Trust-region Methods for Flow Control based on Reduced Order Modelling

Dissertation

zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften

dem Fachbereich IV der Universität Trier
vorgelegt von

Marco Fahl

Trier, 2000

Eingereicht am 14. Dezember 2000

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

Tag der mündlichen Prüfung: 9. Februar 2001

Danksagung

An dieser Stelle möchte ich mich bei allen bedanken, die zum Gelingen dieser Dissertation beigetragen haben.
Mein besonderer Dank gilt Herrn Prof. E. Sachs, der durch seine Betreuung und Unterstützung dieser Arbeit in fachlichen als auch persönlichen Belangen einen massgeblichen Beitrag geleistet hat. Des weiteren danke ich Herrn Prof. J.A. Burns für die Übernahme des Zweitgutachtens, sowie Frau Prof. B.B. King und Herrn Prof. E. Arian für zahlreiche, konstruktive Diskussionen.
Für die finanzielle Unterstützung dieser Dissertation durch ein Stipendium der Deutschen Forschungsgemeinschaft im Rahmen des Graduiertenkollegs *Mathematische Optimierung* an der Universität Trier möchte ich mich gleichfalls bei den Trägern des Graduiertenkollegs bedanken.
Dank gilt auch meinen Kollegen im Graduiertenkolleg für unzählbare Diskussionen, sowohl über mathematische Fragestellungen als auch über die einfachen Dinge des Lebens: L. Abbe, P. Justen, C. Schwarz und T. Voetmann.
Mein grösster Dank gebührt schliesslich meiner lieben Michi für ihr Verständnis und ihre fortwährende Unterstützung meiner Arbeit, sowie meinen Eltern und meiner ganzen Familie.

Contents

Chapter 1

Introduction

1.1 Motivation

Control and optimization of systems or processes in industry is a challenging field, since appropriate mathematical modelling often leads to optimization problems where the governing equations are partial differential equations. Accordingly, optimization problem formulations in all areas of engineering sciences (e.g. geo sciences, chemical processing, aerospace industry) usually involve partial differential equations that describe complex phenomena. If so, standard discretization schemes for the spatial discretization lead to sparse but very large-scale, and in general, nonlinear systems of ordinary differential equations. These serve to describe the solution of the state equation and are characterized by a high computational complexity while being solved. For this reason, these large-scale systems are often not suited to be used during the application of an optimization method, where they have to be solved repeatedly.

A prime example for this kind of computational difficulties is flow control of the time-dependent Navier-Stokes equations, which is an active research area. For a comprehensive survey of applications and flow control concepts in aerodynamics, hydrodynamics and other disciplines, we refer to Gunzburger [43], Gad-el-Hak et al. [32] or Sritharan [104].

Although many important theoretical questions on flow control (existence of optimal controls, optimality conditions etc.) have been investigated in, e.g., Abergel/Temam [1], Fattorini/Sritharan [35, 36] or Fursikov et al. [38], the numerical solution of many of these control problems is still intractable. They only come into reach for being solved due to increasing computer capacity, but with excessive computing times.

On the one hand, accurate finite element or finite difference discretizations should be used to resolve flow patterns sufficiently. Especially for either three-dimensional flows on complex domains and turbulent flows these discretization techniques lead to such high-dimensional systems of ordinary differential equations mentioned above. Finite element meshes with more than 10^8 grid points are common in many real-life applications, cf.

Bewley et al. [15]. On the other hand, the computation of a control problem solution with such highly resolved state equations, in general, is not practicable because of limited computing power.

In order to overcome this problem, there is a demand to use reduced order models that serve as low-dimensional approximation models to the large-scale discretized state equations. While standard spatial discretization approaches, based on finite elements or finite differences, employ standardized discretization schemes (e.g., linear or quadratic local shape functions, 5-point stencils) that are suitable for large classes of problems, reduced order modelling approaches typically use a more problem specific discretization. In this case, detailed information on the underlying problem is employed in order to achieve a dimension reduction. This, of course, requires suitable input data.

Usually, a reduced order model for the Navier–Stokes equations is constructed for a specific flow configuration, e.g., for an uncontrolled flow, such that it is a suitable model for representing the corresponding flow dynamics. If the reduced order modelling idea is applied to a flow control setting, it remains an open question whether a once derived reduced order model will provide useful approximations to flow dynamics altered by a control. This potential insufficiency is one of the main problems in the application of reduced order modelling techniques to control problems. As a matter of fact, working with a low-dimensional model during the computation of a control problem solution has to face the problem that these models are possibly unreliable models if they are not or not correctly updated during the optimization process.

Consequently, some sort of iterative technique is required, in which the construction of reduced order models is coupled with the progression of the optimization process, cf. Bewley et al. [15]. Such an approach leads to the use of reduced order models, that adequately represent the flow dynamics as altered by the control. Crucial at this point is to decide whether or not a reduced order model has to be adapted to a new flow configuration.

A trust-region approach to manage this problem is the method of choice. Since we are interested in solving an optimal control problem with a specified cost functional, we can monitor the reduced order model's quality by comparing the performance of a reduced order solution to the performance of the original large-scale solution with regard to the optimization goal. In using a trust-region method, the range of validity of a reduced order model is automatically restricted and the required update decision for the reduced order models can be made by employing information that is obtained during the control problem solution. By means of this approach, we can benefit from the use of reduced order models with respect to an efficient computation of an approximate solution of the state equations, while the uncertain reliability of such reduced order solutions is eliminated.

From this point of view, this thesis is concerned with the investigation of trust-region methods in the context of reduced order modelling for the solution of flow control prob-

lems governed by the Navier–Stokes equations. Here, we introduce the proper orthogonal decomposition (POD) as reduced order modelling technique. As far as we know, it is the first time that POD based reduced order modelling for flow control is analyzed using a trust-region framework.

1.2 Outline

An outline of the contents of the present work with references to related literature is given in the following.

In Chapter 2 we start with the Dirichlet boundary control problem for fluid flows governed by the Navier-Stokes equations for viscous, incompressible flows

$$Minimize \qquad J(y,u)$$

subject to

$$\begin{aligned} \tfrac{\partial}{\partial t} y - \nu\,\Delta y + (y\cdot\nabla)\,y + \nabla p &= f \qquad \text{in} \quad \Omega\times(0,T) \\ div\, y &= 0 \qquad \text{in} \quad \Omega\times(0,T) \end{aligned}$$

with appropriate initial and boundary conditions, $y(0)=y_0$ on Ω, $y=g(u)$ on $\partial\Omega\times(0,T)$, where y denotes the unknown velocity field and p the unknown pressure field. Here, our main focus is on boundary control. Consequently, the control variable u enters in the boundary conditions of the Navier–Stokes equations.

In order to give a problem formulation that makes use of the concept of weak solutions, we first review an appropriate functional framework for the Navier–Stokes equations. This leads to variational formulations according to Temam [108, 109] and Girault/Raviart [40]. Also, classical results concerning the existence and uniqueness of a weak solution of the Navier–Stokes equations are introduced.

Eventually, we define a variational formulation for the Navier–Stokes equations with boundary control, present cost functionals that are typically employed for flow control and give the final control problem formulation. We close this theoretical part on flow control with results from Fattorini/Sritharan [35] concerning the existence and uniqueness of a weak solution of the Navier–Stokes equations with boundary control and the existence of an optimal control.

Since the driven cavity problem serves as model problem for an illustration of numerical results throughout this work, we introduce this model problem in the last section of the second chapter.

Based on the presented variational formulations, the discretization of the Navier–Stokes equations can be considered where we are interested in employing a reduced order modelling technique.

Hence, a detailed introduction to the proper orthogonal decomposition for reduced order modelling of the Navier–Stokes equations is given in Chapter 3.

A finite element discretization of a variational formulation of the Navier–Stokes equations with a fine mesh to resolve flow patterns sufficiently yields a system of ordinary differential equations, which is often prohibitive large to be used in an optimization context. A remedy for this difficulty are reduced order models. Here, the goal is to find a simplified model that describes the state equations approximately while it also guarantees a fast and efficient computation of a reduced order solution. For this purpose, we employ a variant of the proper orthogonal decomposition that is based on snapshots of the governing equation.

The snapshot proper orthogonal decomposition takes flow solutions at fixed time instances as input data and leads to the computation of a, with respect to the order of the finite element discretization, small number of POD basis functions. Consequently, a discretization of the variational formulation of the Navier–Stokes equations with these POD basis functions yields a low-dimensional system of ordinary differential equations (the reduced order model) for describing the flow behaviour. In this context, the POD basis functions can be seen as global basis functions. Contrasting finite element basis functions (which are local basis functions), they are related to the flow behaviour on the whole flow domain.

In the third chapter, we start with a short introduction to the proper orthogonal decomposition as it can be found in the literature in the context of turbulent flows, cf. Sirovich [101] or Berkooz et al. [14].

Afterwards, we review the close relationship between the proper orthogonal decomposition and the singular value decomposition as a main tool for data analysis, cf. Kunisch/Volkwein [72] or Atwell/King [10]. Here, we derive a problem formulation for the computation of the POD basis functions that exploits these similarities. As a consequence, the POD basis can be computed by means of a truncated singular value decomposition of the data matrix that accumulates the snapshot information.

Furthermore, we give a review of the essential properties of the POD basis functions, in order to illustrate their attractiveness for reduced order modelling, especially for the Navier–Stokes equations. It is important to note, that the POD basis functions, in some sense, provide an optimal basis for the representation of the flow dynamics under consideration. Additionally, properties like incompressibility or no-slip boundary conditions pass from the input snapshots to the POD basis. We also give some comments on the choice of the dimension of the POD basis.

Besides that, we finally give a detailed derivation of the POD based reduced order models for the Navier-Stokes equations with and without boundary control, and summarize the steps that lead to the computation of a reduced order solution. For this purpose, we have to discretize the variational formulation for the Navier–Stokes equations using the POD

basis functions. Here, our approach to the reduced order model with Dirichlet boundary control follows the techniques presented in Tang et al. [107]. Similar approaches for flow control have been investigated in Ito/Ravindran [56], Ravindran [95] or Hinze/Kunisch [51].

After having computed a reduced order model corresponding to a specific flow control configuration, it can be expected that the same reduced order model is also a suitable model for other similar control forcings. Yet, the range of its validity is not clearly determined. Thus, this approach is of particular interest for optimization purposes. It allows for a fast and efficient evaluation of the state equations during the run of an optimization algorithm for the solution of a control problem. We close this chapter with numerical examples for the simulation of fluid flows. These examples serve to present typical features of the POD based model reduction technique. Here, we compare, e.g., the error between a POD based reduced order solution and the original input snapshots. Eventually, the limited reliability of a once derived reduced order model is also illustrated.

As pointed out above, the POD basis functions can be computed by a truncated singular value decomposition of a snapshot data matrix. Since in many applications an appropriate spatial discretization of the Navier–Stokes equations leads to large-scale data matrices, an efficient procedure for the computation of a truncated singular value decomposition is required. Having the efficiency of Krylov subspace methods in mind, we therefore propose to use a Lanczos method for the POD basis computations and analyze this technique in the subsequent chapter.

In Chapter 4 we present a Lanczos method for the computation of the POD basis functions.

For this purpose, we review the basic ideas and properties of the Lanczos iteration following Parlett [87], Cullum/Willoughby [24], and especially concentrate on its convergence behaviour based on results of Saad [97].

Eventually, a Lanczos-type algorithm is presented, tailored to specific aspects of the POD basis computation. Furthermore, we compare its efficiency to other approaches for the computation of POD basis functions. We show that this POD tailored Lanczos method is numerically efficient as long as large-scale snapshot data matrices are involved and the number of wanted POD basis functions remains small.

We close this chapter by giving some numerical examples that confirm our theoretical considerations.

Starting from the considerations in the preceding chapters, we may come then to the main part of this thesis. Here, we are concerned with the application of POD based reduced order modelling for flow control in the context of trust-region frameworks.

Since the presented reduced order models are based on a small number of POD basis functions extracted from a specific input ensemble, the expansion of a flow solution in this

basis provides a small number of degrees of freedom only. Consequently, a single POD based reduced order model is especially suited for representing the input flow and flows that are of similar dynamics. In general, a single reduced order model is not suited to represent a flow behaviour with essentially different dynamics. This property obviously contradicts the requirement of working with a reduced order model producing reliable flow solutions corresponding to a variety of possible controls during the control problem solution. As a consequence, it is inevitable to update the POD based reduced order models during the optimization process. For this reason, we give a comprehensive description how trust-region methods can be used to manage the required update decision successfully.

In Chapter 5, we introduce trust-region methods for the solution of unconstrained optimization problems.

Such a trust-region approach provides a tool for working with approximate objective functions, the model functions, during the iterative solution of an unconstrained optimization problem of the type

$$\min_{u \in \mathbb{R}^n} \; f(u)$$

by means of solving appropriate subproblems successively. Here, the model function m_k at a given current iterate $u_k \in \mathbb{R}^n$ is chosen such that the trust-region subproblem

$$\min_{s \in \mathbb{R}^n} \; m_k(u_k+s), \qquad \|s\| \leq \delta_k,$$

for $\delta_k > 0$, is easier to solve than the original problem. As long as the optimization of the model function, m_k, subject to the trust-region constraint, $\|s\| \leq \delta_k$, provides steps s_k that yield progress towards the optimization goal described by f, we trust the model function to model the true objective appropriately. In this case, we accept the predicted steps s_k. But, if the model function predicts a step s_k that provides no sufficient reduction in the original objective, we reject this step. During the optimization process, the model functions are updated such that good local approximation properties with respect to the actual objective are preserved. Moreover, the trust-region constraint can be modified after each iteration in order to characterize the range of validity of the next model function. A comprehensive survey of trust-region methods can be found in the recent monograph by Conn et al. [23].

In the fifth chapter, we start with a presentation of the standard trust-region approach based on quadratic model functions,

$$m_k(u_k+s) = f(u_k) + \nabla f(u_k)^T s + \tfrac{1}{2} s^T H_k s,$$

for the true objective f. Here, we give a review on the essential ideas and properties of a corresponding trust-region algorithm based on the work of Powell [89], Moré [80] and Dennis/Schnabel [27]. At this point, it is important to note that the global convergence result for this algorithm,

$$\lim_{k \to \infty} \; \|\nabla f(u_k)\| = 0,$$

can be derived by means of the concept of Cauchy decrease, where it is essential that a quadratic model function for f is employed.

We also introduce an extension of the standard trust-region method with quadratic model functions. Instead of the true gradient, $\nabla f(u_k)$, of the actual objective function, it is possible to employ approximate derivative information, g_k, to build a quadratic model for f:

$$m_k(u_k+s) = f(u_k) + g_k^T s + \tfrac{1}{2} s^T H_k s.$$

This idea has been studied, e.g., in Carter [20, 21].

An important aspect in this context is the error control in the gradient approximation. If this error behaves well, the theoretical results for the standard trust-region approach based on quadratic model functions can be transferred to the case where only inexact gradient information is available. In this context, we also present and discuss several error conditions on the gradient error that can be found in the literature.

Finally, we introduce a trust-region approach where general model functions for f can be used. Following the work of Toint [111], no assumptions on the nonlinearity of $m_k(u_k+s)$ are imposed. Merely, the solution of the trust-region subproblem with a general model function m_k requires some special treatment. As in the described quadratic model function case, it is necessary to use a solution method that generates steps satisfying a fraction of Cauchy decrease condition. Therefore, we present the step determination algorithm proposed in Toint [111] for the solution of the trust-region subproblem that meets this criterion. Furthermore, we review its essential properties in order to apply these results in the convergence proofs of the main algorithm in the following chapter.

In Chapter 6, we consider a trust-region framework for flow control that combines the POD based reduced order modelling technique with the trust-region philosophy.

By this means, we can employ reduced order models for the computation of an approximate solution of the Navier–Stokes equations, while the reliability of these reduced order solutions is guaranteed. At this point, it is crucial to note that the accuracy of the reduced order solution is not the most important aspect in the flow control context. In fact, employing a reduced order model it is more important to obtain some information on how the control should be altered in order to get closer to the optimization goal. Since the optimization goal is expressed in the cost functional, its obvious to reconsider reduced order modelling in the trust-region context. By defining a model function (based on the reduced order solution) for the cost functional, we are able to compare the performance of a reduced order solution to the performance of the full order solution.

This requires a reformulation of the flow control problem as an unconstrained optimization problem. In this case, we view the solution of the state equation as an implicitly defined function of the (discretized) control, $u \in \mathbb{R}^n$. Supposing that a high-order finite element solution is denoted by $y^N(u)$, the reduced order solution by $y^M(u)$ and the

discretized cost functional by $\hat{f}$, we may define

$$f(u) = \hat{f}(y^N(u), u)$$

and

$$m_k(u_k+s) = \hat{f}(y^M(u_k+s), u_k+s)$$

for a given current control iterate $u_k \in I\!R^n$. In that the state equation, i.e. the Navier–Stokes equations, is approximated by a POD based reduced order model with a certain approximation error, an error in the objective function occurs. Then, we can interpret the evaluation of the true objective function with an approximate state equation solution as the evaluation of a model function for the actual objective. Consequently, employing a trust-region algorithm with model functions, m_k, as given above also provides us with a mechanism for deciding whether or not a POD based reduced order model should be updated during the course of the optimization. Because, an update of the model function m_k also implies an update of the POD based reduced order model.

This philosophy of combining trust-region methods with general modelling issues is a well-known technique in multidisciplinary design optimization, see Alexandrov/Hussaini [5], also known as surrogate optimization, see Booker et al. [16], Serafini [99] or Alexandrov et al. [6]. For this reason, we give an introduction to surrogate optimization and review specific approaches to the solution of trust-region subproblems in the surrogate optimization context. This review particularly serves to mark the similarities and the differences between the approach presented in this thesis and approaches that can be found in the multidisciplinary design optimization literature. While surrogate optimization bases either on derivative-free direct search methods or once more on quadratic model functions, we adapt the approach of Toint [111] for general, nonlinear model functions.

The main part of Chapter 6 serves to present an algorithm that implements the combination of POD based reduced order modelling and trust-region methods, the TRPOD algorithm. Typically, the gradients of the POD based model functions are computed via an adjoint equation or sensitivity equations approach for the reduced order models. This implies that a POD based model function also provides only approximate gradient information, $g_k = \nabla m_k(u_k)$, to the true gradient, $\nabla f(u_k)$. Thus, as in the quadratic model function case with inexact gradients, a gradient error condition is required in order to derive a convergence theory for the TRPOD algorithm. For this purpose, we choose the error condition

$$\frac{\|\nabla f(u_k) - g_k\|}{\|g_k\|} \leq \zeta,$$

for $\zeta \in (0, 1)$, given in Carter [20]. By combining results of both Carter [20] and Toint [111], we are finally able to prove the convergence result

$$\lim_{k\to\infty} \|\nabla f(u_k)\| = 0$$

for the TRPOD algorithm. In order to achieve this, the above error condition has to be satisfied, where the parameter ζ is coupled with some quantities in the trust-region algorithm. We close the basic TRPOD algorithm section by discussing some further modifications of the algorithm that can lead to an improved performance.

Since it is, in general, questionable whether the gradient error condition can be satisfied, we also present a modification of the basic TRPOD algorithm that does not depend on an error condition. This modification benefits from results presented in Chang et al. [22] and Alexandrov et al. [6]. The key to this problem is to replace the model function m_k by a second model function a_k. This model function a_k can be derived as a scaled version of the model function m_k, provided that the gradient of f itself is available. Obviously, a TRPOD implementation with scaled model functions requires an additional numerical effort for the computation of the true gradient, ∇f. Therefore, we close this section with the discussion of a hybrid algorithm that combines the basic TRPOD algorithm and the TRPOD algorithm with scaled model functions. As long as the basic TRPOD algorithm provides sufficiently accurate gradients, this approach should be used. In case the performance of the basic TRPOD algorithm indicates a deterioration of the gradient information, a switch-over to the TRPOD algorithm with scaled model functions should be made.

To round the contents of this chapter we also supply numerical results for the TRPOD algorithm applied to the control of cavity flows. In addition, we also present a comparison of the performance of the basic TRPOD algorithm with the hybrid TRPOD algorithm (with scaled model functions) when applied to a control problem governed by heat transfer.

In Chapter 7 we outline a trust-region proper orthogonal decomposition modelling concept for industrial application in food processing. Here, we consider the in-container sterilization of food products by thermal processing in an autoclave, cf. Kleis [69], Justen [61], Kleis/Sachs [70].

Presenting an optimal control problem that models the goals of the food sterilization process, an appropriate heat transfer model is required that can be used to represent the evolution of the temperature inside the container.

We introduce different approaches of various complexity for modelling the heat transfer inside the food container which are used in practice and which can be found in the literature. Starting from a simple empirical model (the Ball-Formula), we illustrate that suitable partial differential equations provide more accurate methods for modelling the heat transfer for food sterilization purposes. Depending on the type of food product that is to be heated in the autoclave, we obtain heat transfer models that involve the nonlinear heat equation or even the Navier-Stokes equations.

But, with increasing complexity of the heat transfer model also the computational complexity for the solution of the corresponding control problem increases. Here, we discuss a reduced order modelling idea that can be applied in case the heat transfer model consists of the Navier–Stokes equations coupled with a convection-diffusion equation. For

this type of problem it is attractive to replace the fluid flow part in the heat transfer model by a POD based reduced order model according to the previous discussion.

Closing this work, we finally give a brief summary of the most essential aspects concerning the application of trust-region methods involving POD based reduced order modelling for control problems.

Appendix A contains an outline of the solution concepts of the finite element flow solver FEATFLOW.

Since the presented POD based reduced order modelling approach requires appropriate input data for the computation of the POD basis functions, we employ the flow solver FEATFLOW [113, 115] for the numerical simulation of the Navier–Stokes equations in order to obtain snapshots of a specific flow configuration.

In the appendix we consider several techniques that have been used for the implementation of FEATFLOW and that can be seen as typical ingredients of a finite element flow solver. Our presentation includes the spatial discretization via finite elements, the time discretization scheme, the decoupling of velocity and pressure variables and the solution procedures for systems of linear and nonlinear equations.

The description of these solution concepts also illustrates that solving the Navier-Stokes equations itself is a difficult task that requires sophisticated techniques and computing power.

Chapter 2

Formulation of the Control Problem

2.1 Introduction

In this chapter we introduce the Dirichlet boundary control problem for fluid flows governed by the Navier-Stokes equations for viscous, incompressible flows that shall be investigated in a reduced order modelling approach.

A description of the state equations of the control problem is given in Section 2.2.1 and the functional framework that is necessary to derive variational formulations of the Navier-Stokes equations is described in Section 2.2.2. In Section 2.3 we present cost functionals that have proven to be useful in a flow control setting. Based on this material we are able to formulate the optimal control problem that represents the starting point for the research results of the present work. We introduce the driven cavity problem serving as model problem for the illustration of numerical results in this thesis in Section 2.4.

Related literature on optimal control problems for the Navier-Stokes equations covers a broad spectrum of topics, see Gunzburger [43], Sritharan [104], Gad-el-Hak et al. [32]. Flow control problems in an optimal control theory context can be found in, e.g., Desai/Ito [29], Heinkenschloss [47], Hou et al. [53, 54], where boundary control problems for the stationary Navier-Stokes equations are investigated. The evolutionary Navier-Stokes equations with distributed control and/or boundary controls are addressed in, e.g., Abergel/Temam [1], Fursikov et al. [38], Gunzburger/Manservisi [45], Fattorini/Sritharan [35], and numerical results related to such approaches are documented in, e.g., Berggren [13], Hinze/Kunisch [50] or Bewley et al. [15].

2.2 The State Equation

2.2.1 The Navier-Stokes Equations

Let $\Omega \subset I\!R^2$. We consider the time-dependent Navier-Stokes equations (NSE) for a viscous, incompressible fluid flow in two dimensions

$$\begin{aligned} \frac{\partial}{\partial t} y(x,t) - \nu\, \Delta y(x,t) + (y(x,t)\cdot\nabla)\, y(x,t) + \nabla p(x,t) &= f(x,t) && (2.1)\\ div\, y(x,t) &= 0 && (2.2) \end{aligned}$$

for all $x \in \Omega$, $t \in (0,T)$, with initial condition

$$y(x,0) = y_0(x), \qquad x \in \Omega, \quad y_0 \text{ given} \tag{2.3}$$

and Dirichlet boundary conditions

$$y(x,t) = g(x,t;u(t)), \qquad x \in \Gamma, t \in (0,T) \tag{2.4}$$

that include the control input, u. We assume that Ω is an open bounded domain with sufficiently smooth boundary $\Gamma = \partial\Omega$ and $(0,T)$ with $T > 0$ is the time interval of interest. The two-dimensional velocity field is denoted by y and p is the scalar pressure field. Furthermore, f denotes body forces and the parameter $\nu > 0$ denotes the reciprocal value of the Reynolds number $\nu = 1/Re$, which is a measure for the fluids viscosity.

Here, the pressure field, p, and the velocity field, y, in the *momentum equation* (2.1) are coupled through the *incompressibility constraint* (or *mass equation*) (2.2), where

$$div\, y = \frac{\partial}{\partial x_1} y_1 + \frac{\partial}{\partial x_2} y_2 = 0 \qquad \text{in } \Omega \times (0,T)$$

describes the conservation of mass. The convective part, $(y\cdot\nabla y)\, y$, in (2.1) is defined by

$$(y\cdot\nabla)\, y = \begin{pmatrix} y_1 \frac{\partial}{\partial x_1} y_1 + y_2 \frac{\partial}{\partial x_2} y_1 \\ y_1 \frac{\partial}{\partial x_1} y_2 + y_2 \frac{\partial}{\partial x_2} y_2 \end{pmatrix}.$$

The equations of mass and momentum for the fluid flow are complemented by the initial condition (2.3) and boundary conditions, $y(x,t) = g(x,t;u(t))$ for $x \in \Gamma, t \in (0,T)$, where we assume that a control, $u(t)$ for $t \in (0,T)$, acts on the boundary and thus influences the flow behaviour inside the domain, Ω.

Concerning the boundary conditions, we assume that the boundary of the domain, Γ, can be split into two parts such that Γ_c denotes that part of the boundary where the control is applied and $\Gamma \setminus \Gamma_c$ is the part of the boundary where the boundary data is not controlled. To be more precise, we assume that the boundary conditions can be written as

$$g(x,t;u(t)) = \begin{cases} u(t)b(x), & x \in \Gamma_c,\ t \in (0,T) \\ c(x), & x \in \Gamma \setminus \Gamma_c,\ t \in (0,T) \end{cases} \tag{2.5}$$

i.e., we consider boundary conditions on the control boundary, Γ_c, where the temporal dependence and the spatial dependence are separated. We interpret $b(x),\ x \in \Gamma_c$, as a prescribed control action such that by the choice of $u(t),\ t \in (0,T)$, a temporal variation of the control action can be described.

A typical example for such a setting is flow control by blowing and suction along a portion of the boundary, cf. e.g. Fattorini/Sritharan [35], Joslin et al. [59], Berggren [13].

2.2.2 Function Spaces and Variational Formulations

For the presentation of a variational formulation for the Navier-Stokes equations we first restrict ourselves to the special case

$$g(x,t) = 0, \quad x \in \Gamma, t \in (0,T) \tag{2.6}$$

and adapt the notation of Temam [108, 109] and Girault-Raviart [40]. This requires the introduction of some function spaces, see Adams [2]. We denote by $L^2(\Omega)^d$ the space of square integrable functions from Ω into $\mathbb{R}^d$,

$$L^2(\Omega)^d = \{\phi : \Omega \to \mathbb{R}^d \mid \sum_{i=1}^{d} \int_\Omega \phi_i^2(x)\, dx < \infty\},$$

and set

$$\begin{aligned}
H^1(\Omega)^d &= \{\phi \in L^2(\Omega)^d \mid \tfrac{\partial}{\partial x_i}\phi \in L^2(\Omega)^d \quad \text{for } i = 1,\dots,d\},\\
H_0^1(\Omega)^d &= \{\phi \in H^1(\Omega)^d \mid \phi = 0 \quad \text{on } \Gamma\},\\
L_0^2(\Omega) &= \{\phi \in L^2(\Omega)^1 \mid \int_\Omega \phi(x)\, dx = 0\},\\
\mathbf{H} &= \{\phi \in L^2(\Omega)^d \mid div\, \phi = 0,\ (\phi \cdot \mathbf{n})|_\Gamma = 0\},\\
\mathbf{V} &= \{\phi \in H_0^1(\Omega)^d \mid div\, \phi = 0\},\\
L^2(0,T;X) &= \{w : (0,T) \to X \mid \int_0^T \|w\|_X^2\, dt < \infty\},
\end{aligned}$$

where $\mathbf{n}$ denotes the unit outward normal on Γ. $L^2(\Omega)^d$ endowed with the usual inner product and norm

$$(\psi,\phi)_{L^2} = \sum_{i=1}^{d} \int_\Omega \psi_i(x)\, \phi_i(x)\, dx, \qquad \|\phi\|_{L^2} = (\phi,\phi)_{L^2}^{1/2} \tag{2.7}$$

is a Hilbert Space. In the context of the Navier-Stokes equations it denotes the space of finite kinetic energy velocity fields on Ω. The Sobolev space $H^1(\Omega)^d$ which consists of square integrable vector functions with square integrable gradients (in the sense of weak

derivatives), is a Hilbert space for the inner product and norm

$$(\psi,\phi)_{H^1} = (\psi,\phi)_{L^2} + \sum_{i=1}^{d}\sum_{j=1}^{d}\int_\Omega \tfrac{\partial}{\partial x_i}\psi_j(x)\ \tfrac{\partial}{\partial x_i}\phi_j(x)\ dx, \qquad \|\phi\|_{H^1} = (\phi,\phi)_{H^1}^{1/2}. \tag{2.8}$$

The spaces L_0^2, $\mathbf{H}$, $\mathbf{V}$ are also Hilbert spaces with respect to the inner product and norm in (2.7) and (2.8), respectively. Finally, we endow $L^2(0,T;X)$ with the inner product and norm

$$(v,w)_{L^2(0,T;X)} = \int_0^T (v,w)_X\ dt, \qquad \|w\|_{L^2(0,T;X)} = (w,w)_{L^2(0,T;X)}^{1/2}$$

such that $L^2(0,T;X)$ is also a Hilbert space, if X is a Hilbert space.

On the basis of these function spaces it is possible to obtain variational formulations of the Navier-Stokes equations. In what follows we concentrate on the case $d = 2$ and therefore we drop the corresponding index in the notation of the function spaces.

We define the standard bilinear and trilinear forms

$$\begin{aligned}
\mathbf{a}(\psi,\phi) &= \sum_{i=1}^{2}\sum_{j=1}^{2}\int_\Omega \tfrac{\partial}{\partial x_i}\psi_j(x)\ \tfrac{\partial}{\partial x_i}\phi_j(x)\ dx \qquad \text{for all } \psi,\phi \in H^1(\Omega),\\
\mathbf{c}(\psi,\phi,\varphi) &= \sum_{i=1}^{2}\sum_{j=1}^{2}\int_\Omega \psi_i(x)\ \tfrac{\partial}{\partial x_i}\phi_j(x)\,\varphi_j(x)\ dx \qquad \text{for all } \psi,\phi,\varphi \in H^1(\Omega).
\end{aligned}$$

The following variational formulation is obtained by multiplication of (2.1)–(2.3), (2.6) by a test function $\phi\in\mathbf{V}$ and integration over Ω, where the function $t \mapsto y(t)$ takes its values in $\mathbf{V}$, see Temam [109].

