+\sum_{i=1}^{m}\sum_{k=1}^{m}y_i(t)y_k(t)[(\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_i,\boldsymbol{\Psi}_k\cdot\nabla\boldsymbol{\Psi}_j)+(\boldsymbol{\Psi}_i\cdot\nabla\mathbf{u}_n,\boldsymbol{\Psi}_k\cdot\nabla\boldsymbol{\Psi}_j)\\
&\qquad\qquad+(\boldsymbol{\Psi}_i\cdot\nabla\boldsymbol{\Psi}_k,\mathbf{u}_n\cdot\nabla\boldsymbol{\Psi}_j)]\\
&\qquad+\gamma(t)\sum_{i=1}^{m}\sum_{k=1}^{m}y_i(t)y_k(t)[(\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_i,\boldsymbol{\Psi}_k\cdot\nabla\boldsymbol{\Psi}_j)+(\boldsymbol{\Psi}_i\cdot\nabla\boldsymbol{\Psi}_k,\mathbf{u}_c\cdot\nabla\boldsymbol{\Psi}_j)\\
&\qquad\qquad+(\boldsymbol{\Psi}_i\cdot\nabla\mathbf{u}_c,\boldsymbol{\Psi}_k\cdot\nabla\boldsymbol{\Psi}_j)]\\
&\qquad+\sum_{i=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m}y_i(t)y_k(t)y_l(t)(\boldsymbol{\Psi}_i\cdot\nabla\boldsymbol{\Psi}_k,\boldsymbol{\Psi}_l\cdot\nabla\boldsymbol{\Psi}_j). \qquad \text{(A.19)}
\end{aligned}
$$

After adding the expanded terms to equation (A.10) and still more rearranging, we arrive at the following system of ordinary differential equations:

$$
\begin{aligned}
\dot{\mathbf{y}}(t)=&\ \mathcal{M}_0+\gamma(t)\mathcal{M}_1+\gamma^2(t)\mathcal{M}_2+\dot\gamma(t)\mathcal{M}_c+\mathcal{M}_3\mathbf{y}+\gamma(t)\mathcal{M}_4\mathbf{y}+\mathcal{M}_5(\mathbf{y},\mathbf{y})\\
&+\delta_T^m[\mathcal{M}_0^\delta+\gamma(t)\mathcal{M}_1^\delta+\gamma^2(t)\mathcal{M}_2^\delta+\mathcal{M}_3^\delta\mathbf{y}+\gamma(t)\mathcal{M}_4^\delta\mathbf{y}\\
&\quad+\mathcal{M}_5^\delta(\mathbf{y},\mathbf{y})+\gamma^3(t)\mathcal{M}_6^\delta+\gamma^2\mathcal{M}_7^\delta\mathbf{y}+\gamma(t)\mathcal{M}_8^\delta(\mathbf{y},\mathbf{y})+\mathcal{M}_9^\delta(\mathbf{y},\mathbf{y},\mathbf{y})], \qquad \text{(A.20)}
\end{aligned}
$$

where the new terms, including the vectors $\mathcal{M}_0^\delta$, $\mathcal{M}_1^\delta$, $\mathcal{M}_2^\delta$, $\mathcal{M}_c^\delta$, $\mathcal{M}_6^\delta \in \mathbb{R}^m$, the matrices $\mathcal{M}_3^\delta$, $\mathcal{M}_4^\delta$, $\mathcal{M}_7^\delta \in \mathbb{R}^{m,m}$, the bilinear terms $\mathcal{M}_5^\delta(\mathbf{y},\mathbf{y})$, $\mathcal{M}_8^\delta(\mathbf{y},\mathbf{y}) : \mathbb{R}^{m,m} \to \mathbb{R}^m$ and the trilinear term $\mathcal{M}_9^\delta(\mathbf{y},\mathbf{y},\mathbf{y}) : \mathbb{R}^{m,m,m} \to \mathbb{R}^m$, are given by

$$\begin{aligned}
(\mathcal{M}_0^\delta)_j &= -(\mathbf{u}_n \cdot \nabla\mathbf{u}_n, \mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_j),\\
(\mathcal{M}_1^\delta)_j &= -[(\mathbf{u}_n \cdot \nabla\mathbf{u}_n, \mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_j) + (\mathbf{u}_n \cdot \nabla\mathbf{u}_c, \mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_j) + (\mathbf{u}_c \cdot \nabla\mathbf{u}_n, \mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_j)],\\
(\mathcal{M}_2^\delta)_j &= -[(\mathbf{u}_n \cdot \nabla\mathbf{u}_c, \mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_j) + (\mathbf{u}_c \cdot \nabla\mathbf{u}_n, \mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_j) + (\mathbf{u}_c \cdot \nabla\mathbf{u}_c, \mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_j)],\\
(\mathcal{M}_3^\delta)_{ji} &= -[(\mathbf{u}_n \cdot \nabla\mathbf{u}_n, \boldsymbol{\Psi}_i \cdot \nabla\boldsymbol{\Psi}_j) + (\mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_i, \mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla\mathbf{u}_n, \mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_j)],\\
(\mathcal{M}_4^\delta)_{ji} &= -[(\mathbf{u}_n \cdot \nabla\mathbf{u}_c, \boldsymbol{\Psi}_i \cdot \nabla\boldsymbol{\Psi}_j) + (\mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_i, \mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_j) + (\mathbf{u}_c \cdot \nabla\mathbf{u}_n, \boldsymbol{\Psi}_i \cdot \nabla\boldsymbol{\Psi}_j)]\\
&\qquad + (\mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_i, \mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla\mathbf{u}_n, \mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla\mathbf{u}_c, \mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_j)],\\
(\mathcal{M}_5^\delta)_j &= -\mathbf{y}^T(t)\mathcal{Q}_j^\delta\mathbf{y}(t), \text{ with } (\mathcal{Q}_j^\delta)_{ik} = \qquad\qquad\qquad\qquad\qquad\qquad (A.21)\\
&\qquad [(\mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_i, \boldsymbol{\Psi}_k \cdot \nabla\boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla\mathbf{u}_n, \boldsymbol{\Psi}_k \cdot \nabla\boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla\boldsymbol{\Psi}_k, \mathbf{u}_n \cdot \nabla\boldsymbol{\Psi}_j)],\\
(\mathcal{M}_6^\delta)_j &= -(\mathbf{u}_c \cdot \nabla\mathbf{u}_c, \mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_j),\\
(\mathcal{M}_7^\delta)_{ji} &= -[(\mathbf{u}_c \cdot \nabla\mathbf{u}_c, \boldsymbol{\Psi}_i \cdot \nabla\boldsymbol{\Psi}_j) + (\mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_i, \mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla\mathbf{u}_c, \mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_j)],\\
(\mathcal{M}_8^\delta)_j &= -\mathbf{y}^T(t)\mathcal{R}_j^\delta\mathbf{y}(t), \text{ with } (\mathcal{R}_j^\delta)_{ik} = \qquad\qquad\qquad\qquad\qquad\qquad (A.22)\\
&\qquad [(\mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_i, \boldsymbol{\Psi}_k \cdot \nabla\boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla\boldsymbol{\Psi}_k, \mathbf{u}_c \cdot \nabla\boldsymbol{\Psi}_j) + (\boldsymbol{\Psi}_i \cdot \nabla\mathbf{u}_c, \boldsymbol{\Psi}_k \cdot \nabla\boldsymbol{\Psi}_j)],\\
(\mathcal{M}_9^\delta)_j &= -\mathbf{y}^T(t)\left(\sum_{l=1}^m y_l(t)\mathcal{P}_{jl}^\delta\right)\mathbf{y}(t), \text{ with } (\mathcal{P}_{jl}^\delta)_{ik} = (\boldsymbol{\Psi}_i \cdot \nabla\boldsymbol{\Psi}_k, \boldsymbol{\Psi}_l \cdot \nabla\boldsymbol{\Psi}_j). \qquad (A.23)
\end{aligned}$$

The modification of Section A.1 can again be used to eliminate the term $\dot\gamma(t)\mathcal{M}_c$.