Problem 2.1 (Variational formulation (I))
For given data f, y_0, find $y\in L^2(0,T;\mathbf{V})$ that satisfies

$$\begin{aligned}
\tfrac{\partial}{\partial t}(y,\phi)_{L^2} + \nu\,\mathbf{a}(y,\phi) + \mathbf{c}(y,y,\phi) &= (f,\phi)_{L^2} && (2.9)\\
y(0) &= y_0 && (2.10)
\end{aligned}$$

for all $\phi \in \mathbf{V}$.

It is important to note that in the variational formulation (I) the pressure drops out. Since we embedded the incompressibility constraint into the test function space, integration by parts leads to

$$(\nabla p,\phi)_{L^2} = -(p, div\,\phi)_{L^2} = 0 \qquad \text{for all } \phi \in \mathbf{V}. \tag{2.11}$$

Definition 2.2 (Weak Solution)
Let us consider the variational formulation (I) for given f, y_0. A function $y\in L^2(0,T;\mathbf{V})$ that satisfies (2.9), (2.10) is called a *weak solution* of (2.1), (2.2), (2.3) and (2.6).

According to Temam [109], we have the following result on the existence and uniqueness of a weak solution for the Navier-Stokes equations with homogeneous Dirichlet boundary conditions. Using stronger regularity assumptions on the data involved, yields even more regular solutions, see Temam [108, 109].

Theorem 2.3 (Temam [109], Th. 3.1)
Let $\mathbf{V}'$ *denote the dual space of* $\mathbf{V}$. *Let* $f \in L^2(0,T;\mathbf{V}')$ *and* $y_0 \in \mathbf{H}$ *be given.*

Then, there exists a unique weak solution $y \in L^2(0,T;\mathbf{V})$ *of (2.1), (2.2), (2.3) and (2.6), and* $y \in C([0,T];\mathbf{H})$.

From a practical point of view variational formulation (I) includes some difficulties. Since we embedded the incompressibility condition into the function space $\mathbf{V}$, a discretization of the variational formulation (I) requires to use solenoidal finite element test functions. In order to avoid this, the following alternative variational formulation given in Problem 2.4 allows to treat the incompressibility constraint separately, cf. e.g. Girault/Raviart [40] or Gunzburger/Manservisi [45].

For this purpose, we additionally define the bilinear form, cf. (2.11),

$$\mathbf{b}(\phi,\psi) = -\int_\Omega \psi(x)\, div\, \phi(x)\, dx \qquad \text{for all } \phi \in H^1(\Omega),\ \psi \in L^2(\Omega). \tag{2.12}$$

Problem 2.4 (Variational formulation (II))
For given data f, y_0, find a pair $(y,p) \in L^2(0,T;H_0^1(\Omega)) \times L^2(0,T;L_0^2(\Omega))$ that satisfies

$$\begin{aligned} \tfrac{\partial}{\partial t}(y,\phi)_{L^2} + \nu\, \mathbf{a}(y,\phi) + \mathbf{c}(y,y,\phi) + \mathbf{b}(\phi,p) &= (f,\phi)_{L^2} && (2.13)\\ \mathbf{b}(y,\psi) &= 0 && (2.14)\\ y(0) &= y_0 && (2.15) \end{aligned}$$

for all $\phi \in H_0^1(\Omega),\ \psi \in L_0^2(\Omega)$.

Since the Navier-Stokes system (2.1), (2.2), (2.3) and (2.6) lacks any initial or boundary conditions for the pressure, usually p is only determined up to an additive constant. The variational formulation (II) yields a unique pressure $p \in L^2(0,T;L_0^2(\Omega))$ due to the additional requirement $\int_\Omega \psi(x)\, dx = 0$ for all functions $\psi \in L_0^2(\Omega)$.

2.3 The Control Problem

A variational formulation of the Navier-Stokes equations with nonhomogeneous Dirichlet boundary conditions according to (2.1)–(2.3), (2.5) can be derived similar to the manner described so far.

This requires to specify the state space $\mathbf{Y}$ for the flow fields and the control space $\mathbf{U}$ for the control u and to impose appropriate assumptions on the spatial control action

$b(x),\ x \in \Gamma_c$, and $c(x),\ x \in \Gamma \setminus \Gamma_c$, cf. (2.5). According to Theorem 2.3 and following Abergel/Temam [1], Fattorini/Sritharan [35], we set

$$\mathbf{Y} = L^2(0,T;H^1(\Omega)), \qquad \mathbf{U} = H^1(0,T) \quad \text{and} \quad b \in H_0^{3/2}(\Gamma_c),\ c \in H_0^{3/2}(\Gamma \setminus \Gamma_c).$$

Here, the trace space $H^{3/2}(\Gamma)$ denotes the restriction of functions in $H^2(\Omega)$ to the boundary Γ. Note that we use $b \in H_0^{3/2}(\Gamma_c)$ instead of $b \in H^{3/2}(\Gamma_c)$ in order to ensure compatibility of the boundary data on adjacent boundary parts, cf. e.g. Abergel/Temam [1], Berggren [13] or Heinkenschloss [47].

Similar to Problem 2.1, a variational formulation for the Navier-Stokes equations (2.1)–(2.3) with Dirichlet boundary control (2.5) can be defined as follows, cf. Girault/Raviart [40] and Desai/Ito [29].

Problem 2.5 (Variational formulation with Dirichlet Boundary Control)
For given data f, y_0, b, c, u, find $y \in L^2(0,T;H^1(\Omega))$ that satisfies

$$\begin{aligned} \tfrac{\partial}{\partial t}(y,\phi)_{L^2} + \nu\, \mathbf{a}(y,\phi) + \mathbf{c}(y,y,\phi) &= (f,\phi)_{L^2} \qquad \text{for all } \phi \in \mathbf{V} && (2.16)\\ y(0) &= y_0 && (2.17) \end{aligned}$$

and

$$\begin{aligned} div\ y &= 0 && \text{in } \Omega \times (0,T)\\ y(x,t) &= u(t)b(x), && x \in \Gamma_c,\ t \in (0,T)\\ y(x,t) &= c(x), && x \in \Gamma \setminus \Gamma_c,\ t \in (0,T). \end{aligned}$$

According to Definition 2.2 we call a function $y \in L^2(0,T;H^1(\Omega))$ with $div\, y = 0$ and satisfying the variational formulation in Problem 2.5 a weak solution of (2.1)–(2.3), (2.5).

In Gunzburger [44] and Bewley et al. [15] several objectives for flow control are discussed and corresponding cost functionals are presented. Certainly, a suitable cost functional that measures some properties of the state and control variables, depends on the state space $\mathbf{Y}$ and the control space $\mathbf{U}$. Furthermore, often it is necessary to obtain an optimal behaviour in some certain subregions of the flow domain. This can be modelled by choosing $\Omega_{obs} \subseteq \Omega$ appropriately. Among the most commonly used cost functionals in the literature on flow control are the ones presented below.

If a prescribed, desired flow field $y^d(x,t)$ is available, we can formulate a *tracking-type objective*

$$(\textit{Flow tracking}) \qquad J(y,u) = \tfrac{1}{2}\|y - y^d\|^2_{L^2(0,T;L^2(\Omega_{obs}))} + \tfrac{\gamma}{2}\|u\|^2_U \qquad (2.18)$$

such that (2.18) models the intention to drive the flow close to a desired state. The reduction of vortices in a flow can be modelled using

$$(\textit{Vorticity reduction}) \qquad J(y,u) = \tfrac{1}{2}\int_0^T \int_{\Omega_{obs}} |curl(y(x,t))|^2\, dx\, dt + \tfrac{\gamma}{2}\|u\|^2_U. \qquad (2.19)$$

The *vorticity* of a two-dimensional flow field, as a measure of its 'turbulence', is given by

$$curl(y(x,t)) = \frac{\partial}{\partial x_1} y_2(x,t) - \frac{\partial}{\partial x_2} y_1(x,t).$$

The above cost functionals are characterized by the fact that the pressure solution does not appear explicitly. Nevertheless, the pressure is a state variable that is coupled with the velocity field. Therefore, there are also objectives where the pressure occurs explicitly, if, e.g., stresses are included in the optimization objective, cf. Joslin et al. [59] where it is attempted to track desired stresses at parts of the boundary.

The second part in the cost functional, $\frac{\gamma}{2}\|u\|_U^2$, $\gamma > 0$, is introduced to achieve a regularization for the control problem and can be interpreted as balancing the control costs against the actual control objective.

Based on the variational formulation given in Problem 2.5 and the above cost functionals, we are now in a situation to formulate the optimal control problem that is to be solved.

Problem 2.6 (The Control Problem)
Let $J : \mathbf{Y} \times \mathbf{U} \to I\!R$ be given by (2.18) or (2.19). The control problem to be solved is given by:

$$\text{Minimize } J(y,u)$$

where $u \in \mathbf{U}$ and $y \in \mathbf{Y}$ is a weak solution of (2.1), (2.2), (2.3) and (2.5).

We close this section by giving an existence and uniqueness result of a weak solution for a flow configuration that fits into the framework (2.2)-(2.5) and that can be applied to the model problem to be presented in the next section. We also sketch its proof, since similar techniques appear in the reduced order modelling context in Chapter 3.

Theorem 2.7 (Fattorini/Sritharan [35], Th. 4)
Let $f = 0$, $\Gamma_c = \Gamma$, $b \in H^{3/2}(\Gamma)$ and $y_0 \in \mathbf{H}$ be given. Assume that

$$\int_\Gamma b(x) \cdot \mathbf{n}\, dx = 0 \tag{2.20}$$

holds. Furthermore, let $u \in H^1(0,T)$ be given.

Then, there exists a unique weak solution $y \in L^2(0,T;H^1(\Omega))$ of (2.1), (2.2), (2.3) that satisfies $y(x,t) = u(t)\, b(x)$, $x \in \Gamma$, $t \in (0,T)$, and $y \in C([0,T];\mathbf{H})$.

Sketch of proof (see Fattorini/Sritharan [35]): If the boundary data satisfies the compatibility condition (2.20), by employing Hopf's Lemma [40] we can construct a flow field that lifts the boundary forcing into the domain. For any arbitrary $\delta > 0$ there exists a flow field $w_\delta \in H^2(\Omega)$ that satisfies

$$div\, w_\delta = 0 \quad \text{in } \Omega, \qquad w_\delta(x) = b(x) \quad \text{for } x \in \Gamma \tag{2.21}$$

and

$$|\mathbf{c}(\phi, w_\delta, \phi)| \leq \delta \, \|\phi\|_{H^1}^2 \quad \text{for all } \phi \in \mathbf{V}.$$

Thus, we can set $y(x,t) = \tilde{y}(x,t) + u(t) w_\delta(x)$ and derive a Navier-Stokes-like system with homogeneous Dirichlet boundary conditions

$$\begin{aligned}
\tfrac{\partial}{\partial t}\tilde{y} - \nu\,\Delta\tilde{y} + (\tilde{y}\cdot\nabla)\,\tilde{y} + \nabla p &= \nu\,u\,\Delta w_\delta - u^2\,(w_\delta\cdot\nabla)\,w_\delta - \tfrac{\partial}{\partial t}\,u\,w_\delta \\
&\quad - u\,[(\tilde{y}\cdot\nabla)\,w_\delta + (w_\delta\cdot\nabla)\,\tilde{y}] \qquad \text{in } \Omega\times(0,T) \quad (2.22)\\
div\,\tilde{y} &= 0 \qquad \text{in } \Omega\times(0,T) \quad (2.23)\\
\tilde{y}(x,0) &= y_0(x) - u(0)\,w_\delta(x), \qquad x\in\Omega \quad (2.24)\\
\tilde{y} &= 0 \qquad \text{on } \Gamma\times(0,T). \quad (2.25)
\end{aligned}$$

Due to the homogeneous boundary conditions this yields a variational formulation similar to (2.9), (2.10), such that existence and uniqueness of a weak solution can be shown. □

Furthermore, since the cost functionals (2.18), (2.19) are bounded from below and weakly lower semicontinuous, also the existence of an optimal control follows, cf. e.g. Abergel/Temam [1].

Theorem 2.8 (Fattorini/Sritharan [35], Th. 1)
Under the assumptions of Theorem 2.7 there exists a solution $(y^\star, u^\star) \in \mathbf{Y}\times\mathbf{U}$ *of the optimal control problem given in Problem 2.6.*

2.4 A Model Problem: The Driven Cavity Flow

Since we employ the *driven cavity problem* as model problem for computations related to the proposed solution concepts throughout this work, we briefly introduce this model problem, here.

The driven cavity flow is a classical example for the analysis of simulation and control concepts for fluid flows and has also been used in, e.g., Peterson [88], Ito/Ravindran [57] or Allan [7] for reduced order modelling of fluid flows.

Figure 2.1 illustrates the domain $\Omega = (0,1)\times(0,1)$ and the boundary conditions for the standard cavity problem. On the top wall of the cavity,

$$\Gamma_{top} = \{(x_1, x_2) \in I\!R^2 : x_1 \in [\,0,1\,] \text{ and } x_2 = 1\},$$

there is a prescribed horizontal boundary velocity, i.e. $y = (1,0)^T,\ x\in\Gamma_{top},\ t\in(0,T)$. The body forces for this problem are set to zero: $f(x,t) = (0,0)^T,\ x\in\Omega,\ t\in(0,T)$. Consequently, driven by the wall velocity on Γ_{top}, there evolves a vortex inside the cavity which presents a steady-state solution for a sufficiently long time interval $(0,T)$.

For later use in the control context, we also define the bottom wall of the cavity as

$$\Gamma_{bot} = \{(x_1, x_2) \in I\!R^2 : x_1 \in [\,0,1\,] \text{ and } x_2 = 0\}.$$

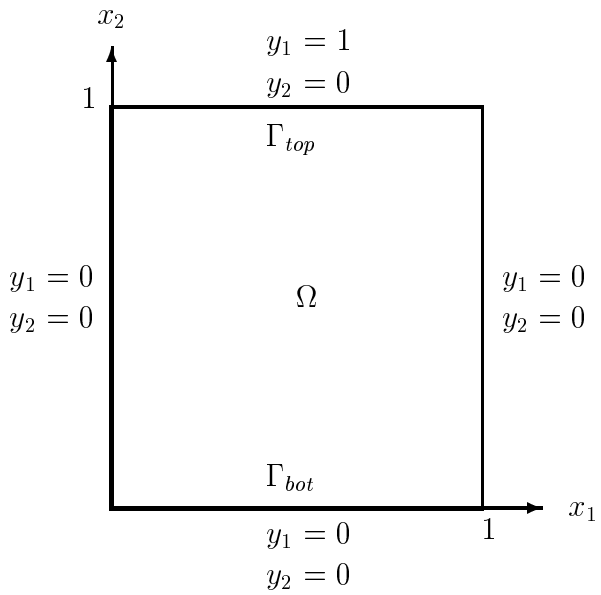

Figure 2.1: The Driven Cavity Problem

Chapter 3

Reduced Order Modelling based on POD

3.1 Introduction

Standard discretization schemes for the Navier-Stokes equations often lead to very large-scale, and in general, nonlinear systems of ordinary differential equations (ODEs). These are often too large to be used in an optimization context. Therefore, there is a demand to use a *reduced order model* that serves as a low-dimensional approximation model to the large-scale discretized state equations in order to get over this problem.

In recent years *Krylov approximation methods* based on the Lanczos or Arnoldi approach have been employed for reduced order modelling of partial differential equations in general and also for fluid flows, cf. Gallopoulos/Saad [39], Edwards et al. [31], Allan [7]. Another reduced order modelling approach that has been applied to flow problems is based on *reduced basis methods* as described, e.g., in Peterson [88], Ito/Ravindran [57] or Ito/Schroeter [58].

In this thesis, we focus on the *proper orthogonal decomposition* (POD) as reduced order modelling technique which is currently an active research field.

The proper orthogonal decomposition is also known as *Karhunen-Loève decomposition*, *method of empirical eigenfunctions* or *principal component analysis*. The POD approach consists of computing a small set of orthonormal functions, the *POD basis functions*, that can be efficiently used to describe the flow behaviour of the specific flow configuration under consideration. In order to achieve this, the POD method, or more precisely the *method of snapshots*, takes an ensemble of solutions of the time-dependent Navier-Stokes equations and computes characteristic patterns of this input set. As a result, the obtained POD basis functions represent these characteristic flow patterns. Here, the POD method resembles techniques used in other disciplines, e.g. in data compression, signal processing or pattern recognition. After those significant structures have been identified, the POD

basis functions are used in a Galerkin projection of the governing equations onto the subspace spanned by this basis. In this context, the POD basis can be interpreted as a basis consisting of global basis elements. Each POD basis element is related to the flow behavior on the whole domain. This is in contrast to the local character of, e.g., finite element basis functions.

The resulting low-dimensional model, the *POD based reduced order model*, permits an efficient approximate simulation of the state equations. Thus, this approach is of particular interest for applications in an optimization context provided that a once derived POD based reduced order model is a suitable model for a whole range of parameters.

The introduction of the proper orthogonal decomposition as a means to turbulence analysis is accredited to Lumley [76], cf. also the monograph [52]. Since then, it has been intensively used for computational studies of all different kind of problems where partial differential equations come into play: feedback control of the heat equation (Atwell/King [9, 10]), simulation and control of the Burgers equation (Park/Lee [85], Kunisch/Volkwein [73]), inverse problems (Banks et al. [12], Park/Lee [86]), simulation and control of the Navier-Stokes equations (Sirovich et al. [102], Rajaee et al. [92], Tang et al. [107], Ravindran [95], Hinze/Kunisch [51]), or chemical vapor deposition (Ly/Tran [78], Kepler et al. [66]). We note that this list is by far not complete, and that there is a vast amount of applications that have been investigated in other engineering areas.

In this chapter, we present the basic ideas of the proper orthogonal decomposition and review its close relationship to the singular value decomposition that serves as a major tool for data analysis, cf. Fukunaga [37], Moonen/De Moor [79], Hansen [46]. We give a review of the essential properties of the POD basis functions that make this approach very attractive for reduced order modelling of time-dependent partial differential equations. Having flow control applications in mind, we give a detailed derivation of the POD based reduced order models for the Navier-Stokes equations subject to different boundary conditions. Numerical results are given for the cavity flow model problem in order to illustrate characteristic properties of the proper orthogonal decomposition.

3.2 The Proper Orthogonal Decomposition

In this section, we introduce the proper orthogonal decomposition as presented in Berkooz et al. [14] and Sirovich [101] for the identification of *coherent structures* in turbulent flows, and as it can be found in most of the literature on POD based reduced order modelling.

Coherent structures are organized characteristic spatial flow patterns that repeatedly appear and undergo a typical temporal life cycle [14]. In this context the proper orthogonal decomposition serves to study the flow behaviour itself and to analyze pattern formation.

We assume that an input ensemble $\mathcal{Y} = \{y_1, \ldots, y_P\}$ of d-dimensional flow fields $y_i(x)$, $x \in \Omega$, is given, where $d = 2, 3$. In principal it is not important how this input ensemble

will be derived for a specific application, say, by experimental studies or *direct numerical simulation* (DNS). However, the POD technique provides a tool to analyze characteristic features of this input ensemble by constructing an orthonormal basis for $span\,\mathcal{Y}$.

Based on the input ensemble $\mathcal{Y}$, in [14] a single flow field that contains most of the energy of the input flow fields in an average sense is sought by maximizing the expression

$$\max_{\|\psi\|=1} \frac{1}{P}\sum_{i=1}^{P} |(y_i,\psi)|^2 \tag{3.1}$$

where $(v,w)=(v,w)_{L^2}$ denotes the L^2-inner product on the flow domain Ω with corresponding norm $\|v\|=\|v\|_{L^2}$. The solution to (3.1) delivers the first POD basis element.

A reformulation of the objective in (3.1) leads to, cf. Ly/Tran [78],

$$\begin{aligned}\frac{1}{P}\sum_{i=1}^{P} |(y_i,\psi)|^2 &= \frac{1}{P}\sum_{i=1}^{P}\left(\int_\Omega y_i(x)^T\psi(x)\,dx\right)\left(\int_\Omega y_i(z)^T\psi(z)\,dz\right)\\ &= \int_\Omega \psi(x)^T\left[\int_\Omega\left(\frac{1}{P}\sum_{i=1}^{P} y_i(x)y_i(z)^T\right)\psi(z)\;dz\right]\;dx\\ &= (\psi,R\psi)\end{aligned}$$

where

$$R\psi=\int_\Omega\left(\frac{1}{P}\sum_{i=1}^{P} y_i(x)y_i(z)^T\right)\psi(z)\;dz \tag{3.2}$$

defines a linear, self-adjoint, nonnegative operator, often termed *autocorrelation operator*. We state the following result which can be found in [14], [101].

Lemma 3.1
The solution to problem (3.1) is given by the (normed) eigenfunction of R that corresponds to the largest eigenvalue of R.

Furthermore, we note that several POD basis elements can be computed simultaneously by solving

$$\max_{\psi_1,\ldots,\psi_M}\sum_{j=1}^{M}\frac{1}{P}\sum_{i=1}^{P}|(y_i,\psi_j)|^2 \tag{3.3}$$

subject to the requirement that $\psi_1,\ldots,\psi_M$ are mutually orthonormal. Here, the solution to (3.3), where M POD basis elements are sought, is given by the (normed) eigenfunctions of R that correspond to the M largest eigenvalues of R.

Definition 3.2 (POD Basis)
Let $\mathcal{Y}$ be given. We call a solution $\{\psi_1,\ldots,\psi_M\}$ to (3.3) a POD basis of order M and set

$$\mathcal{Y}^{POD}=\{\psi_1,\ldots,\psi_M\}. \tag{3.4}$$

Furthermore, we call $span\,\mathcal{Y}^{POD}$ the corresponding *POD subspace* of $span\,\mathcal{Y}$.

3.3 Relationships between the Proper Orthogonal Decomposition and the Singular Value Decomposition

In this thesis, the input ensemble will always represent *time snapshots* of the governing equations obtained by DNS using a finite element method. In case of POD for the Navier-Stokes equations, we consider flow fields $y_i^N(x) = y^N(x,t_i),\ x \in \Omega$, at time instants $t_i \in (0,T),\ i = 1,\ldots,P$, where the superscript N denotes a high-order finite element discretization.

In what follows we discuss the close relationship between the proper orthogonal decomposition and the singular value decomposition. For similar treatments we refer to Kunisch/Volkwein [72, 116] and Atwell/King [10].

Since we are working with snapshots that result from DNS, by adapting the underlying spatial discretization of the domain Ω of the finite element solver we can reformulate problem (3.3) in a finite dimensional setting.

From now on, we assume that the input ensemble, $\mathcal{Y}$, consists of linear independent finite element (FE) snapshots of the Navier-Stokes equations for $d=2$,

$$y_i^N(x) = \begin{pmatrix} \sum_{j=1}^N \mathbf{y}_j^N(t_i)\,\varphi_j(x) \\ \sum_{j=1}^N \mathbf{y}_{N+j}^N(t_i)\,\varphi_j(x) \end{pmatrix}, \tag{3.5}$$

$x \in \Omega,\ t_i \in (0,T)$, where $\{\varphi_j\}_{j=1}^N$ denote finite element basis functions.

Definition 3.3 (Snapshot Data Matrix)
Let the input ensemble $\mathcal{Y} = \{y_i^N\}_{i=1}^P$ be given. We call

$$Y = \begin{pmatrix} \mathbf{y}_1^N(t_1) & \ldots & \mathbf{y}_1^N(t_P) \\ \vdots & & \vdots \\ \mathbf{y}_N^N(t_1) & \ldots & \mathbf{y}_N^N(t_P) \\ \mathbf{y}_{N+1}^N(t_1) & \ldots & \mathbf{y}_{N+1}^N(t_P) \\ \vdots & & \vdots \\ \mathbf{y}_{2N}^N(t_1) & \ldots & \mathbf{y}_{2N}^N(t_P) \end{pmatrix} \in \mathbb{R}^{2N\times P} \tag{3.6}$$

the *snapshot data matrix* of $\mathcal{Y}$.

We note that each column $Y_{:,i} \in \mathbb{R}^{2N}$ of the snapshot data matrix represents a single snapshot y_i^N of the input ensemble $\mathcal{Y}$, implying that the snapshot data matrix has full column rank due to the linear independence assumption on the snapshots.

Since we aim at working with a discretized version of (3.3), we have to specify an appropriate inner product and norm for the corresponding problem statement. In (3.1) we employed the L^2-inner product. Concerning the choice of other norms for the POD basis computation, we refer to Kirby [68], Wu/Shi [117] or Iollo et al. [55].

For the computation of POD basis functions based on FE snapshots of a two-dimensional flow we define the inner product for given

$$\psi(x) = \begin{pmatrix} \sum_{j=1}^{N} \boldsymbol{\psi}_j \, \varphi_j(x) \\ \sum_{j=1}^{N} \boldsymbol{\psi}_{N+j} \, \varphi_j(x) \end{pmatrix}, \qquad \phi(x) = \begin{pmatrix} \sum_{j=1}^{N} \boldsymbol{\phi}_j \, \varphi_j(x) \\ \sum_{j=1}^{N} \boldsymbol{\phi}_{N+j} \, \varphi_j(x) \end{pmatrix},$$

$x \in \Omega$, by

$$(\psi, \phi)_{\mathcal{M}} = \boldsymbol{\psi}^T \mathcal{M} \, \boldsymbol{\phi} \tag{3.7}$$

where $\mathcal{M} \in \mathbb{R}^{2N \times 2N}$ is the FE mass matrix and $\boldsymbol{\psi}, \boldsymbol{\phi} \in \mathbb{R}^{2N}$ are FE coefficient vectors. The corresponding norm is given by $\|\psi\|_{\mathcal{M}} = (\psi, \psi)_{\mathcal{M}}^{1/2}$. Employing a Cholesky factorization, $\mathcal{M} = \mathcal{M}^{1/2}(\mathcal{M}^{1/2})^T$, of the (positive definite) mass matrix $\mathcal{M}$, it is also possible to transform the $\mathcal{M}$-inner product (3.7) to the standard Euclidean inner product such that

$$\|\psi\|_{\mathcal{M}} = \|(\mathcal{M}^{1/2})^T \boldsymbol{\psi}\|_2 \tag{3.8}$$

holds. From now on, we consider the problem of computing a POD basis of order M as given by Problem 3.4 which is equivalent to a problem statement of the form (3.3).

Problem 3.4 (The POD Problem)
Let $\mathcal{Y}$ be given. Find orthonormal functions $\{\psi_j\}_{j=1}^{M}$ solving

$$\min \sum_{i=1}^{P} \|y_i^N - \sum_{j=1}^{M} (y_i^N, \psi_j)_{\mathcal{M}} \, \psi_j\|_{\mathcal{M}}^2 \tag{3.9}$$

Based on (3.7), (3.8), we can reformulate Problem 3.4 in a matrix approximation context. For this purpose, let (similar to (3.6)) $\Psi \in \mathbb{R}^{2N \times M}$ denote a matrix collecting the FE coefficients of the still unknown POD basis functions $\{\psi_j\}_{j=1}^{M}$.

Lemma 3.5
Let $\mathcal{Y}$ with snapshot data matrix Y be given and define

$$A = (\mathcal{M}^{1/2})^T Y \ \in \mathbb{R}^{2N \times P}. \tag{3.10}$$

Then, Problem 3.4 is equivalent to solving

$$\min_{Z \in \mathbb{R}^{2N \times M}} \|A - ZZ^T A\|_F^2 \quad s.t. \quad Z^T Z = I_M \tag{3.11}$$

with $Z = (\mathcal{M}^{1/2})^T \Psi$, where $\|\cdot\|_F$ denotes the Frobenius norm and $I_M \in \mathbb{R}^{M \times M}$ is the identity matrix.

Proof: Problem (3.11) follows by applying the transformation (3.8) to (3.9) and using the relation $\|A\|_F^2 = \sum_{i=1}^{P} \|A_{:,i}\|_2^2$, where $A_{:,i}$ denotes the i-th column of A. □

Similar to (3.9), (3.11) indicates that we are looking for a M-dimensional subspace with orthonormal basis Z such that the projection of A onto the range of Z given by

$$X = ZZ^T A \tag{3.12}$$

is a best approximation to A compared to all subspaces of dimension M. Hence, Problem (3.11) is a special problem of the more general form

$$\min_{X \in \mathbb{R}^{2N \times P}} \|A - X\|_F^2 \quad s.t. \quad rank(X) = M. \tag{3.13}$$

The computation of a matrix with given rank that is nearest to a given data matrix is a classical problem whose solution is presented in Theorem 3.7. For this purpose and in order to fix notation, we briefly review the singular value decomposition of a matrix (cf. e.g. Stewart/Sun [106]).

Definition 3.6 (Singular Value Decomposition)
Let $A \in \mathbb{R}^{m \times n}$, $m \geq n$, with $rank(A) = r \leq n$ be given.

Then, there exist orthonormal matrices $U \in \mathbb{R}^{m \times m}$, $V \in \mathbb{R}^{n \times n}$ such that

$$U^T AV = \Sigma, \qquad \Sigma = \begin{pmatrix} \Sigma_r & 0 \\ 0 & 0 \end{pmatrix} \tag{3.14}$$

where $\Sigma_r = diag(\sigma_1, \ldots, \sigma_r) \in \mathbb{R}^{r \times r}$ with $\sigma_1 \geq \ldots \geq \sigma_r > 0$. The decomposition

$$A = U\Sigma V^T \tag{3.15}$$

is called the *singular value decomposition* (SVD) of A and the diagonal entries of Σ, $(\Sigma)_{ii}$, are called *singular values*. The columns of U and V are corresponding *left* and *right singular vectors* of A.

Theorem 3.7 (Eckart and Young, cf. [49])
Let $A \in \mathbb{R}^{m \times n}$, $m \geq n$, have SVD $A = U\Sigma V^T$ according to (3.15) and let $k < r = rank(A)$.

Then,

$$\min_{rank(X)=k} \|A - X\|_F = \|A - A_k\|_F = \sqrt{\sum_{j=k+1}^{r} \sigma_j^2} \tag{3.16}$$

where

$$A_k = U \begin{pmatrix} \Sigma_k & 0 \\ 0 & 0 \end{pmatrix} V^T.$$

Theorem 3.7 shows that the solution to the matrix approximation problem (3.13) is given by a *truncated singular value decomposition* (TSVD). Thus, the computation of the POD basis functions can be carried out using the relationship between the approximation problem (3.13) and the SVD of A.

Theorem 3.8 (Computation of the POD Basis)
Let $\mathcal{Y} = \{y_i^N\}_{i=1}^P$ denote an input ensemble of FE snapshots. Let the corresponding data matrix $A \in \mathbb{R}^{2N \times P}$ be given by (3.6) and (3.10) with SVD $A = U\Sigma V^T$.

Then, a POD subspace for span $\mathcal{Y}$ is given by the POD basis $\mathcal{Y}^{POD} = \{\psi_i\}_{i=1}^M$ of order $M \leq P$ defined by

$$\psi_i(x) = \begin{pmatrix} \sum_{j=1}^N \Psi_{ji}\, \varphi_j(x) \\ \sum_{j=1}^N \Psi_{N+j,i}\, \varphi_j(x) \end{pmatrix}, \quad i = 1, \ldots, M, \tag{3.17}$$

$x \in \Omega$, where the matrix $\Psi \in \mathbb{R}^{2N \times M}$, collecting the FE coefficients of the POD basis functions, solves

$$(\mathcal{M}^{1/2})^T \Psi = U_M. \tag{3.18}$$

Here, $U_M \in \mathbb{R}^{2N \times M}$ denotes the matrix consisting of the first M left singular vectors of A.

Proof: According to Theorem 3.7 the solution to problem (3.13) is given by a TSVD of A of length M

$$A_M = U \begin{pmatrix} \Sigma_M & 0 \\ 0 & 0 \end{pmatrix} V^T = U_M \Sigma_M V_M^T \tag{3.19}$$

where U_M, V_M correspond to the first M columns of U, V, respectively. Comparing (3.19) and the special form of X for the POD basis computation given by (3.12), and due to the transformation (3.8), the FE coefficients of the POD basis functions can be computed by solving the linear system (3.18). □

Remark 3.9
At this point, we remark that the POD basis functions can also be computed by solving eigenproblems with, e.g., matrices AA^T or A^TA, instead of computing the SVD of A. In this case, AA^T represents a finite-dimensional version of the autocorrelation operator in (3.2). A more detailed discussion of this subject in connection with Lanczos methods for POD basis computations is given in Chapter 4.

3.4 Properties of the POD Basis Functions

3.4.1 Orthonormality

We note that the POD basis functions are orthonormal with respect to the $\mathcal{M}$–inner product

$$(\psi_i, \psi_j)_{\mathcal{M}} = \Psi_{:,i}^T\, \mathcal{M}\, \Psi_{:,j} = \delta_{ij}$$

where $\Psi_{:,i}$ denotes the i-th column of the coefficient matrix Ψ. Since $A = (\mathcal{M}^{1/2})^T Y$ and Ψ is computed via (3.18) one might think that it is advantageous to compute Ψ directly via the SVD of Y. But this approach does not account for the appropriate orthonormality condition with respect to the $\mathcal{M}$–inner product.

3.4.2 Incompressibility and Boundary Conditions

According to (3.5), (3.15) and (3.17) we have

$$\begin{aligned}
\psi_i(x) &= \begin{pmatrix} \sum_{j=1}^{N} \Psi_{ji}\, \varphi_j(x) \\ \sum_{j=1}^{N} \Psi_{N+j,i}\, \varphi_j(x) \end{pmatrix} \\
&= \begin{pmatrix} \sum_{j=1}^{N} (\frac{1}{\sigma_i} \sum_{k=1}^{P} Y_{jk}\, V_{ki})\, \varphi_j(x) \\ \sum_{j=1}^{N} (\frac{1}{\sigma_i} \sum_{k=1}^{P} Y_{N+j,k}\, V_{ki})\, \varphi_j(x) \end{pmatrix} \\
&= \frac{1}{\sigma_i} \sum_{k=1}^{P} V_{ki}\, y_k^N(x), \qquad i = 1, \ldots, M
\end{aligned} \tag{3.20}$$

for all $x \in \Omega$, i.e. the POD basis functions can be represented as linear combinations of solutions of the Navier–Stokes equations. Therefore, certain properties of the snapshots pass to the POD basis functions. The most important property in the context of POD for the Navier–Stokes equations is incompressibility. Since the snapshots are divergence-free, we obtain divergence-free POD basis functions, i.e.

$$div\, \psi_i = \frac{1}{\sigma_i} \sum_{k=1}^{P} V_{ki}\, div\, y_k^N = 0, \qquad i = 1, \ldots, M. \tag{3.21}$$

Due to (3.20), we also obtain POD basis functions satisfying homogeneous boundary conditions,

$$\psi_i(x) = \frac{1}{\sigma_i} \sum_{k=1}^{P} V_{ki}\, y_k^N(x) = 0, \qquad x \in \Gamma = \partial\Omega, \quad i = 1, \ldots, M, \tag{3.22}$$

if the snapshots themselves satisfy homogeneous Dirichlet boundary conditions.

3.4.3 Optimality

As a result of Theorem 3.7 the POD basis inherits an interesting optimality property.

Corollary 3.10 (Optimality of POD basis)
Let $\mathcal{Y} = \{y_i^N\}_{i=1}^P$ denote a given input ensemble and let the corresponding data matrix $A \in \mathbb{R}^{2N \times P}$ be given by (3.6) and (3.10) with SVD $A = U\Sigma V^T$. Let $\{\psi_i\}_{i=1}^P$ denote the maximal POD basis for span $\mathcal{Y}$.

Then,

$$\sum_{i=1}^{P} \|y_i^N - \sum_{j=1}^{M} (y_i^N, \psi_j)_{\mathcal{M}}\, \psi_j\|_{\mathcal{M}}^2 \leq \sum_{i=1}^{P} \|y_i^N - \sum_{j=1}^{M} (y_i^N, \phi_j)_{\mathcal{M}}\, \phi_j\|_{\mathcal{M}}^2 \tag{3.23}$$

for any other orthonormal basis $\{\phi_i\}_{i=1}^P$ of span $\mathcal{Y}$ and for all $M \leq P$.

Moreover, the truncation error, $\varepsilon(M)$, of using M instead of P POD basis functions in representing $\mathcal{Y}$ is given by

$$\varepsilon(M) = \sum_{i=1}^{P} \|y_i^N - \sum_{j=1}^{M} (y_i^N, \psi_j)_{\mathcal{M}} \, \psi_j \|_{\mathcal{M}}^2 = \sum_{j=M+1}^{P} \sigma_j^2 \tag{3.24}$$

where σ_j, $j = M+1, \ldots, P$, denote the smallest $(P-M)$ singular values of A.

Proof: Since (3.9) can be equivalently formulated as matrix approximation problem (3.11), the optimality property (3.23) as well as the representation of the error $\varepsilon(M)$ follow from Theorem 3.7. □

The quantity $\varepsilon(M)$ measures the accumulated squared error in representing the input snapshots, due to neglecting POD basis elements that correspond to small singular values. Including an averaging operation as in (3.3),

$$\bar{\varepsilon}(M) = \tfrac{1}{P} \sum_{i=1}^{P} \|y_i^N - \sum_{j=1}^{M} (y_i^N, \psi_j)_{\mathcal{M}} \, \psi_j \|_{\mathcal{M}}^2, \tag{3.25}$$

we obtain some kind of measure for the average error of each POD approximation to a snapshot.

3.4.4 On the Choice of the POD Subspace Dimension

Following Corollary 3.10, by the choice of M a control of the magnitude of the truncation error (3.24) is possible. Nevertheless, the choice of M is an important and critical task, since it determines the interrelation between accuracy and dimension of the POD based reduced order models that are going to be presented in Section 3.5. A commonly used criterion for choosing M based on heuristic considerations is the *energy criterion*. For a predefined 'percentage of energy' $\bar{\epsilon} > 0$, the POD subspace dimension, M, is chosen such that

$$\frac{\sum_{j=1}^{M} \sigma_j^2}{\sum_{j=1}^{P} \sigma_j^2} \geq \bar{\epsilon} \tag{3.26}$$

holds, see [101, 14, 77, 9, 60].

An additional criterion can be derived in the SVD context, see Hansen [46]. For this purpose, let us assume that a matrix $A \in \mathbb{R}^{m \times n}$, $m \geq n$, with $rank(A) = r$ and SVD $A = U \Sigma V^T$ is given. Then, the range of A is spanned by the first r left singular vectors of A, i.e. $\mathcal{R}(A) = span\, U_r$. For a given small threshold $\delta > 0$, we say that A has numerical δ-rank, r_δ, if there is a well-determined gap between the singular values σ_{r_δ} and $\sigma_{r_\delta+1}$ and if δ separates these singular values: $\sigma_{r_\delta} > \delta > \sigma_{r_\delta+1}$.

In this case, the singular values $\sigma_j > 0$ for $j \geq r_\delta + 1$ can be neglected from a practical point of view, since $span\, U_{r_\delta} \approx span\, U_r$.

These considerations suggest that also a gap criterion concerning the singular values should be used for the choice of M in the POD basis computations. Furthermore, a POD subspace related to clearly separated singular values is well-conditioned and insensitive to perturbations [42].

3.5 POD based Reduced Order Models

Once the POD basis functions have been computed, the reduced order model can be derived by projecting the underlying partial differential equation onto the corresponding POD subspace.

3.5.1 Homogeneous Boundary Conditions

For what follows, we consider the two-dimensional NSE subject to homogeneous Dirichlet boundary conditions. In this case the POD basis functions inherit these boundary conditions according to (3.22). Since they are also divergence-free the POD approach provides a very useful discretization for the variational formulation (I) of the Navier–Stokes equations given in Section 2.2.2.

In order to establish a POD based reduced order model for (2.1)–(2.3) and (2.6), the velocity field has to be expanded in terms of the POD basis

$$y^M(x,t) = \sum_{j=1}^{M} \mathbf{y}_j^M(t)\,\psi_j(x) \tag{3.27}$$

where the superscript M denotes the POD approximation. Then, a Galerkin projection onto the POD subspace is performed, such that we obtain a nonlinear ODE system that serves as an approximation model to the finite element model of the NSE and that can be used to determine the expansion coefficients in (3.27).

Lemma 3.11 (POD based Reduced Order Model)
Let $\mathcal{Y}$ denote an ensemble of FE snapshots of (2.1)–(2.3), (2.6) and let $\{\psi_j\}_{j=1}^M$ denote corresponding POD basis functions.

Then, a POD based reduced order model for (2.1)–(2.3), (2.6) is given by

$$\dot{\mathbf{y}}^M(t) + \nu\mathcal{A}\,\mathbf{y}^M(t) + \mathcal{N}(\mathbf{y}^M,\mathbf{y}^M) = \mathcal{F}(t) \qquad \text{for all } t\in(0,T) \tag{3.28}$$

$$\mathbf{y}^M(0) = \mathbf{y}_0^M \tag{3.29}$$

where $\mathcal{A}\in\mathbb{R}^{M\times M}$ with $(\mathcal{A})_{ij} = (\nabla\psi_i,\nabla\psi_j)$, $\mathcal{F}(t)\in\mathbb{R}^M$ with $(\mathcal{F}(t))_j = (f(t),\psi_j)$, and $(\mathcal{N}(\mathbf{y}^M,\mathbf{y}^M))_j = \mathbf{y}^M(t)^T\,\mathcal{Q}_j\,\mathbf{y}^M(t)$ with $(\mathcal{Q}_j)_{ik} = ((\psi_i\cdot\nabla)\psi_k\,,\,\psi_j)$ for all $i,j,k=1,\ldots,M$. Furthermore, $\mathbf{y}_0^M\in\mathbb{R}^M$ is given by $(\mathbf{y}_0^M)_j = (y^N(0),\psi_j)$ for $j=1,\ldots,M$.

Proof: According to (2.9), (2.10) a variational formulation for (2.1)–(2.3), (2.6) is given by

$$\begin{aligned} \tfrac{\partial}{\partial t}(y,\phi)_{L^2} + \nu\, \mathbf{a}(y,\phi) + \mathbf{c}(y,y,\phi) &= (f,\phi)_{L^2} \qquad \text{for all } \phi \in \mathbf{V} \\ y(0) &= y_0. \end{aligned}$$

Since we assumed that the snapshots satisfy homogeneous Dirichlet boundary conditions and since the POD basis elements are divergence-free, we have $span\,\{\psi_1,\dots,\psi_M\} \subset \mathbf{V}$. Using (3.27), the Galerkin projection onto the POD subspace yields a M-dimensional ODE system of the form (3.28). □

Remark 3.12

For an implementation of the POD based reduced order model (3.28), (3.29), the FE representation of the POD basis functions (3.17) can be exploited. For this purpose, let $\Psi_{:,i}$ denote the i-th column of Ψ.

Then, the POD stiffness matrix $\mathcal{A}$ corresponds to the projected finite element stiffness matrix (which we denote by $\mathcal{S}$) such that $(\mathcal{A}_{ij}) = \Psi_{:,i}^T\, \mathcal{S}\, \Psi_{:,j},\ i,j=1,\dots,M$. The nonlinear part of (3.28) can be reformulated as $(\mathcal{Q}_j)_{ik} = \Psi_{:,i}^T\, \mathcal{K}\,(\Psi_{:,j})\Psi_{:,k}$ for $i,j,k=1,\dots,M$, where $\mathcal{K}(\Psi_{:,j})$ depends linearly on $\Psi_{:,j}$ and has the block structure

$$\mathcal{K}(\Psi_{:,j}) = \begin{pmatrix} \mathcal{K}_{11}(\Psi_{:,j}) & \mathcal{K}_{12}(\Psi_{:,j}) \\ \mathcal{K}_{21}(\Psi_{:,j}) & \mathcal{K}_{22}(\Psi_{:,j}) \end{pmatrix}$$

with $(\mathcal{K}_{11}(\Psi_{:,j}))_{kl} = \sum_{i=1}^N \Psi_{ij}\,(\varphi_k \nabla_{x_1}\varphi_l, \varphi_i)$, $(\mathcal{K}_{12}(\Psi_{:,j}))_{kl} = \sum_{i=1}^N \Psi_{ij}\,(\varphi_k \nabla_{x_2}\varphi_l, \varphi_i)$, $(\mathcal{K}_{21}(\Psi_{:,j}))_{kl} = \sum_{i=1}^N \Psi_{N+i,j}\,(\varphi_k \nabla_{x_1}\varphi_l, \varphi_i)$, $(\mathcal{K}_{22}(\Psi_{:,j}))_{kl} = \sum_{i=1}^N \Psi_{N+i,j}\,(\varphi_k \nabla_{x_2}\varphi_l, \varphi_i)$ for $k,\, l=1,\dots,N$. Analogously, a reduced order representation of the right hand side and the initial condition can be derived.

3.5.2 Nonhomogeneous Boundary Conditions

In Subsection 3.5.1 we considered homogeneous boundary conditions such that the POD basis elements also satisfied these boundary conditions. Next, we consider the NSE with nonhomogeneous Dirichlet boundary conditions according to (2.5).

Compared to the homogeneous boundary condition case, the following problem arises: If the input ensemble consists of snapshots y_i^N that satisfy the required (nonhomogeneous) boundary conditions, then the corresponding POD basis elements are no longer suitable for a discretization of a variational formulation in the sense of Problem 2.5. Due to (3.20) they do not satisfy homogeneous boundary conditions. The solution to this problem is in the spirit of (2.23)–(2.25), i.e. by transforming the actual problem to a problem with homogeneous boundary conditions.

In case $g(x,t)$ does not depend on t, i.e., $g(x,t) = c(x)$ for all $x \in \Gamma,\ t \in (0,T)$, a possibility to overcome this difficulty is given by computing POD basis elements for the fluctuations around the mean flow field, see Sirovich [101].

Given P snapshots, the flow mean value $\bar{y}^N = \frac{1}{P}\sum_{i=1}^{P} y_i^N$ is computed, first. The POD basis functions are computed using the modified input ensemble $\{y_1^N - \bar{y}^N, \ldots, y_P^N - \bar{y}^N\}$, then. Due to its construction, the mean flow field $\bar{y}^N$ is divergence-free and satisfies the prescribed nonhomogeneous Dirichlet boundary conditions. Furthermore, each modified snapshot, $y_i^N - \bar{y}^N$, is also divergence-free, but satisfies homogeneous Dirichlet boundary conditions.

Therefore, in order to derive a reduced order model for this kind of problem we can make use of the expansion

$$y^M(x,t) = \bar{y}^N(x) + \sum_{j=1}^{M} \mathbf{y}_j^M(t)\psi_j(x). \tag{3.30}$$

The resulting reduced order model differs slightly from (3.28), (3.29). We obtain

$$\begin{aligned} \dot{\mathbf{y}}^M(t) + [\nu\mathcal{A} + \bar{\mathcal{A}}]\,\mathbf{y}^M(t) + \mathcal{N}(\mathbf{y}^M, \mathbf{y}^M) &= \mathcal{F}(t) + \bar{\mathcal{F}} \quad \text{for all } t \in (0,T) \\ \mathbf{y}^M(0) &= \mathbf{y}_0^M \end{aligned}$$

where $\bar{\mathcal{A}} \in I\!R^{M\times M}$, $\bar{\mathcal{F}} \in I\!R^M$ are given by

$$\begin{aligned} (\bar{\mathcal{A}})_{ji} &= ((\psi_i \cdot \nabla)\bar{y}^N, \psi_j) + ((\bar{y}^N \cdot \nabla)\psi_i, \psi_j), \\ (\bar{\mathcal{F}})_j &= -\left[\nu\,(\nabla\bar{y}^N, \nabla\psi_j) + ((\bar{y}^N \cdot \nabla)\bar{y}^N, \psi_j)\right] \end{aligned}$$

for $i, j = 1, \ldots, M$.

Even more interesting for reduced order modelling of boundary control problems is the case where the Dirichlet boundary condition on $\Gamma_c \subset \Gamma$ is a time-dependent one. For this purpose, we return to boundary conditions of the type

$$g(x,t) = \begin{cases} u(t)b(x), & x \in \Gamma_c,\ t \in (0,T) \\ c(x), & x \in \Gamma \setminus \Gamma_c,\ t \in (0,T) \end{cases} \tag{3.31}$$

according to (2.5), where $u \in H^1(0,T)$ is assumed, see Section 2.3.

Let $\{y_i^N\}_{i=1}^P$ denote a snapshot set that corresponds to the boundary conditions in (3.31). In order to match these boundary conditions with an appropriate reduced order model, the velocity field has to be expanded in

$$y^M(x,t) = \bar{y}^N(x) + u(t)\,y_b^N(x) + \sum_{j=1}^{M} \mathbf{y}_j^M(t)\psi_j(x). \tag{3.32}$$

To achieve this, we choose a reference flow field $y_b^N(x),\ x \in \Omega$, describing how the control action $u(t)\,b(x),\ x \in \Gamma_c, t \in (0,T)$, influences the fluid flow and satisfying the

boundary conditions

$$u(t)y_b^N(x) = \begin{cases} u(t)\, b^N(x), & x \in \Gamma_c,\ t \in (0,T) \\ 0\,, & x \in \Gamma \setminus \Gamma_c,\ t \in (0,T) \end{cases} \tag{3.33}$$

where $b^N(x)$ denotes the finite element approximation to $b(x),\ x \in \Omega$.

Afterwards, similar to the technique presented in Tang et al. [107] or Ravindran [95], a mean flow field $\bar{y}^N$ of the modified snapshot set $\{y_1^N - u(t_1)y_b^N, \ldots, y_P^N - u(t_P)y_b^N\}$ is computed. Then, the POD basis computations are performed for the input ensemble $\{y_1^N - u(t_1)y_b^N - \bar{y}^N, \ldots, y_P^N - u(t_P)y_b^N - \bar{y}^N\}$.

Since $(y_i^N - u(t_i)y_b^N)|_{\Gamma_c} = 0$ and $\bar{y}^N$ matches all other nonhomogeneous boundary conditions, we again get POD basis elements that satisfy homogeneous boundary conditions on the whole boundary.

Using this approach, the following POD based reduced order model for the Navier–Stokes equations with boundary control according to (2.1)–(2.3), (2.5) is obtained, which we call the *POD based control model*:

$$\begin{aligned} \dot{\mathbf{y}}^M(t) + [\nu\mathcal{A} + \bar{\mathcal{A}} + u(t)\mathcal{A}_b]\,\mathbf{y}^M(t) + \mathcal{N}(\mathbf{y}^M, \mathbf{y}^M) &= \mathcal{F}(t) + \bar{\mathcal{F}} + u(t)\,\bar{\mathcal{F}}_b + u(t)^2\,\mathcal{F}_{db} + \dot{u}(t)\,\mathcal{F}_b \\ \mathbf{y}^M(0) &= \mathbf{y}_0^M \end{aligned}$$

for all $t \in (0,T)$. The novel terms $\mathcal{A}_b \in \mathbb{R}^{M\times M}$ and $\bar{\mathcal{F}}_b$, $\mathcal{F}_{db}$, $\mathcal{F}_b \in \mathbb{R}^M$ in the POD based control model are defined by

$$\begin{aligned} (\mathcal{A}_b)_{ji} &= ((y_b^N \cdot \nabla)\,\psi_i, \psi_j) + ((\psi_i \cdot \nabla)\,y_b^N, \psi_j), \\ (\bar{\mathcal{F}}_b)_j &= -[\nu\,(\nabla y_b^N, \nabla\psi_j) + ((\bar{y}^N \cdot \nabla)\,y_b^N, \psi_j) + ((y_b^N \cdot \nabla)\,\bar{y}^N, \psi_j)], \\ (\mathcal{F}_{db})_j &= -((y_b^N \cdot \nabla)\,y_b^N, \psi_j), \\ (\mathcal{F}_b)_j &= -(y_b^N, \psi_j) \end{aligned}$$

for $i, j = 1, \ldots, M$.

Remark 3.13

We note that the term $\dot{u}(t)\,\mathcal{F}_b$ in the right hand side of the POD based control model is difficult to treat during the computation of a reduced order solution.

Setting $\tilde{\mathbf{y}}^M(t) = \mathbf{y}^M(t) - u(t)\mathcal{F}_b$, allows to reformulate the POD based control model such that an ODE system in the unknowns $\tilde{\mathbf{y}}^M(t),\ t \in (0,T)$, is obtained. As a consequence, this modified reduced order control model does not depend on the time derivative of u explicitly.

A similar transformation is applied in Ravindran [95]. Defining $\hat{\mathbf{y}}^M(t) = (\mathbf{y}^M(t), u(t))$, reformulating the reduced order model accordingly and using the time derivative $\dot{u}$ rather than u itself as the control variable, also eliminates $\dot{u}$ in the reduced order model, see also Fattorini/Sritharan [35].

3.5.3 Computing a Reduced Order Solution

Summing up, the procedure for deriving a POD based reduced order model and for computing a POD based reduced order solution of the Navier-Stokes equations can be formulated as follows.

The POD Procedure

1. Compute snapshots $\{y_i^N\}_{i=1}^P$ of the Navier-Stokes equations (2.1)–(2.3) with given Dirichlet boundary conditions.

2. If necessary, modify the snapshot set in order to obtain an input ensemble $\mathcal{Y}$ in which each flow field satisfies homogeneous boundary conditions, i.e.

   (a) in case of homogeneous boundary conditions: Set $\mathcal{Y} = \{y_i^N\}_{i=1}^P$,

   (b) in case of time-independent (nonhomogeneous) boundary conditions: Compute the mean flow field $\bar{y}^N$ of $\{y_i^N\}_{i=1}^P$ and $\mathcal{Y} = \{y_i^N - \bar{y}^N\}_{i=1}^P$,

   (c) in case of time-dependent boundary conditions: Choose a reference flow field y_b^N and compute $\tilde{\mathcal{Y}} = \{y_i^N - u(t_i)y_b^N\}_{i=1}^P$, compute the mean flow field $\bar{y}^N$ of $\tilde{\mathcal{Y}}$ and set $\mathcal{Y} = \{y_i^N - u(t_i)y_b^N - \bar{y}^N\}_{i=1}^P$.

3. Use the input ensemble $\mathcal{Y}$ for the computation of a POD basis $\mathcal{Y}^{POD} = \{\psi_j\}_{j=1}^M$.

4. Compute the reduced order model by projecting the Navier–Stokes equations onto the POD subspace $span\, \mathcal{Y}^{POD}$.

5. Solve the reduced order model for the expansion coefficients $\mathbf{y}^M$ and compute the reduced order solution $y^M(x,t)$, i.e.

   (a) in case of homogeneous boundary conditions: Set

   $$y^M(x,t) = \sum_{j=1}^{M} \mathbf{y}_j^M(t)\psi_j(x),$$

   (b) in case of time-independent (nonhomogeneous) boundary conditions: Set

   $$y^M(x,t) = \bar{y}^N(x) + \sum_{j=1}^{M} \mathbf{y}_j^M(t)\psi_j(x),$$

   (c) in case of time-dependent boundary conditions: Set

   $$y^M(x,t) = \bar{y}^N(x) + u(t)y_b^N(x) + \sum_{j=1}^{M} \mathbf{y}_j^M(t)\psi_j(x).$$

3.6 Numerical Examples: Simulation of Cavity Flows

In this section we present numerical results that serve to illustrate some characteristic features of POD based reduced order models. We focus on the two-dimensional driven cavity flow as model problem, see Section 2.4, which has also been used in Peterson [88], Ito/Ravindran [57] or Allan [7] for reduced order modelling of fluid flows, but not in connection with the proper orthogonal decomposition. The required snapshot computations were carried out with the finite element flow solver FEATFLOW, see Appendix A.

Example 1. For the simulation of the standard cavity flow we set $T = 5$, $Re = 200$ and $y_0(x) = 0$, $x \in \Omega$. With a uniform time discretization of $\delta t = 0.25$ we first produced $P = 20$ snapshots, where we used a uniform 33×33 grid for the spatial discretization. Thus, we have $N = 1089$.

The first 8 singular values of the data matrix A according to (3.6) and (3.10) (and corresponding to the input ensemble $\{y_i^N - \bar{y}^N\}_{i=1}^P$) are

$$\begin{aligned} &\sigma_1 = 0.2312, \quad \sigma_2 = 0.0827, \quad \sigma_3 = 0.0353, \quad \sigma_4 = 0.0139, \\ &\sigma_5 = 0.0055, \quad \sigma_6 = 0.0021, \quad \sigma_7 = 0.0007, \quad \sigma_8 = 0.0003. \end{aligned}$$

The resulting POD basis functions are illustrated in Figure 3.1. The vortex that evolves in the cavity due to the horizontal top wall movement is clearly reflected in the POD basis. Furthermore, the POD basis functions represent higher frequency components when they correspond to smaller singular values, cf. Hansen [46] in the SVD context.

For the comparison of the quality of a POD approximation y^M to y^N we use both the theoretical measure given in (3.25) as well as the (average) reconstruction error

$$\bar{\varepsilon}_{ROM}(M) = \frac{1}{P} \sum_{i=1}^{P} \|y_i^N - y_i^M\|_{\mathcal{M}}^2. \tag{3.34}$$

While (3.25) is obtained by projecting the snapshots itself onto the POD subspace, (3.34) is based on the simulation of the input flow by means of the corresponding POD based reduced order model.

Table 3.1 compares the theoretical and practical error results for the POD based reduced order model for this input data based on (3.25), (3.34). Obviously, the order of magnitude of $\bar{\varepsilon}_{ROM}(M)$ does not decrease corresponding to $\bar{\varepsilon}(M)$, such that it is likely that a kind of systematic error occurs.

Since it might be possible to improve the reduced order model's simulation capacity by providing improved input data, we decided to increase the number of snapshots in the input ensemble, P, in order to capture the flow behaviour in a better way.

The results in Table 3.2 were computed in order to investigate how an enlargement of the snapshot set affects the quality of the reduced order model with respect to the agreement of theoretical and practical error between finite element solution and reduced order solution. As soon as we achieved a certain error level $\bar{\varepsilon}_{ROM}(M)$ that could not be improved by increasing M we stopped this analysis.

We recall that for the computation of $\bar{\varepsilon}(M)$, $\bar{\varepsilon}_{ROM}(M)$ the squared errors between a snapshot and its POD approximation have to be summed up. Therefore, in order to be able to compare the errors of all the different POD based reduced order models, we also list

$$\bar{\varepsilon}_{P=320}(M_L^\star) = \frac{1}{320}\sum_{i=1}^{320} \|y_i^N - y_i^{M_L^\star}\|_{\mathcal{M}}^2$$

in Table 3.3. For this we used the putative best POD subspaces of order $M_L^\star$ corresponding to $L = |\mathcal{Y}| = 20, \ldots, 320$ snapshots for the computation of approximate solutions to all $P = 320$ snapshots.

Table 3.3 illustrates that increasing the number of snapshots in the input ensemble from 20 to 80 yields better POD based approximations to all snapshots in an average sense. The error $\bar{\varepsilon}_{P=320}(M_{20}^\star) = 7.4674 \cdot 10^{-5}$ is reduced to $\bar{\varepsilon}_{P=320}(M_{80}^\star) = 9.7879 \cdot 10^{-6}$. On the other hand the results for $L = 160, 320$ are worse than for $L = 80$. We assign this to the nature of our model problem.

Since we analyze the evolving vortex due to the top wall movement starting at rest, with a refined time discretization and thus a growing number of snapshots we emphasize the vortex itself in the input ensemble. Consequently, the reduced order model is less accurate in the starting phase of the simulation process, cf. Figure 3.2 where we plotted $\|y_i^N - y_i^{M_L^\star}\|_{\mathcal{M}}$ over time.

This demonstrates that already for the choice of the input ensemble care has to be taken such that the snapshots are really representative for the flow behaviour that has to be modelled.

Nevertheless, this relatively simple example illustrates the most important feature of POD based reduced order modelling. Often, a few POD basis functions suffice to approximate a high-order finite element solution up to a certain accuracy.

For this cavity flow example, we have been able to replace the finite element system for $2N = 2178$ velocity unknowns by a corresponding low-dimensional model of order $M = 6$ (for $P = 80$). Once the POD based reduced order model has been derived, the computation of a reduced order solution can be performed at much lower computational cost than working with the finite element model.

While the computation of the finite element snapshots with FEATFLOW takes about 1500 s CPU-time on a SUN SPARCstation 20, the computation of the POD based reduced

order solution with the MATLAB ODE solver *ode15s* takes about 3.5 s CPU-time.

Since we intend to use the POD approach for flow control, we try to illustrate that a POD based reduced order model that has been computed for a specific flow configuration can also be used for flow configurations with variied input parameters. But, there are also limitations for the application of a specific POD based reduced order model.

In the sequel, we keep the POD based reduced order model for $P = 80$, $M = 6$, that has been used for the simulation of the standard cavity flow with horizontal wall velocity equal to 1 (Example 1) and try to simulate cavity flows that correspond to different top wall velocities (Examples 2-4).

Example 2. We assume that the new flow configuration is characterized by the boundary forcing $g(x,t) = (1.5, 0)^T$ on the top wall of the cavity. The direction of the forcing is the same, but it is more intensive. Consequently, the evolving vortex will be more distinct. In Figure 3.3 we compare a snapshot of this flow configuration computed by FEATFLOW with its approximation based on the POD reduced order model of Example 1. The model is able to reproduce the flow dynamics, or at least its principal behaviour.

Example 3. We set $g(x,t) = (0.1, 0)^T$ and make the same comparison between snapshot and reduced order approximation as above, see Figure 3.4. Figure 3.4 points out that the POD based reduced order model of Example 1 is also suitable for reproducing the main dynamics for this flow configuration.

Example 4. The situation changes significantly, if we choose wall velocities that produce more different flow fields, see Figure 3.5. Here, we consider the boundary velocity $g(x,t) = (-1, 0)^T$, i.e. the direction of the forcing has been changed. The POD basis elements corresponding to $g(x,t) = (1, 0)^T$ are not suitable for presenting these dynamics.

Based on the numerical results presented in Examples 1–4, we expect to be able to employ reduced order models in an optimal control setting. Obviously, reliable approximate flow solutions can be obtained as long as we stay sufficiently close to the flow configuration that served to derive a specific POD based reduced order model.