A.3 Derivation of the Adjoint Equation

As stated in Section 5.3, determination of the gradient of

$$\mathcal{J}(\gamma) = \int_0^T L(\mathbf{y}(\gamma), \gamma, t)\,dt$$
$$\text{s.t. } \dot{\mathbf{y}} = \phi(\mathbf{y}, \gamma, t)$$

requires computation of the terms $L_{\mathbf{y}}$, L_γ, $\phi_{\mathbf{y}}$ and ϕ_γ. In this section we carry out the algebraic reductions necessary to express $L_{\mathbf{y}}$, L_γ, $\phi_{\mathbf{y}}$ and ϕ_γ in terms of the POD basis functions. With

$$L(\mathbf{y}, \gamma, t) = \frac{1}{2}\left\|\mathbf{u}_n + \gamma(t)(\mathbf{u}_c + \boldsymbol{\Phi}\mathcal{M}_c) + \boldsymbol{\Phi}\mathbf{y}(t) - \mathbf{u}_d\right\|_{\mathrm{M}}^2,$$

where M is the finite element mass matrix, and $\boldsymbol{\Phi} \in \mathbb{R}^{2N,m}$ denotes the matrix with columns consisting of the finite element coefficient vectors of the POD basis functions $\boldsymbol{\Psi}_i$, we have

$$L_{\mathbf{y}}(\mathbf{y}, \gamma, t) = \boldsymbol{\Phi}^T\mathrm{M}(\mathbf{u}_n + \gamma(t)(\mathbf{u}_c + \boldsymbol{\Phi}\mathcal{M}_c) + \boldsymbol{\Phi}\mathbf{y}(t) - \mathbf{u}_d)$$

and

$$L\gamma(\mathbf{y}, \gamma, t) = (\mathbf{u}_c + \boldsymbol{\Phi}\mathcal{M}_c)^T\mathrm{M}(\mathbf{u}_n + \gamma(t)(\mathbf{u}_c + \boldsymbol{\Phi}\mathcal{M}_c) + \boldsymbol{\Phi}\mathbf{y}(t) - \mathbf{u}_d).$$

Assuming the modification (A.15) has been made to eliminate the term $\dot{\gamma}(t)\mathcal{M}_c$, we set $\mathbf{z} = \mathbf{y} + \gamma\mathcal{M}_c$ and have from (A.20)

$$\begin{aligned}\phi_{\mathbf{y}}(\mathbf{y},\gamma,t) = \mathcal{M}_3 + \gamma\mathcal{M}_4 + \frac{\partial}{\partial\mathbf{y}}\mathcal{M}_5(\mathbf{z},\mathbf{z}) + \delta_T^m[\mathcal{M}_3^\delta + \gamma\mathcal{M}_4^\delta + \frac{\partial}{\partial\mathbf{y}}\mathcal{M}_5^\delta(\mathbf{z},\mathbf{z})\\ + \gamma^2\mathcal{M}_7^\delta + \gamma\frac{\partial}{\partial\mathbf{y}}\mathcal{M}_8^\delta(\mathbf{z},\mathbf{z}) + \frac{\partial}{\partial\mathbf{y}}\mathcal{M}_9^\delta(\mathbf{z},\mathbf{z},\mathbf{z})],\end{aligned}$$

where the j-th rows of the derivatives of the multilinear forms are given by

$$\begin{aligned}\big(\frac{\partial}{\partial\mathbf{y}}\mathcal{M}_5(\mathbf{z},\mathbf{z})\big)_j &= -\mathbf{z}^T(\mathcal{Q}_j + \mathcal{Q}_j^T) \text{ with } \mathcal{Q}_j \text{ from (A.11)},\\ \big(\frac{\partial}{\partial\mathbf{y}}\mathcal{M}_5^\delta(\mathbf{z},\mathbf{z})\big)_j &= -\mathbf{z}^T(\mathcal{Q}_j^\delta + (\mathcal{Q}_j^\delta)^T) \text{ with } \mathcal{Q}_j^\delta \text{ from (A.21)},\\ \big(\frac{\partial}{\partial\mathbf{y}}\mathcal{M}_8^\delta(\mathbf{z},\mathbf{z})\big)_j &= -\mathbf{z}^T(\mathcal{R}_j^\delta + (\mathcal{R}_j^\delta)^T) \text{ with } \mathcal{R}_j^\delta \text{ from (A.22)},\\ \big(\frac{\partial}{\partial\mathbf{y}}\mathcal{M}_9^\delta(\mathbf{z},\mathbf{z},\mathbf{z})\big)_j &= -\mathbf{z}^T(\mathcal{P} + \mathcal{P}^T) - \mathbf{z}^T\,\mathcal{P}_j\,\mathbf{z} \text{ with } \mathcal{P} = \Big(\sum_{l=1}^m y_l\mathcal{P}_{jl}^\delta\Big),\\ &\qquad \mathcal{P}_j = \big(\mathcal{P}_{j1}^\delta,\dots,\mathcal{P}_{jm}^\delta\big) \text{ and } \mathcal{P}_{jl}^\delta \text{ from (A.23)},\end{aligned}$$

and

$$\begin{aligned}\phi_\gamma(\mathbf{y},\gamma,t) = \mathcal{M}_1 + 2\gamma\mathcal{M}_2 + \mathcal{M}_3\mathcal{M}_c + \mathcal{M}_4\mathbf{z} + \gamma\mathcal{M}_4\mathcal{M}_c + \frac{\partial}{\partial\gamma}\mathcal{M}_5(\mathbf{z},\mathbf{z})\\ + \delta_T^m[\mathcal{M}_1^\delta + 2\gamma\mathcal{M}_2^\delta + \mathcal{M}_3^\delta\mathcal{M}_c + \mathcal{M}_4^\delta\mathbf{z} + \gamma\mathcal{M}_4^\delta\mathcal{M}_c + \frac{\partial}{\partial\gamma}\mathcal{M}_5^\delta(\mathbf{z},\mathbf{z})\\ + 3\gamma^2\mathcal{M}_6^\delta + 2\gamma\mathcal{M}_7^\delta + \gamma^2\mathcal{M}_7^\delta\mathcal{M}_c + \mathcal{M}_8^\delta(\mathbf{z},\mathbf{z})\\ + \gamma\frac{\partial}{\partial\gamma}\mathcal{M}_8^\delta(\mathbf{z},\mathbf{z}) + \frac{\partial}{\partial\gamma}\mathcal{M}_9^\delta(\mathbf{z},\mathbf{z},\mathbf{z})],\end{aligned}$$

where the j-th components of the derivatives of the multilinear forms are given by

$$\begin{aligned}\big(\frac{\partial}{\partial\gamma}\mathcal{M}_5(\mathbf{z},\mathbf{z})\big)_j &= -\mathbf{z}^T(\mathcal{Q}_j + \mathcal{Q}_j^T)\mathcal{M}_c \text{ with } \mathcal{Q}_j \text{ from (A.11)},\\ \big(\frac{\partial}{\partial\gamma}\mathcal{M}_5^\delta(\mathbf{z},\mathbf{z})\big)_j &= -\mathbf{z}^T(\mathcal{Q}_j^\delta + (\mathcal{Q}_j^\delta)^T)\mathcal{M}_c \text{ with } \mathcal{Q}_j \text{ from (A.21)},\\ \big(\frac{\partial}{\partial\gamma}\mathcal{M}_8^\delta(\mathbf{z},\mathbf{z})\big)_j &= -\mathbf{z}^T(\mathcal{R}_j^\delta + (\mathcal{R}_j^\delta)^T)\mathcal{M}_c \text{ with } \mathcal{R}_j^\delta \text{ from (A.22) and}\\ \big(\frac{\partial}{\partial\gamma}\mathcal{M}_9^\delta(\mathbf{z},\mathbf{z},\mathbf{z})\big)_j &= -\mathbf{z}^T(\mathcal{P} + \mathcal{P}^T)\mathcal{M}_c - \mathbf{z}^T\Big(\sum_{l=1}^m(\mathcal{M}_c)_l\mathcal{P}_{jl}^\delta\Big)\mathbf{z}\\ &\qquad \text{with } \mathcal{P} = \Big(\sum_{l=1}^m y_l\mathcal{P}_{jl}^\delta\Big) \text{ and } \mathcal{P}_{jl}^\delta \text{ from (A.23)}.\end{aligned}$$