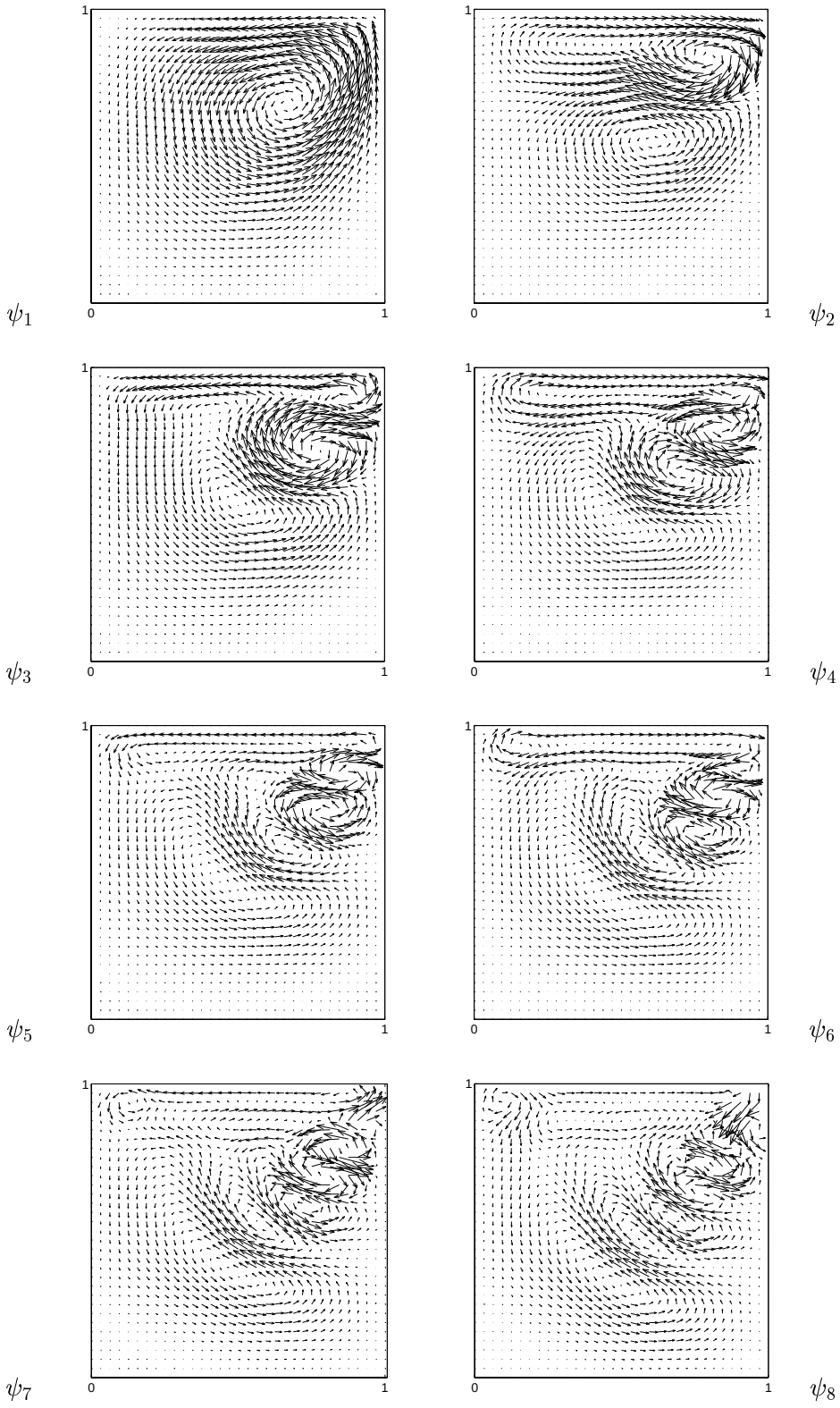

Figure 3.1: Example 1: POD Basis Functions $\psi_1 - \psi_8$

M	$\bar{\varepsilon}(M)$	$\bar{\varepsilon}_{ROM}(M)$
1	4.6150 E-4	4.7459 E-4
2	8.1850 E-5	9.8253 E-5
3	1.2687 E-5	3.5451 E-5
4	1.9699 E-6	3.5192 E-5
5	2.8756 E-7	3.4037 E-5

Table 3.1: Example 1: Comparison between Theoretical and Practical Error

P	M	$\bar{\varepsilon}(M)$	$\bar{\varepsilon}_{ROM}(M)$	P	M	$\bar{\varepsilon}(M)$	$\bar{\varepsilon}_{ROM}(M)$
20	1	4.6150 E-4	4.7459 E-4	40	1	5.7538 E-4	5.7752 E-4
	2	8.1850 E-5	9.8253 E-5		2	1.1975 E-4	1.2545 E-4
	3	1.2687 E-5	3.5451 E-5		3	2.4232 E-5	3.1423 E-5
	4	1.9699 E-6	3.5192 E-5		4	4.6565 E-6	1.6430 E-5
	5	2.8756 E-7	3.4037 E-5		5	9.5004 E-7	1.3831 E-5
					6	1.7193 E-7	1.3072 E-5
80	1	6.2842 E-4	6.3300 E-4	160	1	6.1518 E-4	6.2436 E-4
	2	1.3968 E-4	1.4839 E-4		2	1.4164 E-4	1.5907 E-4
	3	3.1121 E-5	3.8782 E-5		3	3.3239 E-5	5.0274 E-5
	4	6.3274 E-6	1.5098 E-5		4	7.2048 E-6	2.1407 E-5
	5	1.4053 E-6	9.6138 E-6				
	6	3.1051 E-7	8.3010 E-6				
320	1	5.8931 E-4	5.9806 E-4				
	2	1.3427 E-4	1.4656 E-4				
	3	3.1254 E-5	4.3887 E-5				
	4	6.7632 E-6	1.7273 E-5				

Table 3.2: Example 1: Comparison between Theoretical and Practical Error (Enlarged Input Ensemble)

	$\bar{\varepsilon}_{P=320}(M_L^\star)$		
L	$M_L^\star = 4$	$M_L^\star = 5$	$M_L^\star = 6$
20	-	7.4674 E-5	-
40	-	-	1.8853 E-5
80	-	-	9.7879 E-6
160	2.1149 E-5	-	-
320	1.7273 E-5	-	-

Table 3.3: Example 1: Comparison of all Reduced Order Models

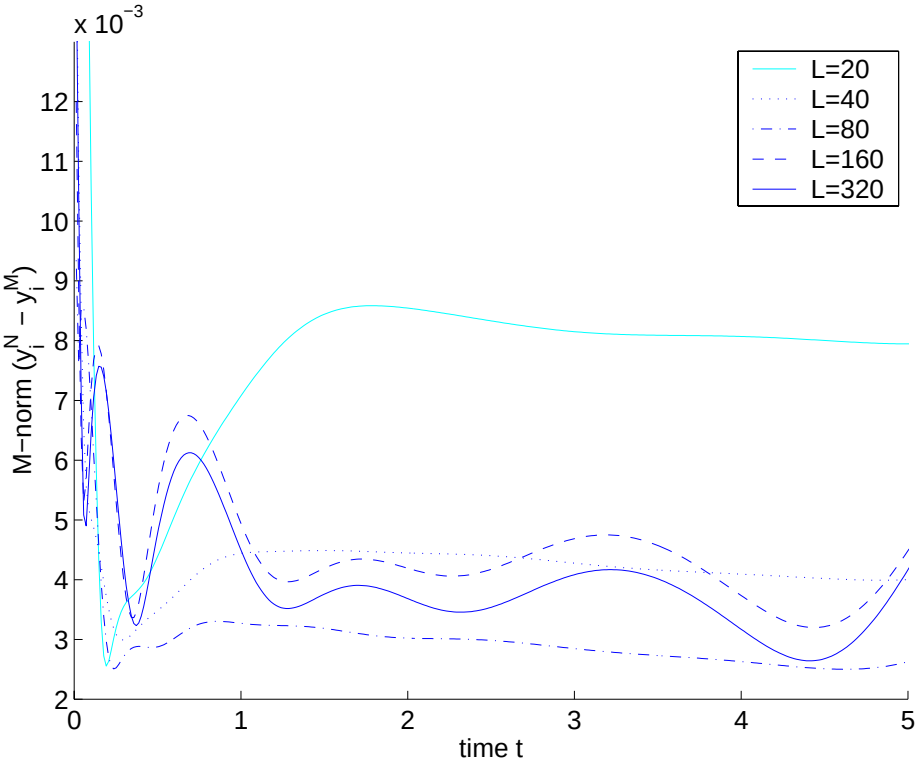

Figure 3.2: Example 1: Comparison of $\|y_i^N - y_i^{M_L^\star}\|_{\mathcal{M}}$

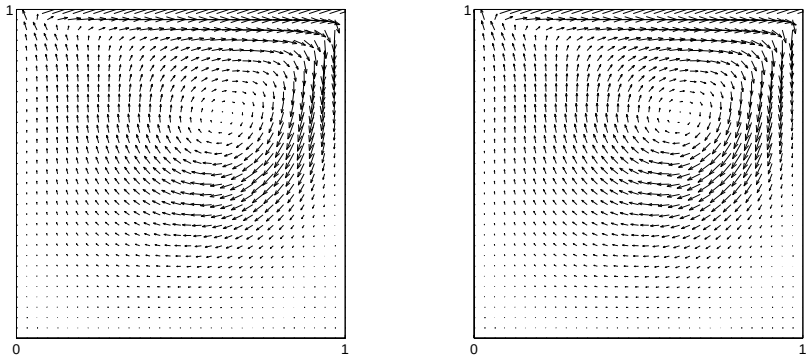

Figure 3.3: Example 2: Snapshot (Left) and its POD Reduced Order Approximation (Right) Based on Model of Example 1, $t=4$

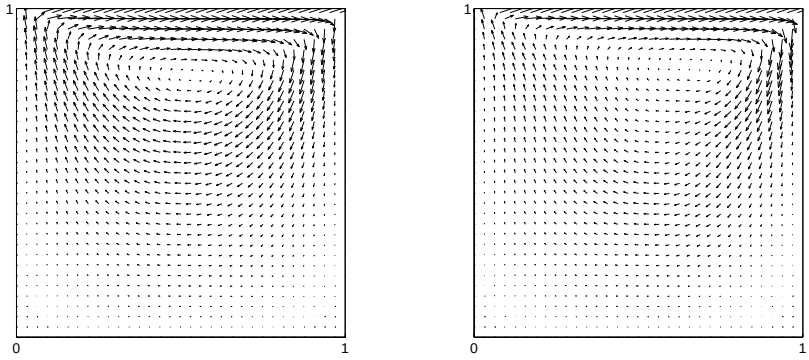

Figure 3.4: Example 3: Snapshot (Left) and its POD Reduced Order Approximation (Right) Based on Model of Example 1, $t=4$

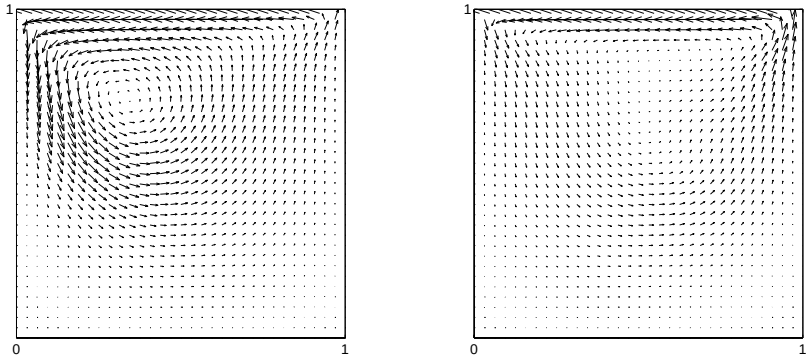

Figure 3.5: Example 4: Snapshot (Left) and its POD Reduced Order Approximation (Right), Based on Model of Example 1, $t=4$

Chapter 4

Lanczos Methods for POD Basis Computations

4.1 Introduction

The computation of POD basis functions by means of the singular value decomposition has the drawback that a truncated SVD cannot be computed directly. Given a data matrix, it is not possible to extract only the largest singular values and corresponding left singular vectors without computing the whole SVD if one uses a standard QR–type algorithm, cf. Golub/Van Loan [42]. This can lead to an overhead of computational work if only the singular vectors associated to a few of the largest singular values of a large-scale data matrix are wanted as it is often the case for POD basis computations. Two main classes of algorithms are in use for the computation of an approximate TSVD

1. algorithms based on QR factorizations, and
2. Lanczos algorithms.

A comprehensive survey of QR-type algorithms can be found in Hansen [46]. Since we are interested in large-scale applications and in view of the efficiency of Lanczos methods, we will concentrate on the second class of algorithms.

In principle, the Lanczos method is an iterative technique that can be used to solve large, symmetric eigenvalue problems of the form

$$Bz = \lambda z \tag{4.1}$$

where information about the extremal eigenvalues of B tends to emerge long before all of the eigenvalues have been computed [42]. It only uses matrix-vector operations and this makes it attractive for large scale applications, especially if B is sparse.

Since we are interested in applying the Lanczos technique for the computation of POD basis functions in a SVD context, we first review the relationships between the

singular value decomposition and symmetric eigenvalue problems that form the basis for the discussion in this chapter, cf. e.g. Cullum/Willoughby [25].

Lemma 4.1 (Relationships between SVD and Eigenvalue Problems)
Let $A \in \mathbb{R}^{m\times n}$, $m \geq n$, with SVD $A = U\Sigma V^T$, where $\{u_i\}_{i=1}^m$ form the columns of $U \in \mathbb{R}^{m\times m}$, $\Sigma = diag(\{\sigma_i\}_{i=1}^n) \in \mathbb{R}^{m\times n}$ and $\{v_i\}_{i=1}^n$ form the columns of $V \in \mathbb{R}^{n\times n}$.

Then, the following results hold:

1. *The eigenvalues of the symmetric matrix $B_1 = A^TA \in \mathbb{R}^{n\times n}$ are σ_i^2. The right singular vectors v_i are corresponding orthonormal eigenvectors.*

2. *The eigenvalues of the symmetric matrix $B_2 = AA^T \in \mathbb{R}^{m\times m}$ are σ_i^2 and $m-n$ zeros. The left singular vectors u_i are corresponding orthonormal eigenvectors for the eigenvalues σ_i^2.*

3. *The eigenvalues of the symmetric matrix*

$$B_3 = \begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix} \in \mathbb{R}^{(m+n)\times(m+n)}$$

are $\sigma_i, -\sigma_i$ and $m-n$ zeros. The vectors $\frac{1}{\sqrt{2}}(u_i, v_i)^T$ and $\frac{1}{\sqrt{2}}(u_i, -v_i)^T$ are corresponding orthonormal eigenvectors for the eigenvalues $\sigma_i, -\sigma_i$, respectively.

Lemma 4.1 shows that, at least from a theoretical point of view, the computation of singular values and singular vectors of a rectangular matrix is equivalent to the computation of the eigenvalues and eigenvectors of several specific symmetric eigenproblems.

In Section 4.2 we review the ideas of the Lanczos method and point out some specific features of this approach. Since we are interested in applying the Lanczos idea for the computation of POD basis functions, we also present convergence results for the Lanczos algorithm of Saad [97, 98]. In Section 4.3 we introduce a Lanczos algorithm that is tailored to the POD approach and discuss its efficiency with respect to other approaches for the computation of the POD basis. Numerical examples confirming our theoretical considerations are given in Section 4.4. At this point, we also refer to [33] where a presentation of these results can be found.

4.2 The Lanczos Algorithm

According to Lemma 4.1 it is possible to compute the TSVD of a given matrix by applying the Lanczos method to an equivalent eigenvalue problem. In this section we review the underlying ideas and properties of the Lanczos method for solving (4.1) with symmetric $B \in \mathbb{R}^{n\times n}$ whose implementation is based on the Rayleigh–Ritz procedure on a sequence of Krylov subspaces.

4.2.1 The Rayleigh-Ritz Approach

For symmetric matrices the Rayleigh quotient provides a useful approach to eigenvalue/-vector computations.

Definition 4.2 (Rayleigh Quotient)
Let $B \in \mathbb{R}^{n\times n}$ be symmetric. For any nonzero vector $z \in \mathbb{R}^n$ we define the *Rayleigh quotient* by

$$\mathrm{r}(z) = \frac{z^T B z}{z^T z}. \tag{4.2}$$

We note that the gradient of r is given by

$$\nabla \mathrm{r}(z) = \frac{2}{z^T z}[Bz - \mathrm{r}(z)z] \tag{4.3}$$

such that each eigenvector of B denotes a stationary point of $\mathrm{r}(z)$ and according to the Courant-Fischer Minimax Theorem (see [42], Th. 8.1.2) the maximization of the Rayleigh quotient (4.2) yields

$$\lambda_{max} = \mathrm{r}(z^\star) = \max_{z\neq 0} \mathrm{r}(z) \tag{4.4}$$

where λ_{max} denotes the largest eigenvalue of B.

If the maximization of $\mathrm{r}(z)$ is restricted to a subspace with orthonormal basis $\{q_1, \ldots, q_j\}$ we can substitute $z = Q_j s$ with matrix $Q_j = [\, q_1, \ldots, q_j \,] \in \mathbb{R}^{n\times j}$ in (4.4) and obtain

$$\max_{s\neq 0} \mathrm{r}(Q_j s) \tag{4.5}$$

with $s \in \mathbb{R}^j$. For $j = n$, (4.5) equals (4.4) and delivers the exact eigenvalue/eigenvector pair $(\lambda_{max}, z^\star)$. However, with the aid of (4.5) a good approximation to λ_{max} can be expected if there is a stationary point $s_j^\star$ of $\mathrm{r}(Q_j s)$ such that $Q_j s_j^\star$ is a good approximation to $z^\star$. Furthermore, if for given $\{q_1, \ldots, q_j\}$ such an approximation is not satisfactory, we should try to construct a vector q_{j+1} such that the corresponding Rayleigh quotient yields $\mathrm{r}(Q_j s_j^\star) \leq \mathrm{r}(Q_{j+1} s_{j+1}^\star) \leq \lambda_{max}$, where $s_{j+1}^\star$ denotes a stationary point for (4.5) with enlarged basis matrix, Q_{j+1}.

In order to achieve improved Rayleigh quotient values, it is possible to use first order information of problem (4.5). By requiring $\nabla \mathrm{r}(Q_j s_j^\star) \in span\{q_1, \ldots, q_j, q_{j+1}\}$ and since (4.3) also implies $\nabla \mathrm{r}(z) \in span\{z, Bz\}$, recursively this leads to the problem: Find q_{j+1} such that

$$span\{q_1, \ldots, q_j, q_{j+1}\} = span\{q_1, Bq_1, \ldots, B^j q_1\}. \tag{4.6}$$

We now introduce the notion of Krylov subspaces.

Definition 4.3 (Krylov Subspace)
Let $B \in \mathbb{R}^{n\times n}$ be symmetric and $q \in \mathbb{R}^n$, $q \neq 0$. We call

$$\mathcal{K}_m(q, B) = span\{q, Bq, \ldots, B^{m-1} q\}$$

the (m-th) *Krylov subspace* (induced by B and q).

Consequently, the relationship (4.6) indicates that we have to compute an orthonormal basis of the Krylov subspace $\mathcal{K}_{j+1}(q_1, B)$ where q_1 denotes the starting vector of this scheme.

4.2.2 The Basic Lanczos Iteration

The orthonormal basis of the Krylov subspace in (4.6) can be efficiently derived by exploiting a tridiagonalization of B. For illustration of the idea, we assume that $Q = Q_n$ is chosen such that

$$Q^T B Q = T \quad \text{with } Qe_1 = q_1, \quad e_1 = (1, 0, \ldots, 0)^T, \tag{4.7}$$

where

$$T = \begin{pmatrix} \alpha_1 & \beta_1 & 0 & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \beta_{n-1} \\ 0 & \cdots & 0 & \beta_{n-1} & \alpha_n \end{pmatrix}.$$

Hence, the orthonormal matrix Q tridiagonalizes B where its first column consists of the starting vector q_1. Using $BQ = QT$, (4.7) guarantees

$$[\, q_1 \,,\, Bq_1 \,, \ldots,\, B^{n-1} q_1] = Q\, [\, e_1 \,,\, Te_1 \,, \ldots,\, T^{n-1} e_1]. \tag{4.8}$$

Thus, (4.8) shows that the columns of Q form an orthonormal basis for the Krylov subspace $\mathcal{K}_n(q_1, B)$.

Moreover, the q_j can be calculated iteratively. Using (4.7), and comparing the columns of $BQ = QT$ leads to the *three-term recurrence*

$$Bq_j = \beta_{j-1} q_{j-1} + \alpha_j q_j + \beta_j q_{j+1} \tag{4.9}$$

with $\beta_0 q_0 \equiv 0$.

Based on the recurrence formula (4.9), the basic Lanczos iteration can be formulated as shown in Algorithm 4.4. Obviously, (4.7) shows that each α_j is given by $\alpha_j = q_j^T B q_j$ for known q_j (Steps 1, 2 in Alg. 4.4). For given β_{j-1}, α_j and q_j, (4.9) yields the relationship

$$\beta_j q_{j+1} = Bq_j - \beta_{j-1} q_{j-1} - \alpha_j q_j \tag{4.10}$$

for the computation of q_{j+1} such that we can use β_j for normalizing q_{j+1} (Steps 3, 4 in Alg. 4.4).

Algorithm 4.4 (Basic Lanczos Iteration)
Initialization: Choose $\beta_0 = 0$, $q_0 = 0$ and $b \neq 0$. Set $q_1 = b/\|b\|_2$.

For $j = 1, 2, 3, \ldots$ repeat

1. Compute $\quad \mathsf{v} = Bq_j$
2. Compute $\quad \alpha_j = q_j^T \mathsf{v}$
3. Set $\quad \mathsf{v} = \mathsf{v} - \beta_{j-1} q_{j-1} - \alpha_j q_j \quad$ and $\quad \beta_j = \|\mathsf{v}\|_2$
4. Set $\quad q_{j+1} = \mathsf{v}/\beta_j$

Thus, starting with an arbitrary vector q_1 of norm $\|q_1\|_2 = 1$, after k Lanczos iteration steps Alg. 4.4 has produced a partial tridiagonalization, $T_k \in \mathbb{R}^{k \times k}$, of the symmetric matrix B. According to (4.7) and (4.10), T_k satisfies

$$BQ_k - Q_k T_k = \beta_k q_{k+1} e_k^T \tag{4.11}$$

where e_k denotes the k-th unit vector in $\mathbb{R}^k$. This tridiagonal matrix T_k permits an efficient computation of its eigenvalues, $\lambda_i^{(k)}$, and eigenvectors, $v_i^{(k)}$, such that

$$T_k V_k = V_k \Lambda_k \tag{4.12}$$

with $\Lambda_k = diag(\lambda_i^{(k)})$ and $V_k = [v_1^{(k)}, \ldots, v_k^{(k)}]$ holds. Hence, (4.12) delivers approximations $(\lambda_i^{(k)}, z_i^{(k)})$ to the sought eigenvalue/-vector pairs (λ_i, z_i) of B through the relationship

$$(V_k Q_k)^T B Q_k V_k = \Lambda_k \qquad \text{and} \quad z_i^{(k)} = Q_k v_i^{(k)}. \tag{4.13}$$

Consequently, increasing the number of Lanczos iterations enlarges the subspace where we are looking for approximations to the wanted eigenvectors belonging to the largest eigenvalues of B.

Remark 4.5
Unfortunately, the basic form of the Lanczos iteration in Alg. 4.4 suffers from numerical instability. Typically, a loss of orthogonality between the *Lanczos vectors* q_j can be observed such that an additional reorthogonalization procedure is necessary to stabilize the Lanczos algorithm. For a detailed description of this phenomenon we refer to Parlett [87].

4.2.3 A Stopping Criterion

A stopping criterion for Algorithm 4.4 can be established based on the residual norm $\|Bz_i^{(k)} - \lambda_i^{(k)} z_i^{(k)}\|_2$ without computing this residual explicitly, cf. Parlett [87]. Since from (4.13), (4.11) and $\|q_{k+1}\|_2 = 1$, we obtain

$$\begin{aligned} \|Bz_i^{(k)} - \lambda_i^{(k)} z_i^{(k)}\|_2 &= \|(BQ_k - \lambda_i^{(k)} Q_k) v_i^{(k)}\|_2 \\ &= \|(BQ_k - Q_k T_k) v_i^{(k)}\|_2 \\ &= \|\beta_k q_{k+1} e_k^T v_i^{(k)}\|_2 \\ &= \beta_k |\, e_k^T v_i^{(k)}|, \end{aligned} \tag{4.14}$$

the pair $(\lambda_i^{(k)}, z_i^{(k)})$ is sufficiently converged, if $\beta_k |\, e_k^T v_i^{(k)}|$ is smaller than some predefined threshold value.

4.2.4 Convergence of the Lanczos Iteration

In [97] the convergence behaviour of the approximate eigenvalues/eigenvectors to the desired eigenvalues/eigenvectors of B has been analyzed, where the absence of numerical errors during the computation is assumed. We review the main results what requires some additional definitions.

Definition 4.6 (Angle between Vector and Subspace)
The *acute angle* $\theta(z, S_m)$ between a nonzero vector $z \in \mathbb{R}^n$ and a m-dimensional subspace S_m is defined by

$$\theta(z, S_m) = \arcsin\left(\frac{\|z - \mathcal{P}_S z\|}{\|z\|}\right), \tag{4.15}$$

where $\mathcal{P}_S z$ denotes the projection of z onto S_m.

Definition 4.7 (Chebyshev Polynomial)
For $t \geq 1$ we define the *Chebyshev polynomial* of the first kind of degree j by

$$C_j(t) = \frac{1}{2}\left[\left(t + \sqrt{t^2 - 1}\right)^j + \left(t + \sqrt{t^2 - 1}\right)^{-j}\right]. \tag{4.16}$$

We emphasize that $C_j(t)$ sharply increases for $t > 1$, if j is sufficiently large. The Chebyshev polynomial (4.16) and also the definition of the angle between a vector and a subspace (4.15) are useful to describe the convergence behaviour of the Lanczos iteration.

Theorem 4.8 (Saad [97, 98], Distance between $\mathcal{K}_m$ and an Eigenvector)
Let $\lambda_1 > \lambda_2 > \ldots > \lambda_k$ be the largest k $(< n)$ eigenvalues and λ_n the smallest eigenvalue of the symmetric matrix $B \in \mathbb{R}^{n \times n}$. Let $\theta(z_i, \mathcal{K}_m)$ denote the acute angle between the exact

eigenvector z_i associated with λ_i and the Krylov subspace $\mathcal{K}_m$ for $i<k$. Let $C_j(t)$, $t \geq 1$, denote the Chebyshev polynomial of the first kind of degree j given by (4.16) and set

$$\kappa_1 = 1, \quad \kappa_i = \prod_{j=1}^{i-1} \frac{\lambda_j - \lambda_n}{\lambda_j - \lambda_i}, \quad \textit{if } i > 1 \qquad \textit{and} \qquad \gamma_i = \frac{\lambda_i - \lambda_{i+1}}{\lambda_{i+1} - \lambda_n}.$$

Then, $\theta(z_i, \mathcal{K}_m)$ satisfies the inequality

$$\tan(\theta(z_i, \mathcal{K}_m)) \;\leq\; \frac{\kappa_i}{C_{m-i}(1+2\gamma_i)} \tan(\theta(q_1, z_i)). \tag{4.17}$$

Theorem 4.8 illustrates that the angle between a wanted eigenvector z_i and the Krylov subspace $\mathcal{K}_m$ depends on the angle between z_i and the initial Lanczos vector q_1, the gaps between the successive eigenvalues $\lambda_1, \ldots, \lambda_{i+1}$ and λ_n and the dimension of the current Krylov subspace. Here, especially the dimension of the Krylov subspace, m, the location of the considered eigenvalue in the eigenvalue order, i, and the ratio, γ_i, between $\lambda_i - \lambda_{i+1}$ and $\lambda_{i+1} - \lambda_n$ play an important role, since those values determine the value of the Chebyshev polynomial in (4.17). Large m and small i imply that the value of $C_{m-i}(1+2\gamma_i)$ is large for sufficiently separated eigenvalues and thus dominates the estimate for the angle between an eigenvector and the Krylov subspace.

The estimate for the acute angle between an eigenvector and the Krylov subspace (4.17) as a measure of distance between both can be used to estimate the relationship between an exact eigenvector and an eigenvector approximation based on a Lanczos procedure.

Theorem 4.9 (Saad [97, 98], Convergence of the Eigenvectors)
Let $B \in \mathbb{R}^{n \times n}$ be a symmetric matrix with i-th eigenvalue λ_i and corresponding eigenvector z_i. Let $(\lambda_i^{(m)}, z_i^{(m)})$ be the approximate eigenvalue/eigenvector pair computed with the Lanczos method on the Krylov subspace $\mathcal{K}_m = span\{q_1, Bq_1, \ldots, B^{m-1}q_1\}$ and β_m as defined in Alg. 4.4. Let $\theta(z_i, z_i^{(m)})$ denote the acute angle between z_i and $span\{z_i^{(m)}\}$, $\theta(z_i, \mathcal{K}_m)$ the acute angle between z_i and the Krylov subspace $\mathcal{K}_m$. Furthermore, we define $\delta_{i,m} = \min_{j \neq i} |\lambda_i - \lambda_j^{(m)}|$.

Then

$$\sin(\theta(z_i, z_i^{(m)})) \leq \sqrt{1 + \frac{\beta_m^2}{\delta_{i,m}^2}} \sin(\theta(z_i, \mathcal{K}_m))$$

holds.

Thus, according to Theorem 4.9 the Lanczos method will deliver good eigenvector approximations if the angle between the desired exact eigenvector and the subspace $\mathcal{K}_m$ is sufficiently small. This angle decreases rapidly to zero as shown in Theorem 4.8.

4.3 An Efficient Method for POD Basis Computations

4.3.1 A POD Tailored Lanczos Method

We now turn to an efficient method for the computation of POD basis functions by means of a truncated singular value decomposition.

As pointed out before the computation of the TSVD by computing the whole SVD and dropping undesired singular values and vectors includes unnecessary computational work. A more efficient method for an accurate approximate computation of the TSVD should reduce the amount of work necessary for the computation of the SVD. Essentially $6mn^2 + 11n^3$ *floating point operations* (flops) are necessary to compute all singular values and the related left singular vectors of $A \in \mathbb{R}^{m\times n}$, adds and multiplications equally weighted, see Golub/Van Loan [42]. Therefore, we turned to the Lanczos scheme, where we expected less computational effort to compute only the required singular values and singular vectors.

From a theoretical point of view all of the symmetric eigenproblem formulations given in Lemma 4.1 are equivalent for computing the SVD. But, from a numerical point of view the third approach offers the most stable way of computing the SVD via an equivalent eigenproblem. Eigenproblems with matrices of the form B_1, B_2 as presented in Lemma 4.1 can suffer from ill-conditioning , see Cullum/Willoughby [24].

For this reason, we are going to apply the Lanczos scheme to the symmetric matrix

$$B = \begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix} \quad \in \mathbb{R}^{(m+n)\times(m+n)} \tag{4.18}$$

in order to compute the POD basis functions according to Theorem 3.8. Then, with a suitable starting vector, Algorithm 4.4 produces $\alpha_j = 0$, $j = 1, 2, 3, \ldots$, and reduces to a two–step scheme where the Lanczos vectors q_j alternate between the forms $(p_j, 0)^T$ and $(0, r_j)^T$. In order to achieve this we have to split the Lanczos vectors q_j into components p_j and r_j corresponding to the blocks in B and to start Alg. 4.4 with $q_1 = (p_1, 0)^T$. According to the iteration scheme presented in Alg. 4.4 a basic Lanczos iteration step with input $q_{j-1} = (0, r_{j-1})^T$, $q_j = (p_j, 0)^T$ results in

$$\begin{aligned}
\mathsf{v} &= \begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}\begin{pmatrix} p_j \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ A^T p_j \end{pmatrix}, \\
\alpha_j &= (p_j^T, 0)\begin{pmatrix} 0 \\ A^T p_j \end{pmatrix} = 0, \\
\mathsf{v} &= \begin{pmatrix} 0 \\ A^T p_j \end{pmatrix} - \beta_{j-1}\begin{pmatrix} 0 \\ r_{j-1} \end{pmatrix}, \quad \beta_j = \|\mathsf{v}\|_2, \\
q_{j+1} &= \mathsf{v}/\beta_j.
\end{aligned}$$

Reversely, a Lanczos step with input $q_{j-1}=(p_{j-1},0)^T$, $q_j=(0,r_j)^T$ yields

$$\mathsf{v} = \begin{pmatrix} Ar_j \\ 0 \end{pmatrix} - \beta_{j-1}\begin{pmatrix} p_{j-1} \\ 0 \end{pmatrix}, \quad \beta_j = \|\mathsf{v}\|_2. \tag{4.19}$$

and it has to be emphasized that this block structure of the Lanczos vectors q_j is maintained during the iteration process. Since each α_j vanishes, it is also possible to rewrite this iteration scheme as a bidiagonalization scheme, cf. Paige [83].

But more important is the fact that if we substitute the normalization step with respect to $\|\cdot\|_2$ by a normalization with respect to $\|\cdot\|_{\mathcal{M}}$ in (4.19) according to (3.8) then we obtain an algorithm that allows to compute approximate POD basis coefficient vectors directly such that these are orthonormal with respect to the $\mathcal{M}$-inner product.

The orthonormality condition for the POD basis functions requires $\Psi^T\mathcal{M}\,\Psi = I_M$, see (3.17), (3.18). For ease of notation we now assume that the snapshot data matrix according to (3.6) is given by $Y \in \mathbb{R}^{m\times n}$, $m>n$, with $rank(Y)=n$, and that $M=n$ POD basis functions are wanted. The SVD of $A = (\mathcal{M}^{1/2})^T Y$ and (3.18) yield

$$A = (\mathcal{M}^{1/2})^T Y = U_M \Sigma_M V^T = (\mathcal{M}^{1/2})^T \Psi \Sigma_M V^T.$$

Therefore, the orthonormality of U_M with respect to $\|\cdot\|_2$ also gives the orthonormality of Ψ with respect to $\|\cdot\|_{\mathcal{M}}$.

This implies that we neither have to compute the Cholesky factor of the FE mass matrix $\mathcal{M}$ to build up $A = (\mathcal{M}^{1/2})^T Y$ nor do we have to solve linear systems of the type (3.18) to get POD basis functions in their FE basis representation. The $\mathcal{M}$–orthonormality of Ψ can be achieved by substituting each normalization in steps such as (4.19) (with respect to $\|\cdot\|_2$) by a normalization with respect to $\|\cdot\|_{\mathcal{M}}$. At this, we recall that it is the upper part p_j of q_j that corresponds to the left singular vectors of A (cf. Lemma 4.1). Taking this different orthonormalization step into account we get the two-step iteration scheme presented in Algorithm 4.10.

For the stabilization of this process we additionally have to employ, e.g., a Gram–Schmidt procedure for the reorthogonalization. Often only a small number of POD basis functions is sought such that this reorthogonalization strategy is acceptable if the number of Lanczos steps, k, remains small.

After k iteration steps Algorithm 4.10 has computed scalars β_j, $j=1,\ldots,2k+1$, such that the largest $\ell\,(\leq k)$ positive eigenvalues of

$$T_{2k+1} = \begin{pmatrix} 0 & \beta_2 & & \cdots & 0 \\ \beta_2 & 0 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \beta_{2k+1} \\ 0 & \cdots & & \beta_{2k+1} & 0 \end{pmatrix} \in \mathbb{R}^{(2k+1)\times(2k+1)}$$

Algorithm 4.10 (POD Lanczos Iteration)
Initialization: Choose $\beta_1 = 0$ and $p_1 = b$ with $\|b\|_{\mathcal{M}} = 1$. Set $r_0 = 0$.

For $j = 1, 2, 3, \ldots$ repeat

$$
\begin{aligned}
\mathsf{v}^{(r)} &= Y^T \mathcal{M} p_j \\
\mathsf{v}^{(r)} &= \mathsf{v}^{(r)} - \beta_{2j-1} r_{j-1} \qquad \textit{(lower block)} \\
\beta_{2j} &= \|\mathsf{v}^{(r)}\|_2 \\
r_j &= \mathsf{v}^{(r)} / \beta_{2j}
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{v}^{(p)} &= Y r_j \\
\mathsf{v}^{(p)} &= \mathsf{v}^{(p)} - \beta_{2j} p_j \qquad \textit{(upper block)} \\
\beta_{2j+1} &= \|\mathsf{v}^{(p)}\|_{\mathcal{M}} \\
p_{j+1} &= \mathsf{v}^{(p)} / \beta_{2j+1}
\end{aligned}
$$

serve as approximations to the first ℓ singular values of $A = (\mathcal{M}^{1/2})^T Y \in \mathbb{R}^{m \times n}$. The corresponding $M\,(\leq \ell)$ POD basis functions can be computed from the upper part of the matrix $Z_\ell^{(2k+1)} = Q_{2k+1} V_\ell^{(2k+1)}$, where $Q_{2k+1} \in \mathbb{R}^{(m+n)\times(2k+1)}$ denotes the matrix that collects the Lanczos vectors $(0, r_j)^T$, $(p_{j+1}, 0)^T$ and $V_\ell^{(2k+1)} \in \mathbb{R}^{(2k+1)\times \ell}$ consists of the eigenvectors associated to the largest ℓ eigenvalues of T_{2k+1}, see Lemma 4.1 and Subsection 4.2.2.

4.3.2 Efficiency Considerations

After having introduced the Lanczos scheme for the computation of the POD basis functions it is worth to compare the amount of computational work that is necessary to obtain the desired POD basis functions using Algorithm 4.10 to alternative computational approaches.

The performance of k POD Lanczos iteration steps according to Algorithm 4.10 requires essentially $4kmn + 2k(m+n)$ flops for the computation of the matrix-vector products and the orthonormalization during the course of the iteration for computing r_j, p_{j+1}. The complete reorthogonalization scheme based on a Gram–Schmidt procedure additionally requires approximately $k^2(m+n)$ flops, but this effort can be further reduced using, e.g., a selective reorthogonalization scheme. In comparison with this the computational work for the solution of the eigenvalue problem with T_k is negligible. Therefore, Algorithm 4.10 requires less computational work (approximately $4kmn + (2k+k^2)(m+n)$ flops) than employing a standard SVD solver (essentially $6mn^2 + 11n^3$ flops), if the number of POD Lanczos iteration steps, k, is sufficiently smaller than the number of columns, n, of A. At this, we recall that in the context of POD based reduced order modelling n denotes the number of snapshots in the input ensemble.

For a rectangular matrix $A \in \mathbb{R}^{m \times n}$ where the relation $n \ll m$ holds (in POD computations the number of snapshots, typically, will be much smaller than the dimension of the FE discretization) the reduction of a $m \times n$ SVD problem to a $n \times n$ symmetric eigenproblem (cf. Lemma 4.1, (1)) seems to be attractive and is usually applied for POD based reduced order modelling. But, if A is ill–conditioned, i.e. $\kappa_2(A) = \sigma_1/\sigma_n$ is large, then the condition number of $A^T A$ even increases, since $\kappa_2(A^T A) = \sigma_1^2/\sigma_n^2$, where σ_1, σ_n denote the largest and smallest singular values of A, respectively.

Furthermore, if we want to use a direct method for solving the equivalent eigenproblem

$$A^T A z = \sigma^2 z \tag{4.20}$$

we first have to build $A^T A$ which can be computationally expensive itself.

It is possible to avoid computing $A^T A$ explicitly if (4.20) is also solved by an iterative Lanczos scheme using only matrix-vector multiplications. Nevertheless, in this case we still have to be aware of eventual ill-conditioning. The amount of computational work for this approach is similar to the work necessary during the performance of Alg. 4.10. Again, this requires essentially $4kmn$ flops for the matrix-vector products, merely the normalization steps and a reorthogonalization can be performed at lower cost.

In addition, if we have computed an approximate eigenvalue/eigenvector pair to $(\sigma_i^2,\ z_i)$ of $A^T A$ we also have to compute the corresponding left singular vector of A for the computation of the POD basis function using the relationship (cf. 3.14)

$$u_i = \frac{1}{\sigma_i} A z_i. \tag{4.21}$$

Therefore, in view of an eventual ill-conditioned matrix $A^T A$ the proposed POD Lanczos scheme (Alg. 4.10) provides an efficient and stable method for the computation of POD basis functions.

4.4 Numerical Examples

In this section the properties of the Lanczos method are illustrated based on POD basis function computations for the driven cavity problem presented in Section 2.4.

Example 1: Standard Driven Cavity. The data that has to be processed in the first example is based on flow calculations for the standard driven cavity model problem at Reynolds number $Re = 100$. We used the software package FEATFLOW (see Appendix A) to generate $P = 44$ snapshots where a uniform spatial discretization with $33 \times 33 = 1089$ grid points has been used. Since we are considering a two-dimensional flow, we get a snapshot data matrix $Y \in \mathbb{R}^{2178 \times 44}$.

The first 10 singular values of $A=(\mathcal{M}^{1/2})^T Y$ (computed with the MATLAB solver *svd* which uses the LINPACK routine *ZSVDC*) are

$$\sigma_1=1.0521497,\quad \sigma_2=0.1252215,\quad \sigma_3=0.0361298,\quad \sigma_4=0.0118624,\quad \sigma_5=0.0037027,$$
$$\sigma_6=0.0012489,\quad \sigma_7=0.0003909,\quad \sigma_8=0.0001249,\quad \sigma_9=0.0000390,\quad \sigma_{10}=0.0000116.$$

The fast decrease in the magnitude of the singular values is obvious and following the energy criterion (3.26) we concentrate on the computation of the first 4 singular values and corresponding POD basis functions ($\bar{\epsilon}=0.9999$) using Algorithm 4.10.

Figure 4.1 illustrates how the approximated eigenvalues of B according to (4.18) (typically) evolve and converge during the Lanczos iteration. In the upper part of this figure, we monitor the largest four eigenvalues of the matrices T_k during the Lanczos iteration. The fast convergence to the exact eigenvalues of B is obvious. In order to produce these results we employed a complete reorthogonalization scheme based on a Gram-Schmidt procedure. In Fig. 4.1 we also present the corresponding results if reorthogonalization is not performed. Again, we monitor the largest four eigenvalues of the matrices T_k during the Lanczos iteration. Due to the loss of orthogonality spurious eigenvalues appear (*ghost eigenvalues*). In this example, an unwanted copy of the largest eigenvalue can be found at iteration $k = 6$.

In Table 4.1 the exact singular values compared to its Lanczos approximations at each Lanczos step are tabulated.

Table 4.2 lists the angles $\theta_i = \theta(u_i, \tilde{u}_i)$ in angular degrees between the corresponding exact and approximate singular vectors according to (4.15) as a measure of distance between both.

Since Algorithm 4.10 computes approximations $\pm\sigma_i^{(k)}$, it provides approximations to the first 4 singular values after at least 4 Lanczos steps. After 7 Lanczos steps all of the wanted singular values/vectors have been computed to satisfactory accuracy. The residual norm according to (4.14) is below $1.0\cdot 10^{-7}$.

In order to illustrate the efficiency of Alg. 4.10 we give detailed flop counts for the alternative approaches for solving the underlying SVD problem discussed in Subsection 4.3.2 (see Table 4.3). The total amount of flops is calculated by adding the flop counts for all necessary subproblem solutions.

For the direct solution of the eigenvalue problem (EVP) with $A^T A$ we employed the MATLAB routine *eig* (which is based on EISPACK routines). The SVD was performed using the MATLAB routine *svd* where for $A\in\mathbb{R}^{m\times n}$, $m>n$, only the first n left singular vectors were computed (*economy size version*). According to the discussion in Subsection 4.3.2 the iterative Lanczos procedures are faster than the direct methods. Algorithm 4.10 requires only slightly more flops than the Lanczos method (Algorithm 4.4 plus Gram-Schmidt Reorthogonalization) applied to $A^T A$.

Example 2: A Cavity Control Problem. The second example consists of data for a cavity flow with varying boundary forcing at $Re = 200$. We employed a spatial discretization with 65×65 grid points and generated $P = 182$ snapshots, i.e. $Y \in \mathbb{R}^{8450 \times 182}$. Based on the energy criterion (3.26) we decided to compute $M = 8$ POD basis functions which corresponds to $\bar{\epsilon} = 0.9999$.

Tab. 4.4 shows the corresponding flop count results for example 2.

The results of the Lanczos iteration concerning the angles, θ_i, between corresponding exact and approximate singular vectors are given in Table 4.5, where '$*$' denotes a value less than $1.0 \cdot 10^{-12}$.

Again, the trend in the numerical results illustrates the discussion in Subsection 4.3.2. The Lanczos technique is more efficient than direct methods. The amount of work for Alg. 4.10 is of the same order of magnitude as Alg. 4.4 applied to $A^T A$.

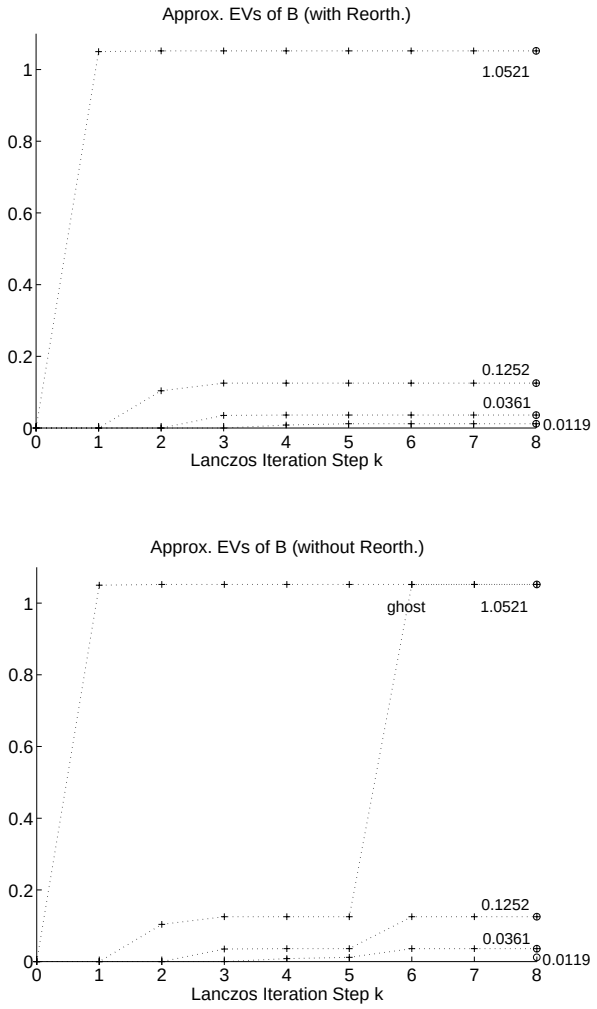

Figure 4.1: Example 1: Approximate Eigenvalues of B (with/without Gram-Schmidt Reorthogonalization)

Lanczos Iteration k	$\sigma_1^{(k)}$	$\sigma_2^{(k)}$	$\sigma_3^{(k)}$	$\sigma_4^{(k)}$
1	1.05004753	-	-	-
2	1.05214957	0.10379310	-	-
3	1.05214965	0.12520965	0.03503557	-
4	1.05214965	0.12522154	0.03612667	0.00831616
5	1.05214965	0.12522154	0.03612975	0.01185789
6	1.05214965	0.12522154	0.03612975	0.01186240
7	1.05214965	0.12522154	0.03612975	0.01186240
exact σ's	1.05214965	0.12522154	0.03612975	0.01186240

Table 4.1: Example 1: Convergence of Singular Values

Lanczos Iteration k	θ_1	θ_2	θ_3	θ_4
1	0.3610	-	-	-
2	0.0017	11.4003	-	-
3	$<$ 1.0 E-7	0.0802	3.5856	-
4	$<$ 1.0 E-11	0.0002	0.2042	18.8103
5	$<$ 1.0 E-13	$<$ 1.0 E-7	0.0001	0.2218
6	$<$ 1.0 E-13	$<$ 1.0 E-12	$<$ 1.0 E-7	0.0004
7	$<$ 2.0 E-13	$<$ 1.0 E-13	$<$ 1.0 E-12	$<$ 1.0 E-7

Table 4.2: Example 1: Convergence of Singular Vectors

	EVP with A^TA	SVD for A	Alg. 4.4 with A^TA	Alg. 4.10
comp. A (3.10)	-	6.51 E+6	-	-
comp. A^TA	5.61 E+6	-	-	-
solve (3.18)	-	5.81 E+5	-	-
solve (4.21)	7.67 E+5	-	7.67 E+5	-
main prob. sol.	6.26 E+5	4.48 E+7	3.29 E+6	4.22 E+6
flops (total)	7.01 E+6	5.19 E+7	4.06 E+6	4.22 E+6

Table 4.3: Example 1: Flop Counts

	EVP with A^TA	SVD for A	Alg. 4.4 with A^TA	Alg. 4.10
comp. A (3.10)	-	2.80 E+8	-	-
comp. A^TA	2.72 E+8	-	-	-
solve (3.18)	-	1.32 E+7	-	-
solve (4.21)	2.27 E+7	-	2.27 E+7	-
main prob. sol.	3.28 E+7	2.47 E+9	1.20 E+8	1.30 E+8
flops (total)	3.27 E+8	2.76 E+9	1.43 E+8	1.30 E+8

Table 4.4: Example 2: Flop Counts

Lanczos Iteration k	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
1	3.0156	-	-	-	-	-	-	-
2	0.0319	15.0261	-	-	-	-	-	-
3	0.0003	4.2519	24.0703	-	-	-	-	-
4	$<$ 2.0 E-7	0.0677	1.9459	46.5498	-	-	-	-
5	$<$ 2.0 E-10	0.0009	0.0749	12.8433	30.2913	-	-	-
6	$*$	$<$ 2.0 E-6	0.0004	0.4138	3.2114	86.7078	-	-
7	$*$	$<$ 2.0 E-9	$<$ 2.0 E-6	0.0071	0.1921	84.4238	89.2339	-
8	$*$	$<$ 2.0 E-11	$<$ 2.0 E-8	0.0007	0.0565	78.2241	79.6811	68.2899
9	$*$	$*$	$<$ 3.0 E-11	$<$ 1.0 E-5	0.0016	2.9343	0.9704	22.2737
10	$*$	$*$	$*$	$<$ 3.0 E-8	$<$ 2.0 E-5	0.0736	0.0310	3.8319
11	$*$	$*$	$*$	$<$ 2.0 E-10	$<$ 3.0 E-7	0.0003	0.0002	0.7101
12	$*$	$*$	$*$	$*$	$<$ 5.0 E-10	$<$ 2.0 E-5	$<$ 2.0 E-5	0.0130
13	$*$	$*$	$*$	$*$	$*$	$<$ 8.0 E-8	$<$ 2.0 E-7	0.0004
14	$*$	$*$	$*$	$*$	$*$	$<$ 5.0 E-10	$<$ 1.0 E-9	$<$ 2.0 E-5
15	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$<$ 2.0 E-8
16	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$

Table 4.5: Example 2: Convergence of Singular Vectors

Chapter 5

Trust-region Methods

5.1 Introduction

In this chapter we consider trust-region methods for the solution of the nonlinear unconstrained optimization problem

$$\min_{u \in \mathbb{R}^n} f(u) \tag{5.1}$$

with a sufficiently smooth objective function $f : \mathbb{R}^n \to \mathbb{R}$. These trust-region methods follow the idea of solving a sequence of 'simpler' subproblems where the objective function f is replaced by model functions.

The material in this chapter serves to introduce the philosophy of trust-region methods that later on can be adapted for using trust-region frameworks in the context of POD based reduced order modelling for optimal control problems.

In Section 5.2 we give an introduction to classical trust-region algorithms with quadratic model functions for the objective f. We present the basic ideas of the trust-region approach and give also a review of classical convergence results for this class of optimization algorithms.

In Section 5.3 we discuss trust-region methods that are based on quadratic model functions with approximate derivative information of the true objective. Since there are many situations where the gradient of f is not available analytically, often ∇f has to be approximated numerically. In such cases, the corresponding quadratic model functions in a trust-region algorithm are built using inexact gradient information. Here, we discuss several ways to take this gradient error into account in the convergence analysis of a corresponding algorithm. Employing an error condition introduced by Carter [20, 21] we obtain a trust-region algorithm based on approximate derivatives of f possessing the same convergence properties as the classical trust-region algorithm.

In Section 5.4 we introduce a trust-region framework that enables us to use general model functions for f that are not necessarily quadratic. In this case, also inexact gradient information can be present. Here, we present an approach to the solution of the trust-

region subproblems that is based on the work of Toint [111]. Furthermore, we give an overview of several important properties of this approach that later on will be helpful for deriving a convergence theory for a trust-region framework in the context of POD based reduced order modelling.

5.2 Quadratic Model Functions: The Standard Approach

Usually, trust-region methods employ quadratic model functions for the iterative solution of the unconstrained optimization problem (5.1) with $f : \mathbb{R}^n \to \mathbb{R}$.

For a given current iterate $u_k \in \mathbb{R}^n$ a quadratic model

$$m_k^{\mathbf{quad}}(u_k + s) = f(u_k) + \nabla f(u_k)^T s + \tfrac{1}{2} s^T H_k s \tag{5.2}$$

for the true objective f around u_k is used during the optimization process. Here, $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$ denotes the gradient of f and the symmetric matrix $H_k \in \mathbb{R}^{n \times n}$ denotes the Hessian of f at u_k (or an approximation to it). Additionally, a *trust-region constraint*

$$\|s\| \leq \delta_k \tag{5.3}$$

with *trust-region radius* $\delta_k > 0$ is employed to define a region where the quadratic model is trusted to adequately represent the true objective. We use $\|s\| = \|s\|_2$, but remark that it is also possible to use other norms.

According to the definition of the model function, we have $m_k^{\mathbf{quad}}(u_k) = f(u_k)$, if $s = 0$. By adjusting the trust-region radius δ_k in (5.3) we adjust the region around u_k where the truncated Taylor expansion of f around u_k given by (5.2) is supposed to model the nonlinearity of f sufficiently. The combination of (5.2) and (5.3) leads to the classical *trust-region subproblem.*

Problem 5.1 (Trust-region Subproblem (TRS))
For a given current iterate u_k and trust-region radius $\delta_k > 0$, solve

$$\min_{s \in \mathbb{R}^n} \; m_k^{\mathbf{quad}}(u_k + s), \quad s.t. \quad \|s\| \leq \delta_k \tag{5.4}$$

approximately, in order to obtain a step s_k.

A priori there is no guarantee that the step computation via TRS leads to an acceptable new point, $u_k + s_k$, in the sense that $f(u_k + s_k) < f(u_k)$. Therefore, a trust-region algorithm incorporates a decision step whether $u_k + s_k$ leads to a sufficient decrease in the true objective or not. For this purpose, it is necessary to evaluate the original objective, if a new trial step s_k is available.

Using $f(u_k + s_k)$ it is possible to compare the *actual reduction*

$$ared_k(s_k) = f(u_k) - f(u_k + s_k) \tag{5.5}$$

to the *predicted reduction*

$$pred_k(s_k) = m_k^{\mathbf{quad}}(u_k) - m_k^{\mathbf{quad}}(u_k + s_k) \tag{5.6}$$

based on the current model $m_k^{\mathbf{quad}}$. Essentially, it is this comparison that gives a measure for the current models prediction capability. Algorithm 5.2 sketches a basic trust-region algorithm that implements these ideas, where we omitted a stopping criterion.

Algorithm 5.2 (Basic Trust-region Algorithm)
Initialization: Choose $0 < \eta_1 < \eta_2 < 1$, $0 < \gamma_1 \leq \gamma_2 < 1 \leq \gamma_3$, an initial trust-region radius $\delta_0 > 0$ and an initial iterate u_0. Set $k = 0$.

1. Build the model $m_k^{\mathbf{quad}}(u_k + s)$ and determine an approximate solution s_k to

$$\min_{\|s\| \leq \delta_k} m_k^{\mathbf{quad}}(u_k + s).$$

2. Compute $f(u_k + s_k)$ and

$$\rho_k = \frac{ared_k(s_k)}{pred_k(s_k)}.$$

3. Update the trust-region radius:

   - If $\rho_k \geq \eta_2$:
     Set $u_{k+1} = u_k + s_k$ and choose $\delta_{k+1} \in [\delta_k, \gamma_3 \delta_k]$.
   - If $\eta_1 \leq \rho_k < \eta_2$:
     Set $u_{k+1} = u_k + s_k$ and choose $\delta_{k+1} \in [\gamma_2 \delta_k, \delta_k)$.
   - If $\rho_k < \eta_1$:
     Set $u_{k+1} = u_k$ and choose $\delta_{k+1} \in [\gamma_1 \delta_k, \gamma_2 \delta_k]$.

4. Set $k = k + 1$ and GOTO 1.

According to Alg. 5.2, we call an iteration *successful* if $\rho_k \geq \eta_1$, i.e., the actual reduction, $ared_k(s_k)$, is large enough compared to the predicted reduction, $pred_k(s_k)$. In the case $\rho_k < \eta_1$, we call the iteration *unsuccessful*.

For a successful iteration the trial step s_k is accepted and the model is updated. Furthermore, the trust-region radius δ_k can be increased (for $\gamma_3 > 1$) if the achieved decrease in f indicates a good behaviour of the model ($\rho_k \geq \eta_2$). If there is only a restraining predicted decrease compared to the actual decrease ($\eta_1 \leq \rho_k < \eta_2$) we have the possibility to decrease the trust-region radius. This serves to restrict the region where we believe the next model function to be reliable.

For unsuccessful iterations we do not accept the trial step s_k, but decrease the trust-region radius and start the step computation (based on the last successful iterate) via TRS

again. With a contraction of the trust-region it is more likely to have a good approximation to the true objective using a truncated Taylor expansion up to second order terms. This implies, that decreasing the trust-region radius allows to improve the model.

However, when we adapt the trust-region according to information of the last step computation, we express whether or not we are satisfied with the performance of the last trial step and if we trust the next model to be reliable in a larger region.

One of the attractive features of the basic trust-region algorithm (Algorithm 5.2) based on quadratic model functions is that an approximate solution to TRS in Step 1 can be computed efficiently. Since the focus of this work is on the philosophy of trust-region methods and its implications for reduced order modelling rather than on a specific implementation for a trust-region subproblem solution with quadratic model functions, we refer to, e.g., Sorensen [103], Dennis/Schnabel [27], Steihaug [105] or Conn et al. [23] for more information on this aspect of trust-region methods.

As suggested earlier, it is the comparison of actual reduction to predicted reduction that allows to quantify the decrease in the true objective and that finally is the basis for a global convergence theory of trust-region methods. In what follows, we briefly sketch the important ideas for proving global convergence of the trust-region scheme based on quadratic model functions of the type (5.2) since similar techniques can be applied to more general trust-region frameworks.

Employing an iterative technique for the solution of (5.1), it is, of course, desirable that an algorithm produces iterates $u_{k+1}=u_k+s_k$ such that

$$f(u_{k+1}) < f(u_k) \tag{5.7}$$

holds. With regard to Algorithm 5.2 we obtain

$$ared_k(s_k) \geq \eta_1 \, pred_k(s_k) \tag{5.8}$$

if the step s_k in the k-th iteration is a successful step. Therefore, if it is possible to derive a lower bound for the predicted decrease, $pred_k(s_k)$, such that

$$pred_k(s_k) > 0 \tag{5.9}$$

holds, the desired relation (5.7) follows.

The predicted decrease based on a quadratic model function can be analyzed using the concept of *Cauchy decrease*, cf. Powell [89], Dennis [27], Conn et al. [23]. For this purpose, we consider the quadratic model for f in steepest descent direction

$$m_k^{\mathbf{quad}}(u_k - \lambda \nabla f(u_k)) = f(u_k) - \lambda \|\nabla f(u_k)\|^2 + \tfrac{1}{2}\lambda^2 \nabla f(u_k)^T H_k \nabla f(u_k), \tag{5.10}$$

for $\nabla f(u_k) \neq 0$ and $\lambda \geq 0$.

The minimization of (5.10) within the trust-region

$$\min_{\lambda \geq 0} \; m_k^{\mathbf{quad}}(u_k - \lambda \nabla f(u_k)), \quad s.t. \quad \|\lambda \nabla f(u_k)\| \leq \delta_k \tag{5.11}$$

yields the *Cauchy step* $s_k^C = -\lambda_k^C \nabla f(u_k)$ which is uniquely given by

$$\lambda_k^C = \begin{cases} \frac{\|\nabla f(u_k)\|^2}{\nabla f(u_k)^T H_k \nabla f(u_k)}, & \text{if } \frac{\|\nabla f(u_k)\|^3}{\nabla f(u_k)^T H_k \nabla f(u_k)} \leq \delta_k \text{ and } \nabla f(u_k)^T H_k \nabla f(u_k) > 0, \\ \frac{\delta_k}{\|\nabla f(u_k)\|}, & \text{otherwise,} \end{cases}$$

as simple convexity arguments show. Accordingly, $u_{k+1}^C = u_k + s_k^C$ denotes the *Cauchy point* of the trust-region subproblem (5.4).

Since s_k^C minimizes (5.10) within the trust-region, it is therefore natural to compare the predicted decrease, $pred_k(s_k)$, to the model decrease related to the Cauchy step. In fact, in view of (5.9) it suffices that the predicted decrease satisfies a *fraction of Cauchy decrease* condition

$$m_k^{\mathbf{quad}}(u_k) - m_k^{\mathbf{quad}}(u_k + s_k) \geq c_{fcd} \, [m_k^{\mathbf{quad}}(u_k) - m_k^{\mathbf{quad}}(u_k + s_k^C)] \tag{5.12}$$

for $c_{fcd} > 0$.