A.4 Calculation of the ODE Coefficients

To calculate the coefficients in (A.10) and (A.20) we expand the flow fields and POD basis functions into their finite element representations, and use the underlying linearity to reduce the evaluation of the scalar product terms to problems involving the finite element basis functions. These can then be evaluated using the code of the finite element solver. These expansions require use of the identities

$$(\boldsymbol{\Psi}, \boldsymbol{\Phi}) = \left(\boldsymbol{\Psi}^{(1)}, \boldsymbol{\Phi}^{(1)}\right) + \left(\boldsymbol{\Psi}^{(2)}, \boldsymbol{\Phi}^{(2)}\right), \tag{A.24}$$

$$(\nabla\boldsymbol{\Psi}, \nabla\boldsymbol{\Phi}) = \sum_{\mu=1}^{2}\sum_{\nu=1}^{2} \left(\nabla_\mu \boldsymbol{\Psi}^{(\nu)}, \nabla_\mu \boldsymbol{\Phi}^{(\nu)}\right), \tag{A.25}$$

$$(\boldsymbol{\Psi} \cdot \nabla\boldsymbol{\Phi}, \boldsymbol{\Upsilon}) = \sum_{\mu=1}^{2}\sum_{\nu=1}^{2} \left(\boldsymbol{\Psi}^{(\mu)} \nabla_\mu \boldsymbol{\Phi}^{(\nu)}, \boldsymbol{\Upsilon}^{(\nu)}\right), \tag{A.26}$$

and

$$(\boldsymbol{\Psi} \cdot \nabla\boldsymbol{\Phi}, \boldsymbol{\Upsilon} \cdot \nabla\boldsymbol{\Lambda}) = \sum_{\mu=1}^{2}\sum_{\nu=1}^{2}\sum_{\sigma=1}^{2} \left(\boldsymbol{\Psi}^{(\mu)} \nabla_\mu \boldsymbol{\Phi}^{(\sigma)}, \Upsilon^{(\nu)} \nabla_\nu \boldsymbol{\Lambda}^{(\sigma)}\right), \tag{A.27}$$

where $\boldsymbol{\Psi}, \boldsymbol{\Phi}, \boldsymbol{\Upsilon}, \boldsymbol{\Lambda} : \mathbb{R}^2 \to \mathbb{R}^2$ are vector-valued with real-valued component functions ($\boldsymbol{\Psi} = (\boldsymbol{\Psi}^{(1)}, \boldsymbol{\Psi}^{(2)})^T$, for example), and ∇_μ, $\mu = 1, 2$ is the μ-th component of the gradient.

Expansion of the diffusive terms. The average flow field $\mathbf{u}_n$ for the POD snapshots $\mathbf{u}_j$, $j = 1, \ldots, n$ is given by

$$\mathbf{u}_n = \frac{1}{n}\sum_{j=1}^{n} \mathbf{u}_j = \frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{N} \begin{pmatrix} u_{ji}\Phi_i \\ u_{j(i+N)}\Phi_i \end{pmatrix} = \sum_{i=1}^{N} \frac{1}{n}\sum_{j=1}^{n} \begin{pmatrix} u_{ji}\Phi_i \\ u_{j(i+N)}\Phi_i \end{pmatrix},$$

where N is the order of the finite element discretization and Φ_i, $i = 1, \ldots, N$ denote the finite element basis functions[1]. By setting $u_i := \frac{1}{n}\sum_{j=1}^{n} u_{ji}$, $i = 1, \ldots, 2N$ the average flow field can be written in terms of the finite element basis functions as

$$\mathbf{u}_n = \sum_{i=1}^{N} \begin{pmatrix} u_i\Phi_i \\ u_{i+N}\Phi_i \end{pmatrix}. \tag{A.28}$$

Similar arguments apply to the POD basis functions, which consist of linear combinations of the POD snapshots as described in Chapter 4, so that using (A.25) we can write

$$\begin{aligned}(\nabla\mathbf{u}_n, \nabla\boldsymbol{\Psi}_j) &= \left(\nabla_1 \mathbf{u}_n^{(1)}, \nabla_1 \boldsymbol{\Psi}_j^{(1)}\right) + \left(\nabla_1 \mathbf{u}_n^{(2)}, \nabla_1 \boldsymbol{\Psi}_j^{(2)}\right) \\ &\quad + \left(\nabla_2 \mathbf{u}_n^{(1)}, \nabla_2 \boldsymbol{\Psi}_j^{(1)}\right) + \left(\nabla_2 \mathbf{u}_n^{(2)}, \nabla_2 \boldsymbol{\Psi}_j^{(2)}\right) \\ &= \sum_{i=1}^{N}\sum_{l=1}^{N} [(u_i \nabla_1 \Phi_i, \Psi_{jl} \nabla_1 \Phi_l) + \left(u_{i+N} \nabla_1 \Phi_i, \Psi_{j(l+N)} \nabla_1 \Phi_l\right) \\ &\qquad + (u_i \nabla_2 \Phi_i, \Psi_{jl} \nabla_2 \Phi_l) + \left(u_{i+N} \nabla_2 \Phi_i, \Psi_{j(l+N)} \nabla_2 \Phi_l\right)]\end{aligned}$$

[1] We ask the reader's forgiveness for the abuse of notation regarding $\mathbf{u}_n$, which has been used to denote the average flow field, as well as the n-th snapshot.

$$\begin{aligned}
&= \sum_{i=1}^{N}\sum_{l=1}^{N}[u_i \Psi_{jl}\left((\nabla_1\Phi_i, \nabla_1\Phi_l) + (\nabla_2\Phi_i, \nabla_2\Phi_l)\right) \\
&\qquad + u_{i+N}\Psi_{j(l+N)}\left((\nabla_1\Phi_i, \nabla_1\Phi_l) + (\nabla_2\Phi_i, \nabla_2\Phi_l)\right)] \\
&= \sum_{i=1}^{N} u_i \sum_{l=1}^{N} \Psi_{jl}\left(\nabla\Phi_i, \nabla\Phi_l\right) + \sum_{i=1}^{N} u_{i+N} \sum_{l=1}^{N} \Psi_{j(l+N)}\left(\nabla\Phi_i, \nabla\Phi_l\right) \\
&= u_n^T \begin{pmatrix} \mathrm{S} & \mathrm{O} \\ \mathrm{O} & \mathrm{S} \end{pmatrix} \Psi_{j:}
\end{aligned}$$

and, likewise,

$$(\nabla\boldsymbol{\Psi}_i, \nabla\boldsymbol{\Psi}_j) = \Psi_{i:}^T \begin{pmatrix} \mathrm{S} & \mathrm{O} \\ \mathrm{O} & \mathrm{S} \end{pmatrix} \Psi_{j:},$$

where $u_n, \Psi_{i:}, \Psi_{j:} \in \mathbb{R}^{2N}$ are coefficient vectors, and $\mathrm{S} = (\nabla\Phi_i, \nabla\Phi_j)_{1\leq i,j\leq N}$ is the finite element stiffness matrix.