Powell [89], Th. 4, has shown that for the Cauchy decrease

$$m_k^{\mathbf{quad}}(u_k) - m_k^{\mathbf{quad}}(u_k + s_k^C) \geq \tfrac{1}{2} \|\nabla f(u_k)\| \min\{\delta_k, \frac{\|\nabla f(u_k)\|}{\|H_k\|}\} \tag{5.13}$$

holds. Therefore, we can derive the *sufficient decrease condition*

$$f(u_k) - f(u_k + s_k) \geq \tfrac{1}{2} \; \eta_1 \; c_{fcd} \; \|\nabla f(u_k)\| \min\{\delta_k, \frac{\|\nabla f(u_k)\|}{\|H_k\|}\}, \tag{5.14}$$

or equivalently

$$f(u_k + s_k) \leq f(u_k) - \tfrac{1}{2} \; \eta_1 \; c_{fcd} \; \|\nabla f(u_k)\| \min\{\delta_k, \frac{\|\nabla f(u_k)\|}{\|H_k\|}\}, \tag{5.15}$$

that is met whenever a trust-region iteration produces a successful step s_k that satisfies the fraction of Cauchy decrease condition (5.12).

Based on the sufficient decrease condition (5.14) it is possible to prove convergence of the trust-region algorithm with quadratic model functions given by (5.2).

Theorem 5.3 (Weak Global Convergence)
Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable and bounded below on the level set $\{u \in \mathbb{R}^n : f(u) \leq f(u_0)\}$. Let $\{u_k\}$ be a sequence of iterates generated by Alg. 5.2 with $\{H_k\}$ uniformly bounded and where the trial steps satisfy (5.12).

Then,

$$\liminf_{k \to \infty} \|\nabla f(u_k)\| = 0 \tag{5.16}$$

holds.

For a proof of (5.16) under the assumptions of Theorem 5.3, we refer, e.g., to Nocedal/Wright [82], cf. also Powell [89] or Moré [80].

We note that the weak assumptions on f do not allow to prove more than convergence of a subsequence to a stationary point. But, since there are no restrictions on the choice of u_0, Theorem 5.3 provides us with a global convergence result. By strengthening the assumptions on f we can also derive a stronger global convergence result.

Theorem 5.4 (Strong Global Convergence)
Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable with Lipschitz continuous gradient and bounded below on the level set $\{u \in \mathbb{R}^n : f(u) \leq f(u_0)\}$. Let $\{u_k\}$ be a sequence of iterates generated by Alg. 5.2 with $\{H_k\}$ uniformly bounded and where the trial steps satisfy (5.12).

Then,

$$\lim_{k \to \infty} \|\nabla f(u_k)\| = 0 \tag{5.17}$$

holds.

A proof for Theorem 5.4 can be found, e.g., in Nocedal/Wright [82], but we also refer to Moré [80] where (5.17) follows under slightly different assumptions.

5.3 Quadratic Model Functions with Inexact Gradient Information

The quadratic model functions in the previous section were built using exact gradient information, $\nabla f(u_k)$, and eventually approximate second-order information, H_k. But, in many practical applications where a trust-region method has to be applied in order to solve (5.1) the gradient of f at u_k is not given analytically, but can be approximated numerically.

This leads to trust-region methods with quadratic model functions

$$m_k^{\mathbf{iquad}}(u_k + s) = f(u_k) + g_k^T s + \tfrac{1}{2} s^T H_k s \tag{5.18}$$

where g_k denotes an approximation to the true gradient $\nabla f(u_k)$, i.e. $m_k^{\mathbf{iquad}}(u_k + s)$ is a model based on inexact gradient information. It is worth noting that in this case H_k denotes an approximation to the true Hessian, since it is unlikely that approximate gradient but exact Hessian information is employed to build (5.18).

We emphasize that the quadratic model function (5.18) with inexact gradient information possesses the same properties as the classical quadratic model function concerning, e.g., the efficient solution of the corresponding trust-region subproblem. Nevertheless, in the convergence analysis of a corresponding trust-region algorithm the error between exact and approximate gradient, $e_k = \nabla f(u_k) - g_k$, has to be taken into account. Several authors have addressed this issue and have derived different error conditions to achieve a global convergence theory.

The first error condition is due to Moré [80]. There, it is assumed that for a convergent series $\{u_k\}$ the approximate gradient, g_k, has to satisfy

$$\lim_{k\to\infty} \|\nabla f(u_k) - g_k\| = 0. \tag{5.19}$$

The *asymptotic consistency condition* (5.19) is also discussed in Borggaard/Burns [17, 18] for a sensitivity equation method for optimal design problems.

In Toint [111] the condition

$$\|\nabla f(u_k) - g_k\| \leq \min\{c_1, c_2\delta_k\} \quad \text{for all} \quad k \tag{5.20}$$

for given constants c_1, $c_2 > 0$ is imposed on the gradient accuracy. In order to avoid that the model $m_k^{\mathbf{iquad}}$ indicates a stationary point, i.e. $g_k = 0$, in cases of nonvanishing $\nabla f(u_k)$, in [111] an additional condition has to be imposed on (5.20). In this work, it is required that the following *critical point condition*

$$\|g_k\| = 0 \qquad \text{implies} \qquad \|\nabla f(u_k)\| = 0 \tag{5.21}$$

holds.

In Carter [20] the relative error condition

$$\frac{\|\nabla f(u_k) - g_k\|}{\|g_k\|} \leq \zeta \tag{5.22}$$

for given $\zeta \in (0, 1)$ is proposed to handle the gradient error, as long as $\|g_k\| \neq 0$. Conn et al. also use condition (5.22) for the treatment of approximate derivatives in trust-region algorithms in [23]. We note that according to Carter [21], condition (5.22) can also be substituted by

$$\frac{|(\nabla f(u_k) - g_k)^T g_k|}{\|g_k\|^2} \leq \zeta \tag{5.23}$$

for given $\zeta \in (0, 1)$.

Summing up one can say that the above error conditions illustrate the need for sufficiently accurate approximate gradients if $\|g_k\|$ approaches 0. Moreover, during the iteration process it is desirable to have some information on the necessary level of accuracy in the gradient computation.

Although condition (5.20) provides some information about an admissible error level on the computation of g_k at every iteration, since δ_k is known at the start of this computation, it has the disadvantage of having to ensure the above critical point condition (5.21).

The error condition (5.22) seems to be favourable if error estimates for the quantity $\|\nabla f(u_k) - g_k\|$ are available. The situation is similar for condition (5.23). In this case, estimates for the error in the directional derivative $|\nabla f(u_k)^T g_k - g_k^T g_k|$ can be used to evaluate (5.23). Both quantities allow to enforce the computation of sufficiently accurate gradients whereas conditions (5.19) and (5.20) mainly serve as theoretical tools.

Concerning a global convergence proof of a trust-region algorithm based on a quadratic model functions with inexact gradient information in terms of Theorem 5.3 and Theorem 5.4, an analogous sufficient decrease condition to (5.14) is required. Similar to the presentation in Section 5.2, the modified sufficient decrease condition

$$f(u_k) - f(u_k + s_k) \geq \tfrac{1}{2}\, \eta_1\, c_{fcd}\, \|g_k\| \min\{\delta_k, \frac{\|g_k\|}{\|H_k\|}\} \tag{5.24}$$

can be derived, see Moré [80]. Compared to the sufficient decrease condition (5.14) simply the norm of the true gradient, $\|\nabla f(u_k)\|$, has to be replaced by the norm of the approximate gradient, $\|g_k\|$. Again, condition (5.24) is based on a fraction of Cauchy decrease condition. In this case, the Cauchy step is given by $s_k^C = -\lambda_k^C g_k$.

Thus, based on the sufficient decrease condition (5.24) and an suitable error condition for the error between exact gradient, $\nabla f(u_k)$, and approximate gradient, g_k, we obtain the global convergence result $\lim_{k\to\infty} \|g_k\| = 0$.

At this place, we purposely renounce to state the proper assumptions for deriving this convergence result precisely, since we discuss and analyze this topic in the context of trust-region frameworks for reduced order modelling in the next chapter.

But, we also note that by employing the error condition (5.22), the convergence results for g_k carry over to $\nabla f(u_k)$, see Carter [20].

Lemma 5.5

Let $\{u_k\}$ be a sequence of iterates such that $\lim_{k\to\infty} \|g_k\| = 0$, where g_k and $\nabla f(u_k)$ satisfy the error condition (5.22) with $0 < \zeta < 1$ for all k.

Then

$$\lim_{k\to\infty} \|\nabla f(u_k)\| = 0$$

follows.

Proof: Using the error condition (5.22), the triangle inequality implies

$$\|\nabla f(u_k)\| - \|g_k\| \leq \zeta\, \|g_k\|. \tag{5.25}$$

Reformulating (5.25), we obtain

$$\|\nabla f(u_k)\| \leq (1 + \zeta)\, \|g_k\|$$

which proves the assertion. □

Remark 5.6

It is possible to consider a further generalization of the quadratic model function in (5.18) by also allowing inaccurately computed function values, f_k, that approximate $f(u_k)$, if the true function values are not available. As a matter of fact, a qualitative analysis of the corresponding trust-region algorithm requires error bounds on absolute/relative errors between $f(u_k)$, f_k and $\nabla f(u_k)$, g_k, cf. Carter [21], Kelley/Sachs [64], Conn et al. [23]. This issue is not further pursued here.

5.4 General Model Functions

Having introduced the basic ideas of trust-region methods with quadratic model functions, this section deals with trust-region methods where a priori no specific assumptions on the nonlinearity of the model function are made. Quadratic models (also with inexact gradients) are included as a special case.

Based on the work of Toint [111] and Carter [20] we are going to derive a sufficient decrease condition that later on can be used to prove convergence of a trust-region algorithm that incorporates POD based reduced order modelling approaches for optimal control problems. We note that the results in [111] are originally derived for constrained optimization problems with convex closed bounded feasible sets. Accordingly, these results are formulated for the unconstrained case, here.

We consider $f : \mathbb{R}^n \to \mathbb{R}$. For given $u_0 \in \mathbb{R}^n$, let U be an open convex set containing the level set of f at u_0 defined by $U_0 = \{ u \in \mathbb{R}^n \,|\, f(u) \leq f(u_0)\}$, $U_0 \subset U$. Throughout this section the following standard assumptions apply to f:

(F1) f is continuously differentiable on U,

(F2) f is bounded below, and

(F3) ∇f is Lipschitz continuous with constant L on U.

Furthermore, following Toint [111] and Carter [20], it is assumed that each model function, $m_k^{\mathbf{nonlin}}(u)$, at the current iterate u_k satisfies the conditions:

(M1) $m_k^{\mathbf{nonlin}}$ is differentiable on an open convex set containing the trust-region $\{ u \in \mathbb{R}^n \,|\, \|u - u_k\| \leq \delta_k \}$ for given trust-region radius $\delta_k > 0$,

(M2) $m_k^{\mathbf{nonlin}}(u_k) = f(u_k)$, and

(M3) $g_k = \nabla m_k^{\mathbf{nonlin}}(u_k)$ approximates $\nabla f(u_k)$ such that

$$\frac{\|\nabla f(u_k) - g_k\|}{\|g_k\|} \leq \zeta \tag{5.26}$$

for given $\zeta \in (0, 1)$.

A trust-region algorithm that employs general model functions that are not necessarily quadratic has the same structure as the basic trust-region algorithm (cf. Algorithm 5.2). In this case, we obtain the following trust-region subproblem.

Problem 5.7
For a given current iterate u_k and trust-region radius $\delta_k > 0$, determine an approximate solution s_k to

$$\min_{s \in \mathbb{R}^n} m_k^{\mathbf{nonlin}}(u_k + s), \quad s.t. \quad \|s\| \leq \delta_k.$$

As pointed out in the previous sections that dealt with quadratic model functions, the convergence behaviour of the trust-region scheme was based on the sufficient decrease conditions (5.14) or (5.24) and its linkage to the fraction of Cauchy decrease condition, cf. (5.12). In Section 5.2 we introduced the Cauchy step as the unique minimizer of the model function in steepest descent direction. Since we are treating more general model functions in this section, the Cauchy step s_k^C in the sense of (5.11) is no longer given.

Nevertheless, it is possible to derive an analogous sufficient decrease condition using a generalization of this concept. Obviously, in the case of general nonlinear model functions one can expect to get the desired sufficient decrease condition, if the predicted decrease for the trial step, $m_k^{\mathbf{nonlin}}(u_k) - m_k^{\mathbf{nonlin}}(u_k + s_k)$, can be related to the decrease in the model in steepest descent direction, $m_k^{\mathbf{nonlin}}(u_k) - m_k^{\mathbf{nonlin}}(u_k - \lambda g_k)$, for a certain step size $\lambda_k > 0$.

Toint [111] has proposed a step determination algorithm (SDA) that realizes the above ideas and that can be used for approximately solving the trust-region subproblem with general model functions. This SDA is outlined in Algorithm 5.8.

Algorithm 5.8 (Step Determination Algorithm (SDA))
Initialization: Choose

$$0 < \alpha < \beta < 1,\ 0 < \nu_1 < 1,\ \nu_2 > 0,\ \nu_3 > 0,\ 0 < \mu \leq 1. \tag{5.27}$$

Phase 1: Find λ_k^A such that

$$m_k^{\mathbf{nonlin}}(u_k - \lambda_k^A g_k) \leq m_k^{\mathbf{nonlin}}(u_k) - \alpha \lambda_k^A \|g_k\|^2 \tag{5.28}$$

$$\|\lambda_k^A g_k\| \leq \delta_k \tag{5.29}$$

and

$$\lambda_k^A \geq \min\{\nu_1 \frac{\delta_k}{\|g_k\|}, \nu_2\} \ \text{ or } \ \lambda_k^A \geq \lambda_k^B \tag{5.30}$$

where $\lambda_k^B > 0$ (if required) has to satisfy

$$m_k^{\mathbf{nonlin}}(u_k - \lambda_k^B g_k) \geq m_k^{\mathbf{nonlin}}(u_k) - \beta \lambda_k^B \|g_k\|^2 \tag{5.31}$$

Phase 2: If $\delta_k \geq \nu_3$: Choose step s_k such that

$$m_k^{\mathbf{nonlin}}(u_k) - m_k^{\mathbf{nonlin}}(u_k + s_k) \geq \mu \left[m_k^{\mathbf{nonlin}}(u_k) - m_k^{\mathbf{nonlin}}(u_k - \lambda_k^A g_k) \right] \tag{5.32}$$

$$\|s_k\| \leq \delta_k \tag{5.33}$$

Phase 1 of Algorithm 5.8 requires the determination of a step $s_k^{GC} = -\lambda_k^A g_k$ in the model's steepest descent direction that also satisfies the trust-region constraint (5.29). The step size λ_k^A has to be chosen such that a certain decrease in the model $m_k^{\mathbf{nonlin}}$ according

to (5.28) is ensured. Additionally, it has to be guaranteed that the step size λ_k^A is bounded away from zero. Therefore, either the first or the second part of condition (5.30) has to be satisfied. Essentially, Phase 1 can be interpreted as an inexact line search in steepest descent direction on the trust-region that yields a generalized Cauchy step s_k^{GC}.

In Phase 2, for a sufficiently large trust-region radius it is possible to leave this steepest descent direction. There, we can compute a step s_k that has to maintain a fraction of the generalized Cauchy decrease, see condition (5.32).

Algorithm 5.8 is well defined in the sense that it always provides a feasible step. For completeness we also review the proof of this result.

Lemma 5.9 (Toint [111], Lemma 5)
Provided that (5.27) holds, there always exists a step s_k satisfying conditions (5.28) - (5.33).

Proof: Without loss of generality we assume that $\|g_k\| > 0$ holds. Otherwise, the step $s_k = 0$ satisfies (5.28)–(5.33). Additionally, the choice $\mu \leq 1$ enables us to restrict the analysis of SDA to Phase 1. If $\lambda_k^A > 0$ is found that satisfies (5.28)–(5.31), then $s_k = -\lambda_k^A g_k$ also satisfies (5.32)–(5.33). For ease of notation, we drop the superscription and denote $m_k(u_k + s) = m_k^{\mathbf{nonlin}}(u_k + s)$ throughout this proof.

Let us assume that there exists a step size $\lambda_1 > 0$ such that

$$m_k(u_k - \lambda_1 g_k) = m_k(u_k) - \beta\lambda_1\|g_k\|^2. \tag{5.34}$$

Since $0 < \alpha < \beta$, equation (5.34) implies

$$m_k(u_k - \lambda_1 g_k) < m_k(u_k) - \alpha\lambda_1\|g_k\|^2. \tag{5.35}$$

In case we have $\lambda_1\|g_k\| < \delta_k$, we are able to choose $\varepsilon > 0$ such that also

$$(\lambda_1 + \varepsilon)\|g_k\| \leq \delta_k \tag{5.36}$$

holds. Additionally, we can rewrite equation (5.34) to obtain

$$\begin{aligned} m_k(u_k - (\lambda_1+\varepsilon)g_k) &= m_k(u_k) - \alpha(\lambda_1+\varepsilon)\|g_k\|^2 - (\beta-\alpha)\lambda_1\|g_k\|^2 + \alpha\varepsilon\|g_k\|^2 \\ &\quad + [m_k(u_k - (\lambda_1+\varepsilon)g_k) - m_k(u_k - \lambda_1 g_k)]. \end{aligned} \tag{5.37}$$

Furthermore, since $\beta > \alpha$, we also have

$$-(\beta-\alpha)\lambda_1\|g_k\|^2 < 0. \tag{5.38}$$

Consequently, the continuity of the model function and (5.38) allow us to choose ε in inequality (5.36) small enough to ensure

$$-(\beta-\alpha)\lambda_1\|g_k\|^2 + \alpha\varepsilon\|g_k\|^2 + [m_k(u_k - (\lambda_1+\varepsilon)g_k) - m_k(u_k - \lambda_1 g_k)] < 0. \tag{5.39}$$

Equation (5.37) together with (5.39) leads to

$$m_k(u_k - (\lambda_1+\varepsilon)g_k) \leq m_k(u_k) - \alpha(\lambda_1+\varepsilon)\|g_k\|^2. \tag{5.40}$$

Due to (5.35) and (5.40) we can choose $\lambda_k^A \in [\lambda_1, \lambda_1 + \varepsilon]$ and $\lambda_k^B = \lambda_1$, in order to get a step size that satisfies (5.28), (5.29), the second part of (5.30) and (5.31).

In case $\lambda_1\|g_k\| \geq \delta_k$ we assume that λ_1 is the smallest positive number such that (5.34) holds. Furthermore, we define the auxiliary function

$$h(\lambda) = m_k(u_k - \lambda g_k) - m_k(u_k) + \beta\lambda\|g_k\|^2.$$

According to (5.34), we have $h(0) = h(\lambda_1) = 0$ such that the continuity of the model function yields $h(\lambda) \leq 0$ for all $\lambda \in [0, \lambda_1]$. Consequently, $\alpha < \beta$ leads to

$$m_k(u_k - \lambda g_k) \leq m_k(u_k) - \alpha\lambda\|g_k\|^2 \tag{5.41}$$

for all $\lambda \in [0, \lambda_1]$. Defining $\lambda_2 = \nu_1\delta_k/\|g_k\|$ with $\nu_1 < 1$, we obtain a step size that satisfies $\lambda_2\|g_k\| = \nu_1\delta_k < \delta_k$ and $\lambda_2 < \lambda_1$. Therefore, we can conclude that there exists $\varepsilon > 0$ such that for all $\lambda_k^A \in [\lambda_2, \lambda_2 + \varepsilon]$ conditions (5.28), (5.29) and the first part of (5.30) are satisfied.

Finally, we have to consider the case that there exists no $\lambda_1 > 0$ such that equation (5.34) holds. In this case we choose λ_1 to be the smallest positive number that satisfies $\lambda_1\|g_k\| = \delta_k$. Since (5.41) holds for all $\lambda \in [0, \lambda_1]$, anyway, we can conclude in the same way as above to get the desired result of Lemma 5.9. □

Next, we state a result that describes the asymptotic behaviour of step directions, if the step determination algorithm SDA is applied for the solution of the trust-region subproblem given in Problem 5.7.

Lemma 5.10 (Cf. Carter [20], Th. 2.1)
Let $\{\delta_k\}$ be the sequence of trust-region radii, $\{g_k\}$ the sequence of model gradients at u_k and $\{s_k\}$ a sequence of steps computed by SDA. We define the angle, Θ_k, between s_k and $-g_k$ implicitly by

$$\cos\Theta_k = \frac{-s_k^T g_k}{\|s_k\|\,\|g_k\|}. \tag{5.42}$$

If $\liminf_{k\to\infty}\|g_k\| > 0$ and $\lim_{k\to\infty}\delta_k = 0$, then

$$\lim_{k\to\infty}\cos\Theta_k = 1 \tag{5.43}$$

follows.

Proof: Let $\|g_k\| > \varepsilon$ for sufficiently large k and for some $\varepsilon > 0$. If $\delta_k \to 0$, $k \to \infty$, there exists an iteration number k_0 such that for all following iterations $k \geq k_0$ the resulting

step s_k is determined according to Phase 1 in SDA, i.e. $s_k = -\lambda_k^A g_k$. In this case we have $\cos\Theta_k = (\lambda_k^A g_k^T g_k)/(\lambda_k^A \|g_k\|^2) = 1$ and the desired result follows. □

Having introduced the step determination algorithm as a solution method for the trust-region subproblem with general, nonlinear model functions according to [111], we now turn to the required sufficient decrease condition.

We recall that for trust-region methods with quadratic model functions based on inexact gradients the sufficient decrease condition reads

$$f(u_k) - f(u_k + s_k) \geq \frac{1}{2} c \, \|g_k\| \min\{\delta_k, \frac{\|g_k\|}{\|H_k\|}\}$$

for given $c > 0$, where $\|H_k\|$ denotes the norm of the Hessian of the model function. Thus, it represents a measure for the model's curvature at u_k. In the general, nonlinear model function case a similar curvature concept is necessary, see [111].

Definition 5.11 (Curvature Measure)
Let f satisfy **(F1)**. We define a measure for the curvature of f along the step s, $s \neq 0$, based at a point u by

$$\omega(f, u, s) = \frac{2}{\|s\|^2}(f(u+s) - f(u) - \nabla f(u)^T s). \tag{5.44}$$

We note that Definition 5.11 implies $|\omega(m_k^{\mathbf{quad}}, u_k, s)| = |s^T H_k s/(s^T s)| \leq \|H_k\|_2$ for a quadratic model function given by (5.2) with symmetric Hessian H_k.

Provided with the curvature measure (5.44) we can prove the following result, that gives a lower bound for the predicted decrease in the model function, $m_k^{\mathbf{nonlin}}$, if the step determination algorithm (Alg. 5.8) is used.

Theorem 5.12 (Toint [111], Th. 7)
Assume that $\|g_k\| \neq 0$ and define according to SDA and (5.44)

$$\omega_k := \begin{cases} 0 & \text{if } \lambda_k^B \text{ is undefined,} \\ \omega(m_k^{\mathbf{nonlin}}, u_k, -\lambda_k^B g_k) & \text{if } \lambda_k^B \text{ is defined.} \end{cases} \tag{5.45}$$

Then

$$\omega_k \geq 0 \tag{5.46}$$

and there exists a constant $c_s > 0$ such that SDA produces a step s_k with

$$m_k^{\mathbf{nonlin}}(u_k) - m_k^{\mathbf{nonlin}}(u_k + s_k) \geq c_s \|g_k\|^2 \min\{\|g_k\|^2/(1+\omega_k), \delta_k\}. \tag{5.47}$$

Proof: Based on the standard assumptions on f there exists a constant $c_f > 0$ such that $\|\nabla f(u_k)\| \leq c_f$. Moreover, error condition (5.26) implies that we can define a constant $c_g = c_f/(1-\zeta) > 0$ such that

$$\|g_k\| \leq \frac{1}{1-\zeta}\|\nabla f(u_k)\| \leq c_g. \tag{5.48}$$

Again, we drop the superscription and denote $m_k(u_k+s) = m_k^{\mathbf{nonlin}}(u_k+s)$ throughout this proof.

According to (5.45) let us assume that λ_k^B is defined. With (5.31), (5.44) and $\beta<1$ we obtain

$$\begin{aligned}\omega(m_k, u_k, -\lambda_k^B g_k) &= \frac{2}{\|\lambda_k^B g_k\|^2}(m_k(u_k-\lambda_k^B g_k) - m_k(u_k) + \lambda_k^B \|g_k\|^2) \\ &\geq \frac{2}{\|\lambda_k^B g_k\|^2}(1-\beta)\lambda_k^B \|g_k\|^2 \\ &\geq \frac{2}{\lambda_k^B c_g^2}(1-\beta)\|g_k\|^2 \ > 0. \end{aligned} \tag{5.49}$$

Thus, the definition of ω_k in (5.45) and (5.49) prove the assertion in (5.46).

For further use we reformulate (5.49) as

$$\lambda_k^B \geq \frac{2(1-\beta)\|g_k\|^2}{\omega(m_k, u_k, -\lambda_k^B g_k)\, c_g^2}. \tag{5.50}$$

We also recall that due to (5.28) and (5.32) SDA produces steps, s_k, that satisfy

$$m_k(u_k) - m_k(u_k+s_k) \geq \alpha\mu\lambda_k^A \|g_k\|^2. \tag{5.51}$$

In case λ_k^A satisfies the first part of condition (5.30), then (5.48) and (5.51) can be used to show

$$\begin{aligned} m_k(u_k) - m_k(u_k+s_k) &\geq \alpha\mu\|g_k\|^2 \min\{\nu_1 \frac{\delta_k}{\|g_k\|}, \nu_2\} \\ &\geq \alpha\mu\|g_k\|^2 \min\{\nu_1 \frac{\delta_k}{c_g}, \nu_2\}. \end{aligned} \tag{5.52}$$

Therefore, we can find a constant $c_A > 0$ such that

$$m_k(u_k) - m_k(u_k+s_k) \geq c_A \|g_k\|^2 \delta_k. \tag{5.53}$$

Let us consider the case where λ_k^A is chosen according to the second part of condition (5.30). Then, (5.51) together with the lower bound for λ_k^B given in (5.50) and (5.45) yield

$$\begin{aligned} m_k(u_k) - m_k(u_k+s_k) &\geq \alpha\mu\lambda_k^B\|g_k\|^2 \\ &\geq \alpha\mu\|g_k\|^2 \frac{2(1-\beta)\|g_k\|^2}{\omega(m_k, u_k, -\lambda_k^B g_k)c_g^2} \\ &= c_B \|g_k\|^2 \frac{\|g_k\|^2}{\omega(m_k, u_k, -\lambda_k^B g_k)} \\ &\geq c_B \|g_k\|^2 \frac{\|g_k\|^2}{\omega_k + 1} \end{aligned} \tag{5.54}$$

where the constant $c_B > 0$ is defined by $c_B = 2\alpha\mu(1-\beta)/c_g^2$. Finally, with (5.53), (5.54) and $c_s = \min\{c_A, c_B\}$ the assertion in (5.47) follows. □

Consequently, we obtain the following sufficient decrease condition, that describes the decrease in the true objective function when the iteration is successful.

Corollary 5.13
Let the assumptions of Theorem 5.12 be satisfied and assume that the k-th iteration was successful.

Then

$$f(u_k) - f(u_k + s_k) \geq \eta_1 c_s \|g_k\|^2 \min\{\frac{\|g_k\|^2}{1+\omega_k}, \delta_k\} \tag{5.55}$$

holds.

Proof: Since for successful iterations $ared_k(s_k) \geq \eta_1 pred_k(s_k)$ follows, we obtain the desired sufficient decrease condition for f using Theorem 5.12. □

We note that the sufficient decrease condition (5.55) for general, nonlinear model functions is of the same type as the sufficient decrease condition (5.24) in the quadratic model function case with inexact gradients. In both cases, the lower bound for the actual reduction $ared_k(s_k) = f(u_k) - f(u_k + s_k)$ depends on the norm of the model gradient, $\|g_k\|$, and on the model curvature.

So far, with Lemma 5.10 and Corollary 5.13 we provided two important results for a convergence proof of the main algorithm that combines the trust-region technique and POD based reduced order modelling in the next chapter.

Chapter 6

Trust-region Frameworks for POD based Reduced Order Modelling

6.1 Introduction

In the previous chapter, the trust-region approach has been considered from a primarily theoretical point of view. We provided a general framework that admits to make use of approximations to the original objective function of the underlying optimization problem (5.1) based on a variety of possible model functions, where the assumptions on the original objective, f, and model functions, m_k, are fairly weak.

Beyond this, the trust-region framework supplies an efficient tool for the solution of optimization problems of the type

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

where $\hat{f}(y, u)$ is a function with $\hat{f} : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}$, which often arise in all areas of practical applications. In problem (6.1), $u \in \mathbb{R}^n$ denotes decision variables and the vector-valued function $y(u)$ from $\mathbb{R}^n$ to $\mathbb{R}^m$ denotes state variables (or system variables) that describe a certain input/output-relation: $y(u)$ is the system output for a given input u.

In many cases, the input/output-relation can be described by a known model, e.g., a finite element model of a partial differential equation that models some physical processes. Thus, the evaluation of

$$f(u) = \hat{f}(y(u), u) \tag{6.2}$$

during the solution of (6.1) is computationally expensive, if an accurate computation of $y(u)$, i.e. the solution of the state equation, for given u is computationally expensive. In such cases it is desirable to replace the objective function, f, by a model function that can be evaluated at less computational expenses. This can be achieved by employing an

approximation model for the state equation such that

$$\hat{f}(y(u),u) \text{ is replaced by } \hat{f}(y^{\mathbf{mod}}(u),u) \tag{6.3}$$

during the optimization process, where $y^{\mathbf{mod}}(u)$ denotes a corresponding approximate state equation solution for given u. Consequently, with regard to the trust-region methods presented in Chapter 5,

$$m_k(u_k+s)=\hat{f}(y^{\mathbf{mod}}(u_k+s),u_k+s) \tag{6.4}$$

provides a model function for

$$f(u_k+s)=\hat{f}(y(u_k+s),u_k+s) \tag{6.5}$$

at a given current iterate u_k. Thus, since we made no assumptions on the 'nonlinearity' of $\hat{f}$ as well as on the nonlinearity of the mapping $u \mapsto y(u)$, we obtain a general model function, m_k, that is not necessarily a quadratic function.

In general, the approximation model that delivers $y^{\mathbf{mod}}(u)$ for given u is characterized by lower accuracy (*low-fidelity model*), but less computational expense, compared to the original model for the computation of $y(u)$ (*high-fidelity model*), cf. Alexandrov et al. [4], Rodriguez et al. [96]. In Conn et al. [23] the term *composite model* can be found for the type of model function given in (6.4). Another popular term for using approximation models for the state equations during the solution of (6.1) is *surrogate optimization*, see e.g. Booker et al. [16].

In many engineering applications the use of approximation models is common practice. But, if approximate modelling is used as a stand-alone technique, it is difficult to make a statement about a problem's solution quality based on approximation models with regard to the true solution of the problem. Even in cases where a surrogate approach, i.e. solving the optimization problem (6.1) with the substitution suggested in (6.3), yields a converged solution, it is necessary to relate this solution to the true solution that one intends to compute. According to the material presented in Chapter 5, this can be accomplished by employing a trust-region framework, also known as *model management* or *surrogate management framework*. This is a well-known technique in *multidisciplinary design optimization* (MDO). For a survey of applications of *variable-fidelity models* in MDO we refer to Alexandrov/Hussaini [5].

In the MDO literature, two approaches for the solution of trust-region subproblems with general model functions based on approximation models for the state equation in the sense of (6.4), (6.5) can be found.