Expansion of the convective term. Expansion of convective term is somewhat more involved, as the final form of the representation depends on the order of summation. We begin by using (A.26) to reduce the term to a linear combination of the finite element basis functions.

$$\begin{aligned}
(\boldsymbol{\Psi}_i \cdot \nabla\boldsymbol{\Psi}_k, \boldsymbol{\Psi}_j) &= \left(\boldsymbol{\Psi}_i^{(1)}\nabla_1\boldsymbol{\Psi}_k^{(1)}, \boldsymbol{\Psi}_j^{(1)}\right) + \left(\boldsymbol{\Psi}_i^{(2)}\nabla_2\boldsymbol{\Psi}_k^{(1)}, \boldsymbol{\Psi}_j^{(1)}\right) \\
&\quad + \left(\boldsymbol{\Psi}_i^{(1)}\nabla_1\boldsymbol{\Psi}_k^{(2)}, \boldsymbol{\Psi}_j^{(2)}\right) + \left(\boldsymbol{\Psi}_i^{(2)}\nabla_2\boldsymbol{\Psi}_k^{(2)}, \boldsymbol{\Psi}_j^{(2)}\right) \\
&= \sum_{m=1}^{N} \Psi_{im} \sum_{p=1}^{N} \Psi_{kp} \sum_{n=1}^{N} \Psi_{jn}\left(\Phi_m\nabla_1\Phi_p, \Phi_n\right) \\
&\quad + \sum_{m=1}^{N} \Psi_{i(m+N)} \sum_{p=1}^{N} \Psi_{kp} \sum_{n=1}^{N} \Psi_{jn}\left(\Phi_m\nabla_2\Phi_p, \Phi_n\right) \\
&\quad + \sum_{m=1}^{N} \Psi_{im} \sum_{p=1}^{N} \Psi_{k(p+N)} \sum_{n=1}^{N} \Psi_{j(n+N)}\left(\Phi_m\nabla_1\Phi_p, \Phi_n\right) \\
&\quad + \sum_{m=1}^{N} \Psi_{i(m+N)} \sum_{p=1}^{N} \Psi_{k(p+N)} \sum_{n=1}^{N} \Psi_{j(n+N)}\left(\Phi_m\nabla_2\Phi_p, \Phi_n\right).
\end{aligned}$$

By checking the other possibilities, one can easily confirm that it is computationally advantageous to sum first over the index m. Taking the inner summation over m leads to

$$\begin{aligned}
(\boldsymbol{\Psi}_i \cdot \nabla\boldsymbol{\Psi}_k, \boldsymbol{\Psi}_j) &= \Psi_{k:}^{(1)^T}\left(\left(\sum_{m=1}^{N} \Psi_{im}\left(\Phi_m\nabla_1\Phi_p, \Phi_n\right)\right)_{p,n}\right)\Psi_{j:}^{(1)} \\
&\quad + \Psi_{k:}^{(1)^T}\left(\left(\sum_{m=1}^{N} \Psi_{i(m+N)}\left(\Phi_m\nabla_2\Phi_p, \Phi_n\right)\right)_{p,n}\right)\Psi_{j:}^{(1)} \\
&\quad + \Psi_{k:}^{(2)^T}\left(\left(\sum_{m=1}^{N} \Psi_{im}\left(\Phi_m\nabla_1\Phi_p, \Phi_n\right)\right)_{p,n}\right)\Psi_{j:}^{(2)}
\end{aligned}$$

$$
\begin{aligned}
&+ \Psi_{k:}^{(2)^T} \left(\big(\sum_{m=1}^{N} \Psi_{i(m+N)} \left(\Phi_m \nabla_2 \Phi_p, \Phi_n\right)\big)_{p,n} \right) \Psi_{j:}^{(2)} \\
&= \Psi_{k:}^T \begin{pmatrix} C(\Psi_{i:}) & \mathbf{0} \\ \mathbf{0} & C(\Psi_{i:}) \end{pmatrix} \Psi_{j:}
\end{aligned}
$$

where

$$
C_{pn}(\Psi_{i:}) = \sum_{m=1}^{N} \Psi_{im} \left(\Phi_m \nabla_1 \Phi_p, \Phi_n\right) + \sum_{m=1}^{N} \Psi_{i(m+N)} \left(\Phi_m \nabla_2 \Phi_p, \Phi_n\right).
$$

Expansion of the streamline diffusion term. The SDPOD model requires expansion of the streamline diffusion term. We begin the expansion by using (A.27) to obtain

$$
\begin{aligned}
&(\boldsymbol{\Psi}_i \cdot \nabla \boldsymbol{\Psi}_k, \boldsymbol{\Psi}_l \cdot \nabla \boldsymbol{\Psi}_j) \\
&\quad = \left(\boldsymbol{\Psi}_i^{(1)} \nabla_1 \boldsymbol{\Psi}_k^{(1)}, \boldsymbol{\Psi}_l^{(1)} \nabla_1 \boldsymbol{\Psi}_j^{(1)}\right) + \left(\boldsymbol{\Psi}_i^{(1)} \nabla_1 \boldsymbol{\Psi}_k^{(1)}, \boldsymbol{\Psi}_l^{(2)} \nabla_2 \boldsymbol{\Psi}_j^{(1)}\right) \\
&\qquad + \left(\boldsymbol{\Psi}_i^{(2)} \nabla_2 \boldsymbol{\Psi}_k^{(1)}, \boldsymbol{\Psi}_l^{(1)} \nabla_1 \boldsymbol{\Psi}_j^{(1)}\right) + \left(\boldsymbol{\Psi}_i^{(2)} \nabla_2 \boldsymbol{\Psi}_k^{(1)}, \boldsymbol{\Psi}_l^{(2)} \nabla_2 \boldsymbol{\Psi}_j^{(1)}\right) \\
&\qquad + \left(\boldsymbol{\Psi}_i^{(1)} \nabla_1 \boldsymbol{\Psi}_k^{(2)}, \boldsymbol{\Psi}_l^{(1)} \nabla_1 \boldsymbol{\Psi}_j^{(2)}\right) + \left(\boldsymbol{\Psi}_i^{(1)} \nabla_1 \boldsymbol{\Psi}_k^{(2)}, \boldsymbol{\Psi}_l^{(2)} \nabla_2 \boldsymbol{\Psi}_j^{(2)}\right) \\
&\qquad + \left(\boldsymbol{\Psi}_i^{(2)} \nabla_2 \boldsymbol{\Psi}_k^{(2)}, \boldsymbol{\Psi}_l^{(1)} \nabla_1 \boldsymbol{\Psi}_j^{(2)}\right) + \left(\boldsymbol{\Psi}_i^{(2)} \nabla_2 \boldsymbol{\Psi}_k^{(2)}, \boldsymbol{\Psi}_l^{(2)} \nabla_2 \boldsymbol{\Psi}_j^{(2)}\right).
\end{aligned}
$$

Use of the representations $\boldsymbol{\Psi}_i^{(1)} = \sum_{i'=1}^{N} \Psi_{ii'} \Phi_{i'}$ and $\boldsymbol{\Psi}_i^{(2)} = \sum_{i'=1}^{N} \Psi_{i(i'+N)} \Phi_{i'}$ and rearrangement of terms leads to the sum

$$
\begin{aligned}
&(\boldsymbol{\Psi}_i \cdot \nabla \boldsymbol{\Psi}_k, \boldsymbol{\Psi}_l \cdot \nabla \boldsymbol{\Psi}_j) \\
&\quad = \sum_{i',k',l',j'=1}^{N} p_1 \left(\Phi_{i'} \nabla_1 \Phi_{k'}, \Phi_{l'} \nabla_1 \Phi_{j'}\right) + p_2 \left(\Phi_{i'} \nabla_1 \Phi_{k'}, \Phi_{l'} \nabla_2 \Phi_{j'}\right) + \\
&\qquad\qquad p_3 \left(\Phi_{i'} \nabla_2 \Phi_{k'}, \Phi_{l'} \nabla_1 \Phi_{j'}\right) + p_4 \left(\Phi_{i'} \nabla_2 \Phi_{k'}, \Phi_{l'} \nabla_2 \Phi_{j'}\right),
\end{aligned}
$$

where

$$
\begin{aligned}
p_1 &= \left[\Psi_{ii'} \Psi_{kk'} \Psi_{ll'} \Psi_{jj'} + \Psi_{ii'} \Psi_{k(k'+N)} \Psi_{ll'} \Psi_{j(j'+N)}\right], \\
p_2 &= \left[\Psi_{ii'} \Psi_{kk'} \Psi_{l(l'+N)} \Psi_{jj'} + \Psi_{ii'} \Psi_{k(k'+N)} \Psi_{l(l'+N)} \Psi_{j(j'+N)}\right], \\
p_3 &= \left[\Psi_{i(i'+N)} \Psi_{kk'} \Psi_{ll'} \Psi_{jj'} + \Psi_{i(i'+N)} \Psi_{k(k'+N)} \Psi_{ll'} \Psi_{j(j'+N)}\right], \\
p_4 &= \left[\Psi_{i(i'+N)} \Psi_{kk'} \Psi_{l(l'+N)} \Psi_{jj'} + \Psi_{i(i'+N)} \Psi_{k(k'+N)} \Psi_{l(l'+N)} \Psi_{j(j'+N)}\right].
\end{aligned}
$$

Finally, rearranging terms and switching in part to matrix notation, we can write

$$
\begin{aligned}
(\boldsymbol{\Psi}_i \cdot \nabla \boldsymbol{\Psi}_k, \boldsymbol{\Psi}_l \cdot \nabla \boldsymbol{\Psi}_j) &= \Psi_{k:}^{(1)^T} \mathcal{Q} \Psi_{j:}^{(1)} + \Psi_{k,:}^{(2)^T} \mathcal{Q} \Psi_{j,:}^{(2)} \\
&= \Psi_{k:}^T \begin{pmatrix} \mathcal{Q} & \mathbf{0} \\ \mathbf{0} & \mathcal{Q} \end{pmatrix} \Psi_{j:},
\end{aligned}
$$

where

$$
\begin{aligned}
\mathcal{Q}_{i'k'} = \sum_{i',l'=1}^{N} &\Psi_{ii'} \Psi_{ll'} \left(\Phi_{i'} \nabla_1 \Phi_{k'}, \Phi_{l'} \nabla_1 \Phi_{j'}\right) + \Psi_{ii'} \Psi_{l(l'+N)} \left(\Phi_{i'} \nabla_1 \Phi_{k'}, \Phi_{l'} \nabla_2 \Phi_{j'}\right) \\
&+ \Psi_{i(i'+N)} \Psi_{ll'} \left(\Phi_{i'} \nabla_2 \Phi_{k'}, \Phi_{l'} \nabla_1 \Phi_{j'}\right) + \Psi_{i(i'+N)} \Psi_{l(l'+N)} \left(\Phi_{i'} \nabla_2 \Phi_{k'}, \Phi_{l'} \nabla_2 \Phi_{j'}\right)
\end{aligned}
$$

Initial condition terms. The initial conditions (A.14) require expansion of the terms $(\mathbf{u}_{t_0}, \boldsymbol{\Psi}_j)$, $(\mathbf{u}_n, \boldsymbol{\Psi}_j)$ and $(\mathbf{u}_c, \boldsymbol{\Psi}_j)$, j=1,...,m. These expansions clearly take the form

$$(\mathbf{u}_{t_0}, \boldsymbol{\Psi}_j) = u_{t_0}^T \begin{pmatrix} \mathrm{M} & \mathrm{O} \\ \mathrm{O} & \mathrm{M} \end{pmatrix} \Psi_{j:}, \quad (\mathbf{u}_n, \boldsymbol{\Psi}_j) = u_n^T \begin{pmatrix} \mathrm{M} & \mathrm{O} \\ \mathrm{O} & \mathrm{M} \end{pmatrix} \Psi_{j:} \quad \text{and}$$

$$(\mathbf{u}_c, \boldsymbol{\Psi}_j) = u_c^T \begin{pmatrix} \mathrm{M} & \mathrm{O} \\ \mathrm{O} & \mathrm{M} \end{pmatrix} \Psi_{j:}, \quad \text{j=1,}\ldots\text{,m,}$$

where $u_{t_0}^T, u_n^T, u_c^T, \Psi_{j:} \in \mathbb{R}^{2N}$ are coefficient vectors, and $\mathrm{M} = (\Phi_i, \Phi_j)_{1 \leq i,j \leq N}$ is the finite element mass matrix.

Bibliography

[1] F. Abergel and R. Temam, *On some control problems in fluid mechanics*, Theoret. Comput. Fluid Dynamics, 1 (1990), pp. 303–325.

[2] R. Adams, *Sobolev Spaces*, Academic Press, New York, 1975.

[3] K. Afanasiev and M. Hinze, *Adaptive control of a wake flow using proper orthogonal decomposition*, in Shape Optimization & Optimal Design, M. P. J. Cagnol and J.-P. Zolsio, eds., no. 216 in Lecture Notes in Pure and Applied Mathematics, Marcel Dekker/CRC Press, 2001, pp. 317–332.

[4] N. Alexandrov, J. Dennis, R. Lewis, and V. Torczon, *A trust region framework for managing the use of approximation models*, Structural Optimization, 15 (1998), pp. 16–23.

[5] B. Allan, *A reduced order model of the linearized incompressible Navier–Stokes equations for the sensor/actuator placement problem*, ICASE Report 2000–19, ICASE, NASA Langley Research Center, Hampton, 2000.

[6] E. Arian, M. Fahl, and E. Sachs, *Trust–region proper orthogonal decomposition for flow control*, ICASE Report 2000–25, ICASE, NASA Langley Research Center, Hampton, 2000.

[7] J. Atwell and B. King, *Proper orthogonal decomposition for reduced basis controllers for parabolic systems*, Mathematical and Computer Modelling, 33 (2001), pp. 1–19.

[8] N. Aubry, *On the hidden beauty of the proper orthogonal decomposition*, Theoret. Comput. Fluid Dynamics, 2 (1991), pp. 339–352.

[9] H. Banks, M. Joyner, B. Wincheski, and W. Winfree, *Evaluation of material integrity using reduced order computational methodology*, Techreport CRSC-TR-99-30, North Carolina State University, Raleigh, 1999.

[10] R. Banks, P. Gui, and R. Marcia, *Interior point methods for a class of elliptic variational inequalities*, in Large–scale PDE–constrained Optimization, Biegler, Ghattas, Heinkenschloss, and V. B. Waanders, eds., vol. 30, Springer Verlag, 2003, pp. 218–235.

[11] S. Benson, L. C. McInnes, J. Mor, and J. Sarich, *Scalable algorithms in optimization: Computational experiments*, Techreport ANL/MCS–P1175–0604, Mathematics and Computer Science Division, Argonne National Laboratory, 2004.

[12] M. Berggren, *Numerical solution of a flow–control problem: Vorticity reduction by dynamic boundary action*, SIAM J. Sci. Computing, 19 (1998), pp. 829–860.

[13] G. Berkooz, P. Holmes, and J. Lumley, *The proper orthogonal decomposition in the analysis of turbulent flows*, Annu. Rev. Fluid Mech., 25 (1993), pp. 539–575.

[14] D. Bertsekas, *Nonlinear Programming*, Athena Scientific, 2nd ed., 1999.

[15] T. Betts and S. Erb, *Optimal low thrust trajectory to the moon*, SIAM Journal on Applied Dynamical Systems, 2 (2003), pp. 144–170.

[16] T. Bewley, P. Moin, and R. Temam, *DNS–based predictive control of turbulence: An optimal benchmark for feedback algorithms*, J. Fluid Mech., 447 (2001), pp. 179–225.

[17] T. Bewley, R. Temam, and M. Ziane, *Existence and uniqueness of optimal control to the Navier-Stokes equations*, C.R. Acad. Sci. Paris, 330 (2000), pp. 1–5.

[18] ——, *A general framework for robust control in fluid mechanics*, Physica D, 138 (2000), pp. 360–392.