If problem (6.1) is characterized by the fact that derivatives for f are not available or cannot be computed except at an undesirable amount of computational cost, then, derivative free methods are of great interest. In, e.g., Dennis/Torczon [28], Serafini [99] or

Booker et al. [16], the trust-region approach is combined with the use of approximation models according to (6.3), but without using derivative information during the solution of the corresponding trust-region subproblems. In those cases, *direct search methods* that merely use function values of f and that do not explicitly use gradient information can be employed (see Kelley [63]). In the context of model management strategies especially *pattern search methods* (as a subclass of direct search algorithms) have been applied for the computation of putative iterates [28, 99, 16]. Pattern search methods are characterized by operating only at a set of discrete points. A *pattern* consists of an ensemble of potential steps such that the current iterate plus each step defines a possible trial iterate. During the optimization process the patterns and an additional *scale factor* (or *step length parameter*) are adapted to the current iterate. At each iteration, the next iterate u_{k+1} is chosen using the pattern such that a *simple decrease condition*, $f(u_{k+1}) < f(u_k), \quad \text{if} \quad u_{k+1} \neq u_k$, is satisfied. The convergence results for model management frameworks that incorporate pattern search algorithms are similar to Theorem 5.3 (cf. Serafini [99]) and reflect the convergence properties of pattern search methods itself (see Torczon [112], Lewis/Torczon [75]).

If derivative information for (6.1) is available, we are led to the second class of surrogate optimization approaches. Similar to the trust-region approach presented in Section 5.4, the research results of Alexandrov et al. [4, 6] are based on general model functions. In these works, a *first-order consistency condition* is assumed, i.e. the model functions have to satisfy

$$m_k(u_k) = f(u_k) \quad \text{and} \quad \nabla m_k(u_k) = \nabla f(u_k) \tag{6.6}$$

for all iterates u_k. This consistency condition can be enforced by a scaling technique that will be introduced in Section 6.3. In the trust-region frameworks presented in [4, 6], again, a trust-region algorithm based on quadratic model functions is used for the solution of the corresponding trust-region subproblems. Thus, this leads to solving trust-region subproblems with an additional norm constraint on an admissible step which have been analyzed in Heinkenschloss [48].

Having introduced the proper orthogonal decomposition for the Navier–Stokes equations in Chapter 3, a reduced order modelling approach for flow control based on POD obviously provides a specific example for deriving an approximation model for the state equation. Furthermore, to our knowledge, POD based reduced order modelling for optimization purposes has been used as a stand-alone technique so far. It is the goal of this chapter to illustrate how the ideas of surrogate optimization and POD based reduced order modelling can be combined. In so doing, a trust-region framework for implementing an optimal control approach based on reduced order modelling can be achieved. Here, trust-region methods with general nonlinear model functions as presented in Section 5.4 provide the necessary theoretical background for an analysis of this approach.

In Section 6.2 we present an algorithmic framework that implements the combination

of POD based reduced order modelling and trust-region methods, the TRPOD algorithm, cf. [8]. Based on the theoretical results for trust-region methods with general model functions presented in Chapter 5, we are able to prove the convergence of the TRPOD algorithm. We also discuss some additional modifications of the basic TRPOD algorithm that can lead to an improved performance.

Since the convergence results are strongly influenced by the gradient accuracy of the model functions during the optimization process, we therefore discuss this topic in Section 6.3 leading to the proposal of a modified algorithm that incorporates gradient information of the true objective function, f. In this context, we introduce a scaling technique that provides model functions satisfying the first-order consistency condition according to (6.6).

Finally, we devote the last section of this chapter to the presentation of numerical results that have been obtained by applying the proposed algorithms to optimal control problems in fluid flows and heating processes.

6.2 The Trust-region Proper Orthogonal Decomposition Approach

6.2.1 The TRPOD Algorithm

In Chapter 3 we gave a detailed description of the POD based reduced order modelling technique based on an input ensemble of snapshots. Since these reduced order models are based on the solution of the underlying partial differential equation, they are especially suited for representing this input ensemble. In Subsection 3.6 we illustrated this behaviour with some numerical results.

For example, if a POD based reduced order model has been set up that reflects a certain control influence on the system, the same reduced order model might deliver no reliable information on the system behaviour under a completely different control influence. Consequently, such reduced order models might be poor models when the controller takes the system from its current state towards the optimal state in cases where (unknown) intermediate dynamics cannot be caught by the reduced order model. This is due to the limited number of degrees of freedom in the reduced order model as prescribed by the POD basis elements and constitutes its main weakness for optimal control purposes. Therefore, there is no guarantee that an optimization approach with POD based reduced order models will converge to a solution of the original system without any additional mechanism.

In order to create a connection between the original optimal control problem and the reduced order modelling approach we therefore propose a Trust-Region Proper Orthogonal Decomposition (TRPOD) method in the spirit of surrogate optimization to overcome those difficulties. By embedding the POD technique into the concept of trust-region frameworks

with general, nonquadratic model functions and inexact gradient information it is possible to provide a mechanism to decide when an update of the POD based control model is necessary during the optimization process.

In the following, we describe the ideas of the trust-region proper orthogonal decomposition approach for flow control. For this purpose we consider the control problem given in Problem 2.6 as an unconstrained optimization problem

$$\min_{u \in \mathbb{R}^n} \quad \hat{f}(y^N(u), u) \tag{6.7}$$

where $u \in \mathbb{R}^n$ denotes the discretized control due to a time discretization and $y^N(u)$ denotes a finite element approximation to the weak solution of (2.1), (2.2), (2.3) and (2.5) for a given control. We emphasize that due to Theorem 2.7 for an appropriate problem formulation, there always exists a unique weak solution to the Navier–Stokes equations for a given control.

Since we are interested in applying a reduced order model based on POD for the computation of an approximate solution to $y^N(u)$, according to the above discussion, we have to be aware of possibly unreliable reduced order solutions. It is therefore natural to adapt the POD based control model successively during the course of the optimization by adapting the snapshot set that is used to generate the POD basis. The corresponding TRPOD algorithm is given in Algorithm 6.1, where a stopping criterion should be added in practice, cf. Algorithm 5.2.

Given a current trust-region radius, δ_k, and an iterate for the control, u_k, we first compute snapshots that correspond to the flow dynamics forced by u_k by solving a finite element system for the Navier–Stokes equations. These snapshots form the input ensemble $\mathcal{Y}_k$. Based on the finite element solution, we can also evaluate the objective function $f(u_k) = \hat{f}(y^N(u_k), u_k)$ at the current iterate.

Furthermore, we use the snapshot ensemble, $\mathcal{Y}_k$, to compute a POD basis, $\mathcal{Y}_k^{POD}$. Then, based on the techniques presented in Section 3.5.2, we derive a POD based control model at u_k. This reduced order model can be employed for an optimization cycle.

For this purpose, according to (6.4), (6.5) we define

$$m_k(u_k + s) = \hat{f}(y^M(u_k + s), u_k + s) \tag{6.8}$$

as a model function for

$$f(u_k + s) = \hat{f}(y^N(u_k + s), u_k + s). \tag{6.9}$$

on the trust-region $\|s\| \leq \delta_k$ around u_k. In the definition of the model function (6.8), $y^M(u_k + s)$ denotes the POD based reduced order solution at $u_k + s$ computed with the reduced order model at u_k. We recall that the superscript M indicates the number of POD basis functions that have been used to build this reduced order model.

Algorithm 6.1 (Basic TRPOD Algorithm)
Initialization: Choose $0 < \eta_1 < \eta_2 < 1$, $0 < \gamma_1 \leq \gamma_2 < 1 \leq \gamma_3$, an initial trust-region radius $\delta_0 > 0$ and an initial iterate u_0. Compute a snapshot set $\mathcal{Y}_0$ corresponding to control u_0 and compute $f(u_0)$ according to (6.9). Set $k = 0$.

1. Compute a POD basis $\mathcal{Y}_k^{POD}$ and build a POD based control model.

2. Compute an approximate minimizer, s_k, of the model function $m_k(u_k + s)$ given by (6.8) within the trust-region, $\|s\| \leq \delta_k$, using SDA.

3. Compute a snapshot set $\mathcal{Y}_{k+}$ corresponding to control $u_k + s_k$ and compute $f(u_k + s_k)$ according to (6.9). Set

$$\rho_k = \frac{ared_k(s_k)}{pred_k(s_k)}.$$

4. Update the trust-region radius:

- If $\rho_k \geq \eta_2$:
  Set $u_{k+1} = u_k + s_k$, $f(u_{k+1}) = f(u_k + s_k)$, $\mathcal{Y}_{k+1} = \mathcal{Y}_{k+}$ and choose $\delta_{k+1} \in [\delta_k, \gamma_3 \delta_k]$.
  Set $k = k + 1$ and GOTO 1.

- If $\eta_1 \leq \rho_k < \eta_2$:
  Set $u_{k+1} = u_k + s_k$, $f(u_{k+1}) = f(u_k + s_k)$, $\mathcal{Y}_{k+1} = \mathcal{Y}_{k+}$ and choose $\delta_{k+1} \in [\gamma_2 \delta_k, \delta_k)$.
  Set $k = k + 1$ and GOTO 1.

- If $\rho_k < \eta_1$:
  Set $u_{k+1} = u_k$ and choose $\delta_{k+1} \in [\gamma_1 \delta_k, \gamma_2 \delta_k]$. Set $k = k + 1$ and GOTO 2.

An exact solution of the corresponding trust-region subproblem

$$\min_{s \in \mathbb{R}^n} m_k(u_k + s), \quad s.t. \quad \|s\| \leq \delta_k,$$

would deliver a solution that is optimal for the current surrogate problem. But, following the trust-region philosophy it is sufficient to compute a trial step that achieves a certain amount of decrease for the model function. Computing such a trial step via the step determination algorithm (SDA) presented in Alg. 5.8, we denote this trial step by s_k.

In order to estimate the quality of the putative next iterate $u_{k+1} = u_k + s_k$ with regard to the optimization goal, we compare the actual reduction in the true objective, $ared_k(s_k) = f(u_k) - f(u_k + s_k)$, to the predicted reduction, $pred_k(s_k) = m_k(u_k) - m_k(u_k + s_k)$. This requires to compute a new finite element solution, $y^N(u_k + s_k)$, in order to evaluate $f(u_k + s_k)$. If the trial step, s_k yields a satisfactory decrease in the original objective, we accept this step, and because of the last finite element solution, a new snapshot ensemble $\mathcal{Y}_{k+1}$ is available.

Consequently, this new snapshot ensemble is adapted to flow dynamics as altered by the control and the next POD based reduced order model incorporates these new dynamics.

6.2.2 Convergence Results

Before we turn to the convergence analysis of the TRPOD algorithm, we again emphasize that, in general, the model function m_k given by (6.8) is a nonquadratic model function for f, since it depends on u_k+s through the solution of the reduced order state equation. Moreover, we made no assumptions on the nonlinearity of f.

According to Chapter 5, we denote the gradient of the model function m_k at the center point of the trust-region by $g_k = \nabla m_k(u_k)$. Obviously, g_k is an approximation to the gradient, $\nabla f(u_k)$, of the actual objective function since we compute this quantity on reduced order information only. Consequently, a condition on the gradient error between g_k and $\nabla f(u_k)$ is required. Based on the discussion in Section 5.3 we choose the error condition (5.22) proposed in Carter [20].

Beside this, the trust-region proper orthogonal decomposition approach also includes inexact model function values at the trust-region center point. Due to a POD approximation error in the reduced order model, we have $m_k(u_k) \neq f(u_k)$ which relaxes assumption **(M2)** on the model function, see Section 5.4. Nevertheless, the results of Section 5.4 still hold, since we considered questions particularly related to the model decrease whereas the sufficient decrease condition (5.55) assumes that we meet a successful iteration.

We will now state under which assumptions Algorithm 6.1 will always find an acceptable step. For the definition of various parameters that occur in the step determination algorithm (Algorithm 5.8) and that are used in the statement of the following theorem and its proof, we refer to Section 5.4.

Theorem 6.2
Let f satisfy the standard assumptions **(F1)**–**(F3)** *and consider a fixed iterate u_k such that m_k satisfies* **(M1)**. *Let g_k denote a gradient approximation to $\nabla f(u_k)$, where we assume that $g_k \neq 0$ and that the gradient error condition (5.26) in* **(M3)** *is satisfied for $\zeta \in (0, 1-\eta_1)$. Furthermore, we suppose that Alg. 5.8 is used to compute a trial step s_k, where we also assume that $|\omega(m_k, u_k, s_k)| \leq c_\omega$ holds, for some constant $c_\omega > 0$.*

Then, for sufficiently small δ_k we have $\rho_k > \eta_1$, such that iteration k is successful.

Proof: We refer to Carter [20] for a similar proof for quadratic model functions.

Since $\|g_k\| \neq 0$ holds, Theorem 5.12 guarantees that Alg. 5.8 produces steps s_k satisfying $pred_k(s_k) > 0$. With the definition of the curvature measure, $\omega(\cdot)$, given in (5.44) and $\cos\Theta_k = -s_k^T g_k/(\|s_k\|\|g_k\|)$ defined in (5.42) we have

$$1 - \rho_k \quad = \quad 1 - \frac{ared_k(s_k)}{pred_k(s_k)}$$

$$
\begin{aligned}
&= \frac{m_k(u_k) - m_k(u_k + s_k) - (f(u_k) - f(u_k + s_k))}{m_k(u_k) - m_k(u_k + s_k)} \\
&\leq \frac{|(\nabla f(u_k) - g_k)^T s_k| + \frac{1}{2}\|s_k\|^2 |\omega(f, u_k, s_k) - \omega(m_k, u_k, s_k)|}{-g_k^T s_k - \frac{1}{2}\|s_k\|^2 \omega(m_k, u_k, s_k)} \qquad (6.10) \\
&\leq \frac{\|\nabla f(u_k) - g_k\| \, \|s_k\| + \frac{1}{2}\|s_k\|^2 (|\omega(f, u_k, s_k)| + |\omega(m_k, u_k, s_k)|)}{\|g_k\| \, \|s_k\| \cos\Theta_k - \frac{1}{2}\|s_k\|^2 \omega(m_k, u_k, s_k)} \\
&= \frac{\frac{\|\nabla f(u_k) - g_k\|}{\|g_k\|} + \frac{\|s_k\|}{2\|g_k\|}(|\omega(f, u_k, s_k)| + |\omega(m_k, u_k, s_k)|)}{\cos\Theta_k - \frac{\|s_k\|}{2\|g_k\|}\omega(m_k, u_k, s_k)}. \qquad (6.11)
\end{aligned}
$$

Furthermore, the assumptions on f imply

$$
\begin{aligned}
|\omega(f, u_k, s_k)| &= \frac{2}{\|s_k\|^2} |f(u_k + s_k) - f(u_k) - \nabla f(u_k)^T s_k| \\
&= \frac{2}{\|s_k\|^2} \left| \int_0^1 \nabla f(u_k + \sigma s_k)^T s_k \, d\sigma - \nabla f(u_k)^T s_k \right| \\
&\leq \frac{2}{\|s_k\|^2} \int_0^1 \|\nabla f(u_k + \sigma s_k) - \nabla f(u_k)\| \, \|s_k\| d\sigma \;\leq\; L \qquad (6.12)
\end{aligned}
$$

where L is the Lipschitz constant of ∇f.

According to Alg. 5.8, we obtain $\cos\Theta_k = 1$, if $\delta_k < \nu_3$ for given $\nu_3 > 0$ (see Lemma 5.10). Thus, we assume $\delta_k < \nu_3$ for the rest of this proof.

Inserting (6.12), $|\omega(m_k, u_k, s_k)| \leq c_\omega$ and $\|\nabla f(u_k) - g_k\|/\|g_k\| \leq \zeta$ in (6.11) yields

$$
1 - \rho_k \leq \frac{\zeta + \frac{\delta_k}{2\|g_k\|}(L + c_\omega)}{1 - \frac{\|s_k\|}{2\|g_k\|}\omega(m_k, u_k, s_k)}. \qquad (6.13)
$$

Consequently, since $\|g_k\| > 0$, we have

$$
\lim_{\delta_k \to 0} \; \tfrac{1}{2} \, \delta_k (L + c_\omega)/\|g_k\| = 0 \qquad (6.14)
$$

and

$$
0 \leq \lim_{\delta_k \to 0} \; \tfrac{1}{2} \, \|s_k\| \, |\omega(m_k, u_k, s_k)|/\|g_k\| \leq \lim_{\delta_k \to 0} \; \tfrac{1}{2} \, \delta_k c_\omega/\|g_k\| = 0. \qquad (6.15)
$$

Thus, with (6.14), (6.15) follows

$$
\lim_{\delta_k \to 0} (1 - \rho_k) \leq \zeta < 1 - \eta_1 \qquad (6.16)
$$

proving the existence of a sufficiently small δ_k, that leads to the desired result $\rho_k > \eta_1$. □

Theorem 6.2 leads to the first convergence result for the TRPOD algorithm which is similar to Theorem 5.3, cf. Toint [111], Lemma 9 and Theorem 10, and Carter [20], Theorem 3.3.

Theorem 6.3 (Weak Global Convergence of the TRPOD Algorithm)
Let f satisfy the standard assumptions **(F1)**-**(F3)**. *Assume that $\{u_k\}$ is a sequence of iterates generated by Alg. 6.1 with step determination according to Alg. 5.8. Let $\{s_k\}$ be the sequence of steps. Furthermore, we assume that each model function m_k satisfies* **(M1)** *and* **(M3)** *for some $\zeta \in (0, 1-\eta_2)$. We define*

$$b_k = 1 + \max\{\max\{\omega_i, |\omega(m_i, u_i, s_i)|\}, i = 0, \ldots, k\} \tag{6.17}$$

according to (5.44), (5.45) and assume that there exists some constant $c_b > 0$ such that

$$b_k \leq c_b \quad \textit{for all} \quad k. \tag{6.18}$$

Then

$$\liminf_{k\to\infty} \|g_k\| = 0$$

follows.

Proof: The proof follows by contradiction where we assume that

$$\liminf_{k\to\infty} \|g_k\| \geq \varepsilon \tag{6.19}$$

for some $\varepsilon > 0$.

The first intermediate result can be derived using standard arguments and is based on the sufficient decrease condition for general model functions, see Cor. 5.13. Using (6.18), at each successful iteration for the actual reduction holds

$$f(u_k) - f(u_k + s_k) \geq \eta_1 c_s \|g_k\|^2 \min\{\|g_k\|^2/c_b, \delta_k\} \tag{6.20}$$

for $c_s > 0$ and $0 < \eta_1 < 1$. At this point, we remark that Theorem 6.2 and the relation $0 < \zeta < 1 - \eta_2 < 1 - \eta_1$ guarantee that such successful iterations exist. Since f is bounded below, (6.19), (6.20) yield $\sum_{k\in S} \delta_k < \infty$ where S denotes the set of indices of successful iterations. But, since the trust-region radius is reduced for unsuccessful iterations anyway we get $\sum_{k=0}^{\infty} \delta_k < \infty$ and thus $\delta_k \to 0$ for $k \to \infty$, cf. Moré [80].

Without loss of generality we assume that k is sufficiently large such that $\|g_k\| \geq \varepsilon$ holds. According to (6.11) and by combining (6.12), (6.17) and $\|s_k\| \leq \delta_k$ we obtain

$$1 - \rho_k \leq \frac{\|\nabla f(u_k) - g_k\| + \frac{1}{2}\delta_k(L + b_k)}{\|g_k\| \cos\Theta_k - \frac{1}{2}\|s_k\|\,\omega(m_k, u_k, s_k)}. \tag{6.21}$$

Similar to (6.16) we can now use (5.43), (M3), (6.18) and $\delta_k \to 0$, $k \to \infty$ to deduce that

$$\lim_{k\to\infty} (1 - \rho_k) \leq \zeta < 1 - \eta_2. \tag{6.22}$$

The result in (6.22) implies that there exists an iteration index k_0 such that $\rho_k > \eta_2$ for all $k \geq k_0$. Thus, the trust-region radius δ_k is not reduced for all $k \geq k_0$ which leads to a contradiction with $\delta_k \to 0$ for $k \to \infty$. □

Moreover, we can also prove strong global convergence of the TRPOD algorithm in the sense of Theorem 5.4, cf. Carter [20] and Conn et al. [23]. Again, the specific choice of the error condition (5.26) plays an important role in the proof of this result.

Theorem 6.4 (Strong Global Convergence of the TRPOD Algorithm)
Let the assumptions of Theorem 6.3 be satisfied.

Then,

$$\lim_{k\to\infty} \|g_k\| = 0$$

follows.

Proof: We refer to Carter [20] for a similar proof for quadratic model functions, also cf. Moré [80] and Nocedal/Wright [82].

Let us consider any iteration index l such that $\|g_l\| \neq 0$ and, for technical reasons, we set

$$\varepsilon = \tfrac{1}{2}(1-\zeta)/(1+\zeta) \tag{6.23}$$

such that the choice $\zeta \in (0, 1-\eta_2)$ implies $\varepsilon \in (0,1)$. Theorem 6.3 guarantees that for any $\varepsilon \in (0,1)$ there exists a second iteration index $m \geq l$ such that

$$\|g_{m+1}\| \leq \varepsilon \|g_l\| \tag{6.24}$$

and

$$\|g_k\| > \varepsilon \|g_l\| \tag{6.25}$$

for all $k = l, \ldots, m$.

For the objective function values of successive iterates, we have

$$\begin{aligned} f(u_l) - f(u_{m+1}) &= \sum_{k=l}^{m} f(u_k) - f(u_{k+1}) \\ &= \sum_{\substack{k=l \\ k\in S}}^{m} ared_k(s_k) \end{aligned} \tag{6.26}$$

where we restricted the sum to the indices of successful iterations, S, in $\{l, \ldots, m\}$ such that $u_{k+1} \neq u_k$. We remark that due to (6.24) the index set $S \cap \{l, \ldots, m\}$ is nonempty. Since for any successful iteration the sufficient decrease condition (5.55) holds, (6.26) leads to

$$f(u_l) - f(u_{m+1}) \geq \sum_{\substack{k=l \\ k\in S}}^{m} \eta_1 c_s \|g_k\|^2 \min\{\frac{\|g_k\|^2}{c_b}, \delta_k\} \tag{6.27}$$

$$\geq \sum_{\substack{k=l \\ k\in S}}^{m} \eta_1 c_s \varepsilon^2 \|g_l\|^2 \min\{\frac{\varepsilon^2\|g_l\|^2}{c_b}, \|s_k\|\} \tag{6.28}$$

where we also employed the curvature bound (6.18) to obtain (6.27), and (6.25) and $\|s_k\| \leq \delta_k$ to obtain (6.28).

Furthermore, inequality (6.24) leads to

$$\begin{aligned} \|g_l\| &\leq \|g_l - g_{m+1}\| + \|g_{m+1}\| \\ &\leq \|g_l - g_{m+1}\| + \varepsilon\|g_l\| \end{aligned}$$

which is equivalent to

$$(1-\varepsilon)\|g_l\| \leq \|g_l - g_{m+1}\|. \tag{6.29}$$

Using the gradient error definition, $e_k = \nabla f(u_k) - g_k$ (see Section 5.3), we can bound the right hand side of (6.29) by

$$\|g_l - g_{m+1}\| \leq \|\nabla f(u_l) - \nabla f(u_{m+1})\| + \|e_l - e_{m+1}\|$$

such that the Lipschitz continuity of ∇f, see **(F3)**, and $u_{k+1} = u_k + s_k$ yields

$$\begin{aligned} \|g_l - g_{m+1}\| &\leq L\,\|u_l - u_{m+1}\| + \|e_l - e_{m+1}\| \\ &\leq L \sum_{\substack{k=l \\ k\in S}}^{m} \|s_k\| + \|e_l - e_{m+1}\|. \end{aligned} \tag{6.30}$$

For the second term of the right hand side in (6.30), one obtains with (5.26) and (6.24)

$$\begin{aligned} \|e_l - e_{m+1}\| &\leq \|e_l\| + \|e_{m+1}\| \\ &\leq \zeta\|g_l\| + \zeta\|g_{m+1}\| \\ &\leq (1+\varepsilon)\zeta\|g_l\|. \end{aligned} \tag{6.31}$$

Combining (6.29), (6.30) and (6.31) results in

$$\begin{aligned} (1-\varepsilon)\|g_l\| &\leq L \sum_{\substack{k=l \\ k\in S}}^{m} \|s_k\| + \|e_l - e_{m+1}\| \\ &\leq L \sum_{\substack{k=l \\ k\in S}}^{m} \|s_k\| + (1+\varepsilon)\zeta\|g_l\| \end{aligned}$$

which is equivalent to

$$\tfrac{1}{L}[(1-\zeta) - \varepsilon(1+\zeta)]\|g_l\| \leq \sum_{\substack{k=l \\ k\in S}}^{m} \|s_k\|. \tag{6.32}$$

Thus, defining $\bar{\varepsilon} = ((1-\zeta) - \varepsilon(1+\zeta))/L$, inequality (6.32) implies

$$\sum_{\substack{k=l \\ k\in S}}^{m} \|s_k\| \geq \bar{\varepsilon}\|g_l\| \tag{6.33}$$

where $\bar{\varepsilon} > 0$ due to the choice of ε in (6.23).

If we return to (6.28) now, two cases can occur. If $\|s_k\| \leq \varepsilon^2 \|g_l\|^2 / c_b$ for all successful iterations $k \in \{l, \ldots, m\}$, due to (6.33) we have

$$f(u_l) - f(u_{m+1}) \geq \eta_1 c_s \varepsilon^2 \bar{\varepsilon} \|g_l\|^3 \tag{6.34}$$

which can be reformulated as

$$\|g_l\|^3 \leq \frac{1}{\eta_1 c_s \varepsilon^2 \bar{\varepsilon}} (f(u_l) - f(u_{m+1})). \tag{6.35}$$

Since f is bounded below, see **(F2)**, and the sequence $\{f(u_k)\}$ is nonincreasing, we have that $f(u_k) \downarrow f^\star$ for some $f^\star > -\infty$. Thus, the right hand side of (6.35) converges to zero, which implies that $\lim_{k \to \infty} \|g_k\| = 0$.

Otherwise, assuming there exists a successful iteration $k_0 \in \{l, \ldots, m\}$ such that $\|s_{k_0}\| > \varepsilon^2 \|g_l\|^2 / c_b$, we obtain in (6.28)

$$f(u_l) - f(u_{m+1}) \geq \frac{\eta_1 c_s \varepsilon^4}{c_b} \|g_l\|^4, \tag{6.36}$$

which allows to conclude as above that $\lim_{k \to \infty} \|g_k\| = 0$. □

Finally, we obtain the following result. Its proof follows immediately with Lemma 5.5.

Corollary 6.5

Let the assumptions of Theorem 6.4 be satisfied.

Then,

$$\lim_{k \to \infty} \|\nabla f(u_k)\| = 0$$

follows.

Remark 6.6

For the above convergence proofs we have to impose an error condition on the gradient accuracy since g_k only is an approximation to $\nabla f(u_k)$. Here, we used the one proposed in Carter [20]. As pointed out before, the assumption on the model function value at the trust-region center point, **(M2)**, is also violated. Nevertheless, in the convergence proofs no explicit error condition on the approximation quality of $m_k(u_k)$ compared to $f(u_k)$ is necessary. This seems to be surprising at a first sight.

At this point, we recall that the model function quality at the base point u_k solely depends on the accuracy of the POD based reduced order solution compared to the finite element solution at u_k. Therefore, the accuracy of the POD based reduced order model enters implicitly into the gradient computation, e.g. via an adjoint equation or a sensitivity equations approach. Thus, inaccurate reduced order solutions affect the gradient accuracy and this is a quantity that is handled by **(M3)**.

Remark 6.7
It is also important to note that the bound on the gradient error, ζ, given by (5.26) is coupled with the trust-region parameters, η_1, η_2, which implies that the choice of these parameters influences the admissible error level. We remark that it is sufficient to satisfy a gradient error condition of the type

$$\frac{|(\nabla f(u_k) - g_k)^T s_k|}{\|g_k\| \, \|s_k\|} \leq \zeta \tag{6.37}$$

for appropriately chosen $\zeta > 0$, which is weaker than (5.26), in order to obtain the convergence result in Theorem 6.3. For this purpose, we have to employ the error condition (6.37) in inequality (6.10) which allows to proceed in the proofs of Theorem 6.2 and Theorem 6.3 as before, cf. Carter [19], Theorem 3.3 and see also error condition (5.23) in Section 5.3. Nevertheless, we decided to employ error condition (5.26) in the convergence proofs, since in this case also the strong global convergence result in Theorem 6.4 can be shown. For a general discussion and illustration of the gradient error conditions we refer to Carter [19].

Remark 6.8
We further remark that since $\omega(m_k, u_k, s_k)$ is a measure for the model's curvature in step direction, assumption (6.18) is an analogue to the standard assumption of uniform boundedness of the model's Hessian in the quadratic model function case, see Th. 5.3.

We close this section with a reference to the recent monograph by Conn et al. [23] which gives a comprehensive survey of trust-region methods for unconstrained and constrained optimization. Presenting a convergence theory for general model functions where also approximate derivatives may be used, in [23] many aspects of the presented convergence theory for the TRPOD algorithm can be found again. However, the assumptions on the problem data in [23] are different from the assumptions used here. We therefore briefly address some of these differences next.

The convergence theory for model functions with inexact gradients in [23], Section 8.4, is based, e.g., on the assumptions that the objective function, f, and each model function, m_k, are twice (continuously) differentiable and that $m_k(u_k) = f(u_k)$ at each trust-region center point holds. The gradient accuracy is also handled with an error condition of the type of error condition (5.26). Moreover, Conn et al. employ model functions with $m_k(u_k) \neq f(u_k)$ in the context of *conditional models*, but do not use the gradient error condition (5.26) for these cases, see [23], Section 9.1. As pointed out in the introduction of this chapter, Conn et al. also treat composite problems and models in the sense of surrogate optimization by considering objective functions of the type $\hat{f}(y(u), u)$. At this, they assume that $y(u_k)$ and the approximation model for $y(\cdot)$ at u_k as well as corresponding derivatives coincide at the trust-region center point, see [23], Section 8.5.

6.2.3 Additional Modifications of the TRPOD Algorithm

We note that the TRPOD algorithm given by Alg. 6.1 provides a lot of space for additional modifications that can lead to interesting variants of the basic algorithm. Especially, variants of the TRPOD algorithm that exploit characteristic properties of the POD based reduced order modelling approach offer the way to refined implementations. Here, we briefly discuss some of these options.

Reducing the Trust-region Radius

The first modification of the basic TRPOD algorithm concerns a practical issue that should be taken into account in an implementation. In the case of a rejection of the new trial step in Algorithm 6.1 we can adapt the reduction of the trust-region radius to the step length of the unsuccessful step. The modified update rule for the trust-region radius then reads

$$\delta_{k+1} \in \begin{cases} [\,\delta_k, \gamma_3\,\delta_k\,], & \text{if} \quad \rho_k \geq \eta_2 \\ [\,\gamma_2\,\delta_k, \delta_k\,), & \text{if} \quad \eta_1 \leq \rho_k < \eta_2 \\ [\,\gamma_1\|s_k\|, \gamma_2\|s_k\|\,], & \text{if} \quad \rho_k < \eta_1 \end{cases} . \tag{6.38}$$

Eventually, this avoids a number of unnecessary intermediate TRS solutions that show no effect in the computation of a new trial step.