[19] J. Borggaard and J. Burns, *Asymptotically consistent gradients in optimal design*, in Multidisciplinary design optimization - State of the Art. Proceedings of the ICASE/NASA Langley workshop on multidisciplinary design optimization, N. Alexandrov and M. Hussaini, eds., Philadelphia, 1997, SIAM, pp. 303–314.

[20] F. Brezzi, J. Rappaz, and P. Raviart, *Finite dimensional approximation of non–linear problems I. Branches of non–singular solutions*, Numer. Math., 36 (1980), pp. 1–25.

[21] M. Bristeau, R. Glowinski, and J. Periaux, *Numerical methods for the Navier–Stokes equations: Applications to the simulation of compressible and incompressible viscous flows*, Comput. Phys. Reports, 6 (1987), pp. 73–187.

[22] A. Brooks and T. Hughes, *Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations*, Comp. Methods Appl. Mech. Engrg., 32 (1982), pp. 199–259.

[23] J. Burkardt and J. Peterson, *Control of steady incompressible 2D channel flow*, in Flow control, M. Gunzburger, ed., Springer, New York, 1995, pp. 11–126.

[24] J. Burns and Y.-R. Ou, *On active control of vortex shedding*, Techreport 94–0182, AIAA, 1994.

[25] R. Carter, *Numerical optimization in Hilbert space using inexact function and gradient information*, Techreport 89–45, Institute for Computer Applications in Science and Engineering, 1989.

[26] ——, *On the global convergence of trust region algorithms using inexact gradient information*, SIAM J. Numer. Anal., 28 (1991), pp. 251–265.

[27] ——, *Numerical experience with a class of algorithms for nonlinear optimization using inexact function and gradient information*, SIAM J. Sci. Comp., 14 (1993), pp. 368–388.

[28] W. Cazemier, R. Verstappen, and A. Veldman, *Proper orthogonal decomposition and low–dimensional models for driven cavity flows*, Physics of Fluids, 10 (1998), pp. 1685–1699.

[29] K. Chang, R. Haftka, G. Giles, and P.-J. Kao, *Sensitivity–based scaling for approximating structural response*, J. of Aircraft, 30 (1993), pp. 283–288.

[30] F. Chatelin, *Spectral Approximation of Linear Operators*, Computer Science and Applied Mathematics, Academic Press, New York, 1983.

[31] A. Chorin, *Numerical solution of the Navier–Stokes equations*, Math. Comp., 22 (1968), pp. 746–762.

[32] P. Ciarlet, *The Finite Element Method for Elliptic Problems*, vol. 4, North–Holland, Amsterdam, 1978.

[33] P. Clment, *Approximation by finite element functions using local regularization*, RAIRO Anal. Numr., 9 (1975), pp. 77–84.

[34] T. Coleman and Y. Zhang, *Optimization toolbox user's guide.* The Mathworks Inc. `http://www.mathworks.com/`, 2005.

[35] A. Conn, N. Gould, and P. Toint, *Trust–Region Methods*, MPS–SIAM Series on Optimization, SIAM, Philadelphia, 2000.

[36] C. Cuvelier, A. Segal, and A. van Steenhoven, *Finite Element Methods and Navier-Stokes Equations*, D. Reidel, Dordrecht, 1986.

[37] R. Dautray and J.-L. Lions, *Mathematical Analysis and Numerical Methods for Science and Technology*, vol. 1–6, Springer-Verlag, New York, 1993.

[38] J. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.

[39] M. Desai and K. Ito, *Optimal controls of Navier–Stokes equations*, SIAM J. Control and Optimization, 32 (1994), pp. 1428–1446.

[40] M. Fahl, *Computation of PODs for fluid flows with Lanczos methods*, Techreport 99–13, University of Trier, Germany, 1999.

[41] ——, *Trust–region Methods for Flow Control Based on Reduced Order Modelling*, PhD thesis, Univerity of Trier, Trier, Germany, 2000.

[42] H. Fattorini and S. Sritharan, *Existence of optimal controls for viscous flow problems*, Proc. R. Soc. of Lond. A, 439 (1992), pp. 81–102.

[43] ——, *Necessary and sufficient conditions for optimal controls in viscous flow problems*, Proc. R. Soc. of Edinb., 124A (1994), pp. 211–251.

[44] L. Franca and S. Frey, *Stabilized finite element methods: II. The incompressible Navier–Stokes equations*, Comp. Methods Appl. Mech. Engrg., 99 (1992), pp. 209–233.

[45] L. Franca, S. Frey, and T. Hughes, *Stabilized finite element methods: I. Applications to the advective–diffusive model*, Comp. Methods Appl. Mech. Engrg., 95 (1992), pp. 253–276.

[46] T.-P. Fries and H. Matthies, *A review of Petrov–Galerkin stabilization approaches and an extension to meshfree methods*, Informatikbericht 2004–01, Technical University Braunschweig, Brunswick, Germany, 2004.

[47] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, Boston, 1990.

[48] A. Fursikov, M. Gunzburger, and L. Hou, *Boundary value problems and optimal boundary control for the Navier–Stokes system: The two–dimensional case*, SIAM J. Control Optim., 36 (1998), pp. 852–894.

[49] M. Gad–el–Hak, A. Pollard, and J.-P. Bonnet, eds., *Flow Control: Fundamentals and Practices*, Springer-Verlag, Berlin, 1998.

[50] V. Girault and P. Raviart, *Finite Element Methods for Navier–Stokes Equations*, vol. 749 of Lecture Notes in Mathematics, Springer–Verlag, Berlin, Heidelberg, 1981.

[51] ——, *Finite Element Methods for Navier–Stokes Equations*, vol. 5 of Springer Series in Computational Mathematics, Springer–Verlag, Berlin, Heidelberg, 1986.

[52] R. Glowinski and J. Periaux, *Numerical methods for nonlinear problems in fluid dynamics*, in Proc. Intern. Seminar on Scientific Supercomputers, North–Holland, Paris, Feb. 2–6, 1987.

[53] G. Golub and C. V. Loan, *Matrix Computations*, John Hopkins University Press, 1989.

[54] N. Gould, D. Orban, A. Sartenaer, and P. Toint, *Sensitivity of trust–region algorithms to their parameters*, Techreport TR04/07, Department of Mathematics University of Namur, 61, rue de Bruxelles, B–5000 Namur (Belgium), 2004.

[55] W. Graham, J. Peraire, and K. Tang, *Optimal control of vortex shedding using low order models. Part I: Open–loop model development*, International Journal for Numerical Methods in Engineering, 44 (1999), pp. 945–972.

[56] ——, *Optimal control of vortex shedding using low order models. Part II: Model based control*, International Journal for Numerical Methods in Engineering, 44 (1999), pp. 973–990.

[57] S. Gratton, A. Sartenaer, and P. Toint, *Recursive trust–region methods for multilevel nonlinear optimization (part I): Global convergence and complexity*, Techreport TR04/06, Department of Mathematics University of Namur, 61, rue de Bruxelles, B–5000 Namur (Belgium), 2004.

[58] A. Griewank and P. Toint, *Local convergence analysis for partitioned quasi–Newton updates*, Numerische Mathematik, 39 (1982), pp. 429–448.

[59] R. Grimshaw, *Nonlinear Ordinary Differential Equations*, CRC–Press, 1990.

[60] P. Grisvard, *Elliptic Problems in Nonsmooth Domains*, Pitman, Boston, 1985.

[61] M. Gunzburger, ed., *Flow Control*, Springer–Verlag, New York, 1995.

[62] ——, *A prehistory of flow control and optimization*, in Flow control, M. Gunzburger, ed., Springer, New York, 1995, pp. 185–195.

[63] M. Gunzburger and S. Manservisi, *The velocity tracking problem for Navier-Stokes flows with bounded distributed controls*, SIAM J. Control Optim., 37 (1999), pp. 1913–1945.

[64] ——, *Analysis and approximation of the velocity tracking problem for Navier–Stokes flows with distributed control*, SIAM J. Numer. Anal., 37 (2000), pp. 1481–1512.

[65] ——, *The velocity tracking problem for Navier–Stokes flows with boundary control*, SIAM J. Control Optim., 39 (2000), pp. 594–634.

[66] R. Haberman, *Applied Elementary Partial Differential Equations*, Prentice–Hall, Inc., New Jersey, 1987.

[67] P. Hansbo and A. Szepessy, *A velocity–pressure streamline diffusion finite element method for the incompressible navier–stokes equations*, Comput. Methods Appl. Mech. Engrg., 84 (1990), pp. 175–192.

[68] P. Hansen, *Rank–Deficient and Discrete Ill–Posed Problems. Numerical Aspects of Linear Inversion*, SIAM Monographs on Mathematical Modeling and Computation, SIAM, Philadelphia, 1998.

[69] M. Heinkenschloss, *Formulation and analysis of a sequential quadratic programming method for the optimal Dirichlet boundary control of Navier–Stokes flow*, Appl. Optim., 15 (1998), pp. 178–203.

[70] M. Hinze and K. Kunisch, *Three control methods for time–dependent fluid flow*, Flow, Turbulence and Combustion, 65 (2000), pp. 273–298.

[71] M. Hinze and S. Volkwein, *Proper orthogonal decomposition surrogate models for nonlinear dynamical systems: Error estimates and suboptimal control.* To appear in Lecture Notes in Computational Science and Engineering, Vol. 45, 2004.

[72] P. Holmes, J. Lumley, and G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*, Cambridge University Press, Cambridge, 1996.

[73] D. Hömberg and S. Volkwein, *Control of laser surface hardening by a reduced–order approach using proper orthogonal decomposition*, Mathematical and Computer Modeling, 38 (2003), pp. 1003–1028.

[74] L. Hou, M. Gunzburger, and T. Svobodny, *Analysis and finite element approximations of optimal control problems for the stationary Navier–Stokes equations with Dirichlet controls*, Math. Model. Numer. Anal., 25 (1991), pp. 711–748.

[75] ——, *Analysis and finite element approximations of optimal control problems for the stationary Navier–Stokes equations with distributed and Neumann controls*, Math. Comp., 57 (1991), pp. 123–151.

[76] L. Hou and S. Ravindran, *A penalized Neumann control approach for solving an optimal Dirichlet control problem for the Navier–Stokes equations*, SIAM J. Control Optim., 36 (1998), pp. 1795–1814.

[77] ——, *Numerical approximation of optimal flow control problems by a penalty method: Error estimates and numerical results*, SIAM J. Sci. Comput., 20 (1999), pp. 1753–1777.

[78] T. Hughes and A. Brooks, *A multi–dimensional upwind scheme with no crosswind diffusion*, in Finite Element Methods for Convection Dominated Flow, vol. 34 of AMD, Amer. Soc. Mech. Engrs. (ASME), New York, 1979, pp. 19–35.

[79] K. Ito and S. Ravindran, *A reduced-order method for simulation and control of fluid flows*, J. Comput. Phys., 143 (1998), pp. 403–425.

[80] V. John, G. Matthies, F. Schieweck, and L. Tobiska, *A streamline–diffusion method for nonconforming finite element approximations applied to convection–diffusion problems*, Comput. Methods Appl. Mech. Engrg., 166 (1998), pp. 85–97.

[81] V. John, J. Maubach, and L. Tobiska, *Nonconforming streamline–diffusion–finite–element–methods for convection–diffusion problems*, Numer. Math, 78 (1997), pp. 165–188.

[82] C. Johnson, U. Nävert, and J. Pitkäranta, *Finite element methods for linear hyperbolic problems*, Comput. Methods Appl. Mech. Engrg., 45 (1984), pp. 285–312.

[83] C. Johnson and J. Saranen, *Streamline diffusion methods for the incompressible Euler and Navier–Stokes equations*, Math. Comp., 47 (1986), pp. 1–18.

[84] C. Johnson, A. Schatz, and L. Wahlbin, *Crosswind smear and pointwise errors in streamline diffusion finite element methods*, Math. Comput., 49 (1987), pp. 25–38.

[85] D. Jordan and P. Smith, *Nonlinear Ordinary Differential Equations*, Oxford Applied Mathematics and Computing Science Series, Oxford University Press, 2nd ed., 1987.

[86] J. Jorgensen, J. Sorensen, and M. Brons, *Low-dimensional modeling of a driven cavity flow with two free parameters*, Theoret. Comput. Fluid Dynamics, 16 (2003), pp. 299–317.

[87] C. Kelley and E. Sachs, *Truncated Newton methods for optimization with inaccurate functions and gradients*, Techreport CRSC-TR-99-20, North Carolina State University, Raleigh, 1999.

[88] ——, *A trust region method for parabolic boundary control problems*, SIAM J. Optim., 9 (1999), pp. 1064–1081.

[89] A. King, J. Billingham, and S. Otto, *Differential Equations*, Cambridge University Press, 2003.

[90] P. Klouček and F. Rys, *Stability of the fractional step θ-scheme for the nonstationary Navier-Stokes equations*, SIAM J. Numer. Anal., 31 (1994), pp. 1312–1335.

[91] K. Kunisch and S. Volkwein, *Galerkin proper orthogonal decomposition methods for parabolic problems*, Numer. Math., 90 (2001), pp. 117–148.

[92] ——, *Crank–Nicolson Galerkin proper orthogonal decomposition approximations for a general equation in fluid dynamics*, in Proceedings of the 18th GAMM–Seminar, Leipzig, 2002, pp. 97–114.

[93] ——, *Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics*, SIAM J. Numer. Anal., 40 (2002), pp. 492–515.

[94] K. Kunisch, S. Volkwein, and L. Xie, *HJB–POD based feedback design for the optimal control of evolution problems*. Submitted, 2003.

[95] O. Ladyzhenskaya, *The Mathematical Theory of Viscous Incompressible Flow*, Gordon and Breach, New York, 2nd ed., 1969.

[96] J. Lambert, *Numerical Methods for Ordinary Differential Systems*, John Wiley & Sons Ltd., Chichester, 1991.

[97] F. Leibfritz and S. Volkwein, *Numerical feedback controller design for PDE systems using model reduction: Techniques and case studies*, techreport, University of Trier, 2004.

[98] ——, *Reduced order output feedback control design for PDE systems using proper orthogonal decomposition and nonlinear semidefinite programming*. To appear in Linear Algebra and its Applications, 2004.

[99] Z. Li, I. Navon, M. Hussaini, and F.-X. Dimet, *Optimal control of cylinder wakes via suction and blowing*, Computers & Fluids, 32 (2003), pp. 149–171.

[100] J. Lions and E. Magnaes, *Non–Homogeneous Boundary Value Problems and Applications*, vol. I-III, Springer–Verlag, 1972.

[101] G. Lube, *Stabilized Galerkin finite element methods for convection dominated and incompressible flow problems*, in Numerical Analysis and Mathematical Modelling, vol. 29 of Banach Center Publications, Institute of Mathematics, Polish Academy of Sciences, Warsaw, 1994, pp. 85–104.

[102] G. Lube and L. Tobiska, *A nonconforming finite element method of streamline diffusion type for the incompressible Navier–Stokes equations*, J. Comput. Math., 8 (1990), pp. 147–158.

[103] H. Ly and H. Tran, *Modeling and control of physical processes using proper orthogonal decomposition*, Mathematical and Computer Modelling, 33 (2001), pp. 223–236.

[104] S. Manservisi, *Optimal Boundary and Distributed Control of the Time Dependent Navier–Stokes Equations*, PhD thesis, Virginia Tech, Blacksburg, VA, 1997.

[105] C. Meyer, *Matrix Analysis and Applied Linear Algebra*, SIAM, Philadelphia, 2000.

[106] R. Mickens, *Difference Equations: Theory and Applications*, Chapman & Hall, New York, 2nd ed., 1990.

[107] S. Müller, A. Prohl, R. Rannacher, and S. Turek, *Implicit time–discretization of the nonstationary incompressible Navier–Stokes equations*, in Proc. 10th GAMM–Seminar, Kiel, January 14–16, 1994, G. Wittum and W. Hackbusch, eds., Vieweg, 1994.

[108] S. Müller-Urbaniak, *Eine Analyse des Zwischenschritt–Θ–Verfahrens zur Lösung der instationären Navier–Stokes Gleichungen*, PhD thesis, Universität Heidelberg, 1993.

[109] L. S. M.W. and Reichelt, *The MATLAB ODE suite*, SIAM J. Sci. Comput., 18 (1997), pp. 1–22.

[110] U. Nävert, *A Finite Element Method for Convection–Diffusion Problems*, PhD thesis, Chalmers University of Technology, Göteborg, 1982.

[111] K. Niijima, *Pointwise error estimates for a streamline diffusion finite element scheme*, Numer. Math., 56 (1990), pp. 707–719.

[112] J. Nocedal and S. Wright, *Numerical Optimization*, Springer, New York, 1999.

[113] K. Ohmori and T. Ushijima, *A technique of upstream type applied to a linear nonconforming finite element approximation of convective diffusion equations*, R.A.I.R.O. Numer. Anal., 18 (1984), pp. 309–332.

[114] J. Peterson, *The reduced basis method for incompressible viscous flow calculations*, SIAM J. Sci. Stat. Comput., 10 (1989), pp. 777–786.

[115] A. Prohl, *Projection and Quasi–Compressibility Methods for solving the Incompressible Navier–Stokes Equations*, Teubner, Stuttgart, 1997.

[116] R. Rannacher, *Finite element solution of diffusion problems with irregular data*, Numer. Math., 43 (1984), pp. 309–327.

[117] ——, *Finite element methods for the incompressible Navier–Stokes equations*, Preprint No. 1999–37, Universität Heidelberg, 1999.

[118] R. Rannacher and S. Turek, *A simple nonconforming quadrilateral Stokes element*, Numer. Meth. Part. Diff. Eqs., 8 (1992), pp. 97–111.

[119] S. Ravindran, *Adaptive reduced–order controllers for a thermal flow system using proper orthogonal decomposition*, SIAM J. Sci. Comput., 23 (2002), pp. 1924–1942.

[120] ——, *Control of flow separation over a forward–facing step by model reduction*, Comput. Methods Appl. Mech. Engrg., 191 (2002), pp. 4599–4617.

[121] M. Reed and B. Simon, *Functional Analysis*, Academic Press, New York, 1980.

[122] H.-G. Roos, M. Stynes, and L. Tobiska, *Numerical Methods for Singularly Perturbed Differential Equations*, no. 24 in Springer Series in Computational Mathematics, Springer, 1996.

[123] P. Schreiber, *A New finite Element Solver for the Nonstationary Incompressible Navier–Stokes Equations in Three Dimensions*, PhD thesis, University of Heidelberg, Heidelberg, Germany, 1996.

[124] F. Shakib, T. Hughes, and Z. Johan, *A new finite element formulation for computational fluid dynamics: X. The compressible Euler and Navier–Stokes equations*, Comp. Methods Appl. Mech. Engrg., 89 (1991), pp. 141–219.

[125] R. Showalter, *Hilbert space methods for partial differential equations.* Electronic Journal of Differential Equations, Monograph 01 (1994), 1994.

[126] L. Sirovich, *Turbulence and the dynamics of coherent structures, Part I: Coherent structures, Part II: Symmetries and transformations, Part III: Dynamics and scaling*, Q. Appl. Math., 45 (1987), pp. 561–571, 573–582, 583–590.

[127] L. Sirovich and R. Everson, *A Karhunen–Loeve analysis of episodic phenomena.* submitted for publication, 1998.

[128] S. Sritharan, *Optimal Control of Viscous flows*, SIAM, Philadelphia, 1998.

[129] R. Temam, *Navier–Stokes Equations: Theory and Numerical Analysis*, AMS Chelsea Publishing, Providence, 3rd ed., 1984.

[130] ——, *Infinite–Dimensional Dynamical Systems in Mechanics and Physics*, Springer–Verlag, New York, 1988.

[131] ——, *Navier–Stokes Equations and Nonlinear Functional Analysis*, SIAM, Philadelphia, 2nd ed., 1995.

[132] T. Tezduyar, *Stabilized finite element formulations for incompressible flow computations*, Advances in Applied Mechanics, 28 (1992), pp. 1–44.

[133] T. Tezduyar, S. Mittal, S. Ray, and R. Shih, *Incompressible flow computations with stabilized bilinear and linear equal–order–interpolation velocity–pressure elements*, Comp. Methods Appl. Mech. Engrg., 95 (1992), pp. 221–242.

[134] T. Tezduyar and Y. Osawa, *Finite element stabilization parameters computed from element matrices and vectors*, Comp. Methods Appl. Mech. Engrg., 190 (2000), pp. 411–430.

[135] L. Tobiska, *Stabilized finite element methods for the Navier–Stokes problem*, in Applications of Advanced Computational Methods for Boundary and Interior Layers, J. Miller, ed., Boole Press, Dublin, 1993, pp. 173–191.

[136] L. Tobiska and G. Lube, *A modified streamline diffusion method for solving the stationary Navier–Stokes equations*, Numer. Math., 59 (1991), pp. 13–29.

[137] L. Tobiska and R. Verfürth, *Analysis of a streamline diffusion finite element method for the Stokes and the Navier–Stokes equations*, SIAM J. Numer. Anal., 33 (1996), pp. 107–127.

[138] P. Toint, *Global convergence of a class of trust–region methods for nonconvex minimization in Hilbert space*, IMA Journal of Numerical Analysis, 8 (1988), pp. 231–252.

[139] S. Turek, *Tools for simulating nonstationary incompressible flow via discretely divergence–free finite element models*, Int. J. Numer. Meth. Fluids, (1994), pp. 71–105.

[140] ——, *On discrete projection methods for the incompressible Navier–Stokes equations: An algorithmical approach*, Comput. Methods Appl. Mech. Engrg., (1997), pp. 271–288.

[141] ——, *FEATFLOW - Finite Element Software for the Incompressible Navier-Stokes Equations: User Manual, Release 1.1*, Universität Heidelberg, 1998.

[142] ——, *Efficient Solvers for Incompressible Flow Problems. An Algorithmic and Computational Approach*, Springer, Berlin, 1999.

[143] M. Ulbrich, *Constrained optimal control of Navier-Stokes flow by semismooth Newton methods*, Systems & Control Letters, 48 (2003), pp. 297–311.

[144] J. Van Kan, *A second–order accurate pressure–correction scheme for viscous incompressible flows*, SIAM J. Sci. Stat. Comp., 7 (1986), pp. 870–891.

[145] S. Volkwein, *Optimal control of a phase–field model using proper orthogonal decomposition*, Z. Angew. Math. Mech., 81 (2001), pp. 83–97.

[146] ——, *Condition number of the stiffness matrices arising in POD Galerkin schemes for dynamical systems.* Submitted, 2004.

[147] ——, *Interpretation of proper orthogonal decomposition as singular value decomposition and HJB–based feedback design*, in Proceedings of the Sixteenth International Symposium on Mathematical Theory of Networks and Systems (MTNS), Leuven, Belgium, July 5–9, 2004, 2004.

[148] K. Willcox, *Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition*, in Proceedings of the 5th SMA Symposium, January 2004, AIAA Fluid Dynamics Conference and Exhibit, 2004.

[149] K. Willcox and J. Peraire, *Balanced model reduction via the proper orthogonal decomposition*, Techreport 2001–2611, AIAA, 2001.

[150] G. Zhou, *How accurate is the streamline diffusion finite element method?*, Math. Comp., 66 (1997), pp. 31–44.