Augmenting the POD Subspace

In a trust-region framework based on quadratic model functions the only way to improve the reliability of the model function is to shrink the trust-region by decreasing the trust-region radius, δ_k. In Algorithm 6.1 an insufficient reduced order model as well as an inappropriatly large trust-region radius can lead to the computation of an unsuccessful step. Therefore, in the TRPOD framework the model function quality cannot only be improved by a further restriction of the trust-region, but also if it is possible to use a POD based reduced order model that produces more accurate reduced order state solutions and model function gradients.

Following the discussion in Chapter 3, for the POD technique the number of POD basis functions, M, provides a parameter that gives some control on this approximation quality, if we assume that the input ensemble is left unchanged and cannot be improved significantly by adding snapshots. Increasing the number of POD basis functions, M, yields an augmentation of the corresponding POD subspace for the Galerkin projection. The subtle point of deciding whether to manipulate the trust-region or to manipulate the approximation model's fidelity is discussed in Dennis/Torczon [28].

However, if the decision has been made that the POD subspace should be augmented, the Lanczos method as presented in Chapter 4 is a useful tool for computing the POD

basis elements as they are needed. For this purpose, it is necessary to save the (converged) Lanczos vectors of the recent POD basis computation. If more POD basis functions are needed for building the POD based reduced order model, the Lanczos procedure can be restarted and the building blocks of the reduced order model have to be recomputed, cf. Section 3.5.

Enriching the POD Subspace

In Algorithm 6.1 each POD based control model is based on snapshots related to the current control, snapshots related to previous control iterates are neglected. It is possible to enrich the POD subspace by adding some (or all) snapshots related to previous control iterates to the snapshot set related to the current control, cf. Afanasiev/Hinze [3].

But, caution should be exercised in doing so, in order to avoid the inclusion of unsuitable snapshot information. For large steps, s_k, it can happen that the combined input ensemble of snapshots related to u_k and snapshots related to $u_k + s_k$ includes snapshots of completely different dynamics. Thus, the resulting POD basis elements will reflect an undesirable mixture of dynamics. Furthermore, enriching the input ensemble will always lead to an increased POD based reduced order model dimension if some kind of energy criterion is used to estimate the required POD subspace dimension that should be used for the Galerkin projection, cf. Cor. 3.10 and the discussion afterwards.

Finally, we remark that by including previous snapshot information in the POD basis one intends to include more global aspect of the POD based model function, m_k, but maybe at the price of weakening the local approximation properties. We also refer to Conn et al. [23] for a discussion of *model functions with memory* in the context of trust-region methods.

6.3 Enforcing Gradient Accuracy

6.3.1 Motivation

According to the convergence results presented in Theorem 6.3, Theorem 6.4 and Corollary 6.5, we emphasize the importance of sufficiently accurate model gradients in order to achieve good performance of the TRPOD algorithm. But, we also remark that in the context of POD based reduced order modelling the error condition

$$\|\nabla f(u_k) - g_k\| \leq \zeta \|g_k\| \tag{6.39}$$

is hard to satisfy for $\zeta \in (0, 1-\eta_2)$, especially when $\|g_k\|$ is small (cf. Kelley/Sachs [64]). At this, it is worth noting that $\|g_k\| \to 0,\ k \to \infty$, implies the asymptotic consistency condition $\|\nabla f(u_k) - g_k\| \to 0,\ k \to \infty$.

In the TRPOD approach presented so far, the POD basis functions are derived using snapshot information of the state equation. Then, these basis functions are employed for the discretization of the state equation. On the other hand, we have to compute gradients of the model function m_k during the solution of the trust-region subproblem. If we compute the model gradient ∇m_k using, e.g., an adjoint equation approach, then we use the adjoint equation of the reduced order state equation for this computation. Hence, this implies that we implicitly discretize the original adjoint equation using basis functions that are especially suited for the discretization of the state equation.

Based on these considerations we therefore introduce an approach that takes derivative information of the original problem into account.

6.3.2 The TRPOD Algorithm with Scaled Model Functions

In this subsection we are going to apply a modelling technique that has been proposed for trust-region frameworks with variable-fidelity models by Alexandrov et al. [4, 6] based on an approach presented in Chang et al. [22]. In [22] this idea is introduced as *sensitivity-based scaling*. Its most important property is that for a given current iterate u_k a model function a_k for the objective function f is derived, such that the first-order consistency conditions (6.6), i.e. $a_k(u_k) = f(u_k)$, $\nabla a_k(u_k) = \nabla f(u_k)$, are enforced. In this case, the trust-region method based on the model function a_k operates with exact gradients.

For the definition of this first-order consistent model a_k, let us consider an objective function f and a corresponding model m_k at u_k where the first-order consistency conditions (6.6) are not necessarily satisfied. Furthermore, we assume that $f : \mathbb{R}^n \to \mathbb{R}$ is a nonnegative function, i.e. $f(u) \geq 0$ for all $u \in \mathbb{R}^n$, and that this property carries over to each model function, m_k, for f. We note that this property is satisfied, e.g., if f represents a discretization of one of the cost functionals given in (2.18) or (2.19). Additionally, we assume that the gradient ∇f is available.

Definition 6.9 (Scaling Factor)
Consider a model function m_k for f at the current iterate u_k with $\|g_k\| \neq 0$, where the first-order consistency conditions (6.6) are not necessarily satisfied. We define the *scaling factor* $\sigma_k(u)$ by

$$\sigma_k(u) = \frac{f(u)}{m_k(u)} \tag{6.40}$$

and the *linearized scaling factor* $\bar{\sigma}_k(u)$ for $u = u_k + s$ by

$$\bar{\sigma}_k(u_k + s) = \sigma_k(u_k) + \nabla \sigma_k(u_k)^T s. \tag{6.41}$$

We remark that, if the scaling factor, $\sigma_k(u_k)$, at u_k given by (6.40) is close to one, the model function value $m_k(u_k)$ is close to $f(u_k)$ and little scaling is required in the representation $f(u_k) = \sigma_k(u_k)\, m_k(u_k)$. In case the scaling factor is large, the model function underestimates the actual function value, and vice versa.

Furthermore, the scaling factor $\sigma_k(u)$ for $u = u_k + s$ can be (locally) approximated by the linearized scaling factor $\bar{\sigma}_k(u_k + s)$. Based on the linearized scaling factor (6.41) we can derive a first-order consistent model function a_k for f at u_k.

Lemma 6.10
Consider a model function m_k for f at the current iterate u_k with $\|g_k\| \neq 0$, where the first-order consistency conditions (6.6) are not necessarily satisfied. We define $\bar{\sigma}_k(u)$ as in (6.41).

We then have that the model function a_k for f at u_k defined by

$$a_k(u_k + s) = \bar{\sigma}_k(u_k + s)\, m_k(u_k + s) \tag{6.42}$$

satisfies

$$a_k(u_k) = f(u_k) \quad \textit{and} \quad \nabla a_k(u_k) = \nabla f(u_k).$$

Proof: The choice $s = 0$ in (6.42), together with (6.41) and (6.40) yields

$$a_k(u_k) = \bar{\sigma}_k(u_k)\, m_k(u_k) = \sigma_k(u_k)\, m_k(u_k) = f(u_k).$$

Furthermore, the components of the gradient $\nabla a_k \in I\!R^n$ are given by

$$\begin{aligned} (\nabla a_k(u))_i &= (\nabla \bar{\sigma}_k(u))_i\, m_k(u) + \bar{\sigma}_k(u)\, (\nabla m_k(u))_i \\ &= (\nabla \sigma_k(u_k))_i\, m_k(u) + \Big[\sigma_k(u_k) + \nabla \sigma_k(u_k)^T (u - u_k)\Big]\, (\nabla m_k(u))_i \end{aligned} \tag{6.43}$$

for $i = 1, \ldots, n$, where we employed (6.41) to obtain (6.43). By inserting

$$(\nabla \sigma_k(u_k))_i = \frac{(\nabla f(u_k))_i\, m_k(u_k) - f(u_k)\, (\nabla m_k(u_k))_i}{m_k(u_k)^2} \tag{6.44}$$

and $u = u_k$ in (6.43), we obtain $\nabla a_k(u_k) = \nabla f(u_k)$. □

Consequently, if the model function m_k is based on a reduced order modelling approach as presented in Section 6.2, we can perform a correction of that model function such that a new model function a_k is obtained that is characterized by $a_k(u_k) = f(u_k)$ and $\nabla a_k(u_k) = \nabla f(u_k)$. The corresponding TRPOD algorithm with model function a_k is given in Algorithm 6.11.

We emphasize that, in order to obtain the scaled model function a_k, it is important to have the true gradient $\nabla f(u_k)$. Compared to the basic TRPOD approach, this requires an additional computational effort for the computation of $\nabla f(u_k)$ after each successful iteration. However, the convergence analysis of the corresponding TRPOD algorithm where the model function m_k given by (6.8) is replaced by a model function a_k given by (6.42) is independent of a gradient error condition in the sense of (6.39). Thus, the convergence results of Section 6.2 are still valid.

Algorithm 6.11 (TRPOD Algorithm with Scaled Model Functions)
Initialization: Choose $0 < \eta_1 < \eta_2 < 1$, $0 < \gamma_1 \leq \gamma_2 < 1 \leq \gamma_3$, an initial trust-region radius $\delta_0 > 0$ and an initial iterate u_0. Compute a snapshot set $\mathcal{Y}_0$ corresponding to control u_0, compute $f(u_0)$ and $\nabla f(u_0)$. Set $k = 0$.

1. Compute a POD basis $\mathcal{Y}_k^{POD}$ and build a POD based control model.

2. Compute an approximate minimizer, s_k, of the model function $a_k(u_k + s)$ given by (6.42) within the trust-region, $\|s\| \leq \delta_k$, using SDA.

3. Compute a snapshot set $\mathcal{Y}_{k+}$ corresponding to control $u_k + s_k$ and compute $f(u_k + s_k)$. Set

$$\rho_k = \frac{ared_k(s_k)}{pred_k(s_k)}.$$

4. Update the trust-region radius:

- If $\rho_k \geq \eta_2$:
  Set $u_{k+1} = u_k + s_k$, $f(u_{k+1}) = f(u_k + s_k)$, $\mathcal{Y}_{k+1} = \mathcal{Y}_{k+}$.
  Compute $\nabla f(u_{k+1})$ and choose $\delta_{k+1} \in [\delta_k, \gamma_3 \delta_k]$. Set $k = k + 1$ and GOTO 1.

- If $\eta_1 \leq \rho_k < \eta_2$:
  Set $u_{k+1} = u_k + s_k$, $f(u_{k+1}) = f(u_k + s_k)$, $\mathcal{Y}_{k+1} = \mathcal{Y}_{k+}$.
  Compute $\nabla f(u_{k+1})$ and choose $\delta_{k+1} \in [\gamma_2 \delta_k, \delta_k)$. Set $k = k + 1$ and GOTO 1.

- If $\rho_k < \eta_1$:
  Set $u_{k+1} = u_k$ and choose $\delta_{k+1} \in [\gamma_1 \delta_k, \gamma_2 \delta_k]$. Set $k = k + 1$ and GOTO 2.

A detailed presentation of $\nabla a_k \in \mathbb{R}^n$ reveals another important property of the model function a_k. The model gradient ∇a_k with components

$$\begin{aligned}(\nabla a_k(u_k + s))_i = &\left(\frac{(\nabla f(u_k))_i m_k(u_k) - f(u_k)(\nabla m_k(u_k))_i}{m_k(u_k)^2}\right) m_k(u_k + s) \\ &+ \left(\frac{f(u_k)}{m_k(u_k)} + \sum_{j=1}^n \frac{(\nabla f(u_k))_j m_k(u_k) - f(u_k)(\nabla m_k(u_k))_j}{m_k(u_k)^2}(s)_j\right)(\nabla m_k(u_k + s))_i \quad (6.45)\end{aligned}$$

for $i = 1, \ldots, n$, only depends on $f(u_k)$, $\nabla f(u_k)$, $m_k(u_k + s)$ and $\nabla m_k(u_k + s)$. Thus, once the true objective and its gradient at the current iterate have been computed, the evaluation of the new model function $a_k(u_k + s)$ for given s and its gradient only requires computations with the model m_k and no further evaluations of f or ∇f are necessary. We recall that $a_k(u_k + s)$, $\nabla a_k(u_k + s)$ have to be evaluated, e.g., during Phase 2 of the step determination algorithm (SDA).

6.3.3 A Hybrid TRPOD Algorithm

Since we expected that it is hard to satisfy the gradient error condition (6.39) for the TRPOD algorithm, especially for $\|\nabla g_k\|$ approaching zero, we introduced the TRPOD algorithm with scaled model functions in the previous section.

On the other hand, the basic TRPOD algorithm does not require to compute the true gradient of the objective function if a model update is necessary. Therefore, it is interesting to derive an algorithm that combines both, the features of TRPOD with and without scaled model functions. This can be accomplished by concatenating both algorithms.

Such a hybrid algorithm starts with a trust-region framework according to Algorithm 6.1, because in an early phase of the optimization process it is likely that the basic TRPOD algorithm will reveal a good performance. When the TRPOD iterates start to exhibit a convergent behaviour it may be preferable to switch to a TRPOD algorithm with scaled model functions that uses exact gradient information. In this case, we can guarantee accurate derivatives.

Crucial at this, is to decide whether to switch from TRPOD using only reduced order information during the solution of the trust-region subproblems to the TRPOD algorithm with scaled model functions. We are aware of the fact that it is not easy to find a criterion that always guarantees that the TRPOD component (without scaling) is maintained as long as possible, since it avoids the computation of ∇f.

6.4 Numerical Results

6.4.1 Results for the TRPOD Algorithm

In this section we present numerical results for the the TRPOD algorithm (Algorithm 6.1) applied to the control of the driven cavity problem with objective functions of tracking-type, cf. (2.18),

$$J(y,\mathtt{u}) = \tfrac{1}{2}\|y - y^d\|^2_{L^2(0,T;L^2(\Omega_{obs}))} + \tfrac{\gamma}{2}\|\mathtt{u}\|^2_U. \tag{6.46}$$

All flow calculations were carried out with the flow solver FEATFLOW at Reynolds number $Re = 200$, on the time interval $[0,T] = [0,5]$, using a uniform 33×33 grid for the spatial discretization (see Appendix A), all other computations with MATLAB, on a SUN SPARCstation 20. The parameters for the TRPOD algorithm are given in Table 6.1.

TR Framework		**Step Determination**	
$\eta_1,\ \eta_2$	0.25, 0.75	$\alpha,\ \beta$	0.25, 0.75
$\gamma_1,\ \gamma_2,\ \gamma_3$	0.5, 0.5, 2	$\nu_1,\ \nu_2,\ \nu_3,\ \mu$	0.1, 0.1, 0.1 , 1

Table 6.1: TRPOD Parameters

Example 1: In the first example we consider the control of the standard cavity flow, where the control, $\mathtt{u}$, is the horizontal velocity on the top wall, i.e.

$$y(x,t) = (\mathtt{u}(t), 0)^T, \quad x \in \Gamma_{top},\ t \in (0,T),$$

and

$$y(x,t) = (0,0)^T, \quad x \in \Gamma \setminus \Gamma_{top}, t\ \in (0,T).$$

These boundary conditions can be represented by $b(x) = (1,0)^T,\ x \in \Gamma_{top},\ c(x) = (0,0)^T$, $x \in \Gamma \setminus \Gamma_{top}$, with respect to the notation in Chapter 2.

Since we intend to check the assumptions of Theorem 6.3 for this example, we restrict the number of (discrete) control variables using the expansion $\mathtt{u}(t) = \sum_{j=1}^{n} u^j \varphi_j(t)$ with $\varphi_1(t) \equiv 1$ and $\varphi_{2j}(t) = \cos(2j\pi t/T),\ \varphi_{2j+1}(t) = \sin(2j\pi t/T),\ j = 1,2,3,$ for all $t \in [0,T]$. Thus, we obtain $n = 7$ discrete control variables and the discretization of the cost functional in (6.46) yields $f : I\!R^7 \to I\!R$.

For this example, the desired state, y^d, corresponds to a prescribed control given by

$$u^d = (0.8,\ 0.1,\ -0.3,\ 0,\ 0.1,\ -0.3,\ 0)^T \in I\!R^7.$$

Furthermore, we choose $\Omega_{obs} = (\,0.44,\ 0.56\,) \times (\,0.44,\ 0.56\,)$ in the center of the domain $\Omega = (0,\ 1) \times (0,1)$, initialized the TRPOD algorithm with $\mathtt{u}_0(t) \equiv 0.01$ for all $t \in [0,T]$, $\delta_0 = 0.25$, and set $\gamma = 0$.

Table 6.2 lists the values of the objective function, $f(u_k)$, given by (6.9) and the model function, $m_k(u_k)$, given by (6.8) at the beginning of each iteration k, for given current iterate $u_k \in I\!R^7$. Furthermore, the trust region radius, δ_k, the step length, $\|s_k\|$, of the trial step, the quotient of actual reduction to the predicted reduction, ρ_k, and the number of POD basis elements, M, for each TRPOD iteration are shown. Additionally, we give the norm of the model gradient, $\|g_k\|$, at the trust-region center point for each model function.

k	$f(u_k)$	$m_k(u_k)$	δ_k	$\|s_k\|$	ρ_k	$\|g_k\|$	$\zeta_k^{(1)}$	$\zeta_k^{(2)}$	M
0	5.99658 E-4	5.960310 E-4	0.25	0.25	1.58	1.0017 E-5	0.88	0.54	4
1	3.16328 E-4	3.195711 E-4	0.5	0.46	1.51	1.3771 E-5	0.15	0.13	4
2	3.07127 E-5	3.768390 E-5	1	0.20	0.81	4.8561 E-6	0.17	0.13	4
3	2.60034 E-6	2.454936 E-6	2	0.22	0.97	2.8924 E-7	0.93	0.12	6
4	4.67446 E-7								

Table 6.2: Example 1: TRPOD Results

The dimensions of the POD reduced order models, M, are chosen according to the energy criterion (with $\bar{\epsilon}=0.999$) combined with a singular value gap criterion, cf. Section 3.4. In each iteration, $k=1,2,3$, we employ $P=80$ snapshots for the computation of the POD basis functions. Merely, in the initial iteration, $k=0$, we choose $P=10$, because for $\mathtt{u}_0 \equiv 0.01$ we get no significant dynamics. For this example we also check if the gradient error condition (5.26)

$$\zeta_k^{(1)} = \frac{\|\nabla f(u_k) - g_k\|}{\|g_k\|} \leq \zeta$$

or the error condition (6.37)

$$\zeta_k^{(2)} = \frac{|(\nabla f(u_k) - g_k)^T s_k|}{\|g_k\|\|s_k\|} \leq \zeta$$

is satisfied for $\zeta \in (\,0, 1-\eta_2)$. For the computation of $\nabla f(u_k)$ and $\nabla f(u_k)^T s_k$ we use finite difference approximations to the derivatives, which is workable because of the small number of discrete control variables. The results for $\zeta_k^{(1)}$ in Table 6.2 illustrate the discussion in Section 6.3 concerning the model gradient accuracy. Nevertheless, we can realize that $\zeta_k^{(2)} < 1-\eta_2 = 0.25$ holds for $k \geq 1$. Based on the values of s_k, $m_k(u_k)$, $m_k(u_k+s_k)$ and g_k we also find that the bound on the model curvature given by (6.18) is satisfied with $b_k \leq c_b = 2$.

Finally, we remark that we stopped the TRPOD algorithm after four iterations where the objective function value is reduced from $f(u_0) = 5.99 \cdot 10^{-4}$ to $f(u_4) = 4.67 \cdot 10^{-7}$.

In Figure 6.1, the control iterates $\mathtt{u}_k(t)$, $k=0,\ldots,4$, for all $t \in [0,T]$ are presented. This illustrates how the iterates approach $\mathtt{u}^d$ which defines the desired flow field, y^d.

Figure 6.2 depicts $\mathtt{u}^d$, $\mathtt{u}_4$, which is referred to as $\mathtt{u}_{opt}$ with TRPOD, and the solution of the reduced order control problem without TRPOD mechanism. For the computation of the latter, we take the POD based reduced order model corresponding to $k=0$ and perform the optimization without additional trust-region constraint, employing only this single model. We denote the corresponding solution by $\mathtt{u}_{opt}$ without TRPOD. Figure 6.3 shows the corresponding improvements in the objective function for this example, when the control is applied to the full system, $\|y^N(\mathtt{u}) - y^d\|_{L^2(\Omega_{obs})}$ for each $t \in [0,T]$. Again, we compare $\mathtt{u}_{opt}(t)$ with TRPOD to $\mathtt{u}_{opt}(t)$ without TRPOD. Both figures illustrate that the trust-region framework with its update mechanism for the POD based reduced order models improves the results significantly.

For illustration purposes, we also give vector plots of the POD basis functions and the reference flow field for the control, y_b^N, see (3.33). These plots, for the initial flow configuration ($k=0$) in Figure 6.4 and the final flow configuration ($k=4$) in Figure 6.5, visualize the changes in the POD bases for this example.

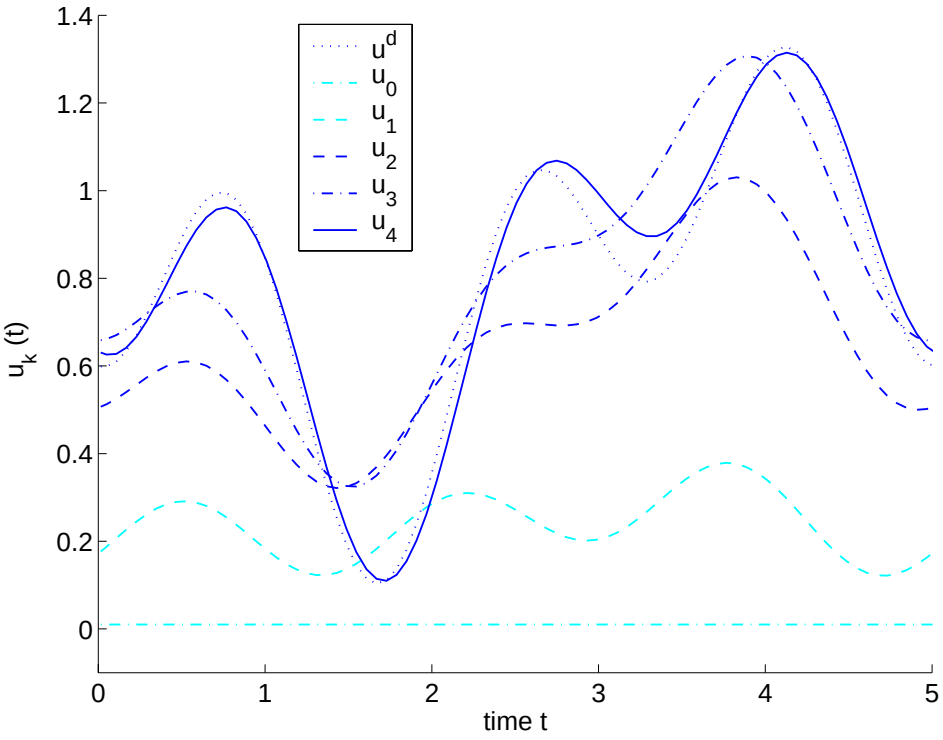

Figure 6.1: Example 1: Control Iterates

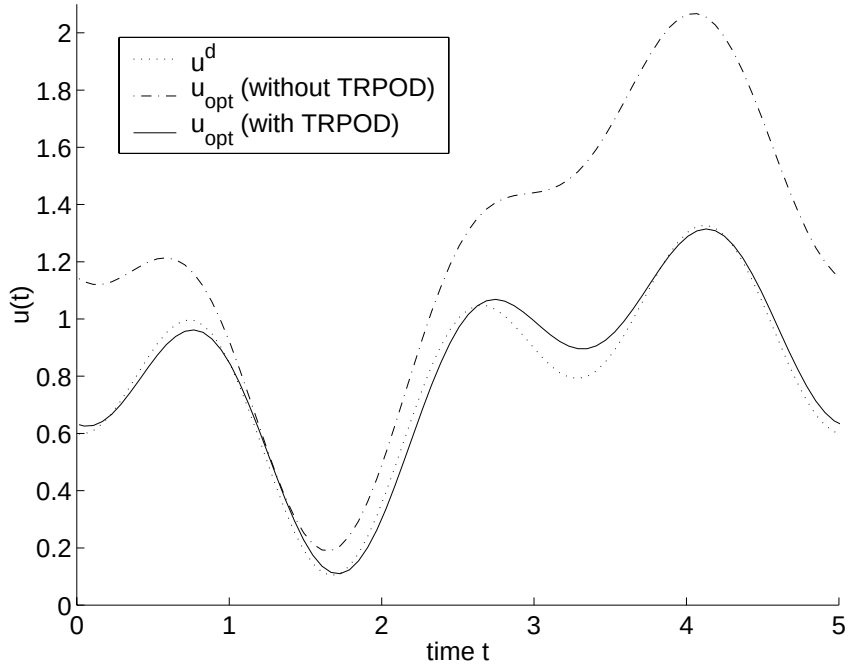

Figure 6.2: Example 1: Comparison of Controls

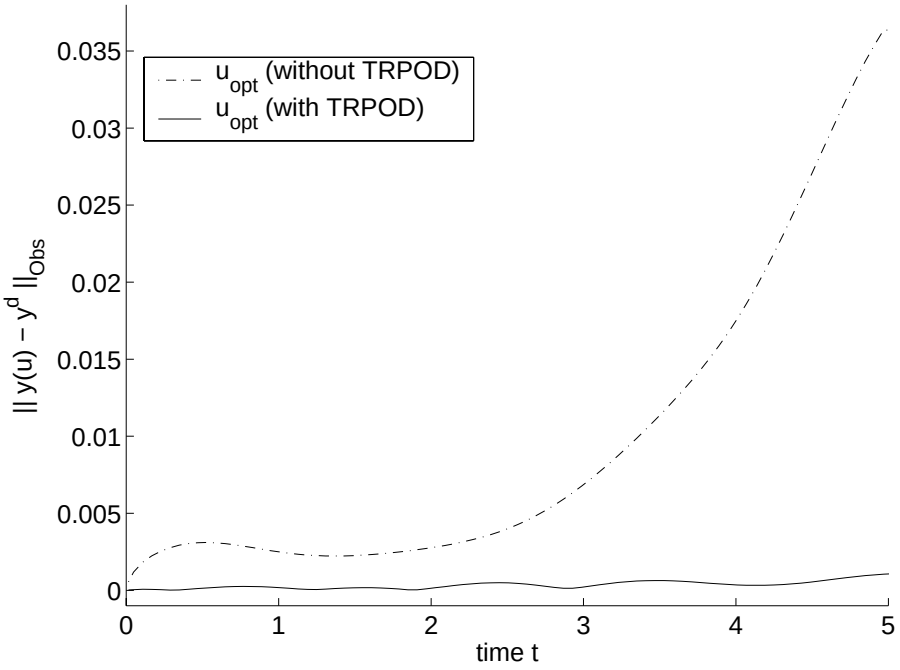

Figure 6.3: Example 1: Comparison of $\|y(\mathtt{u}) - y^d\|$

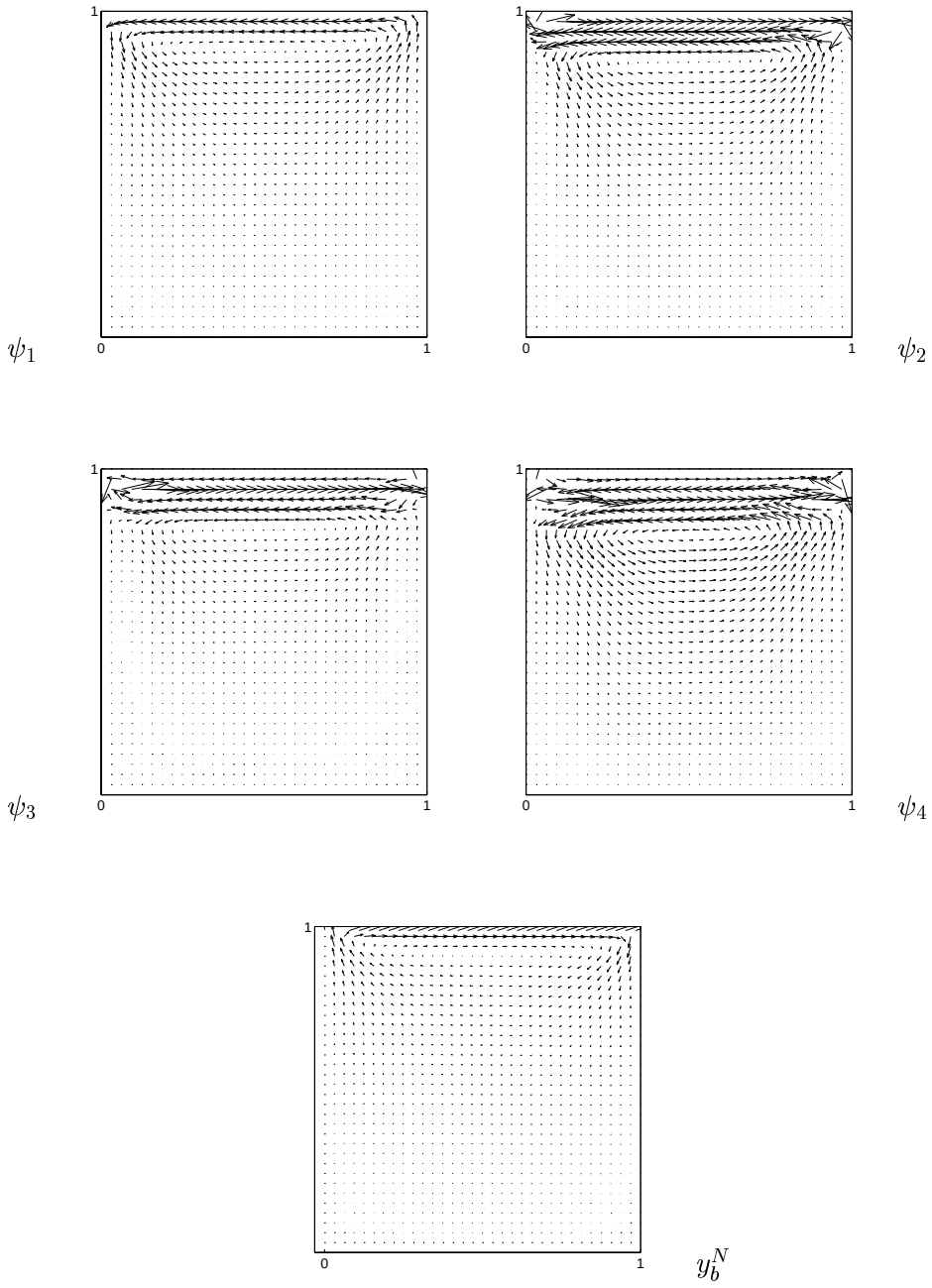

Figure 6.4: Example 1: POD basis functions $\psi_1 - \psi_4$ and y_b^N corresponding to u_0

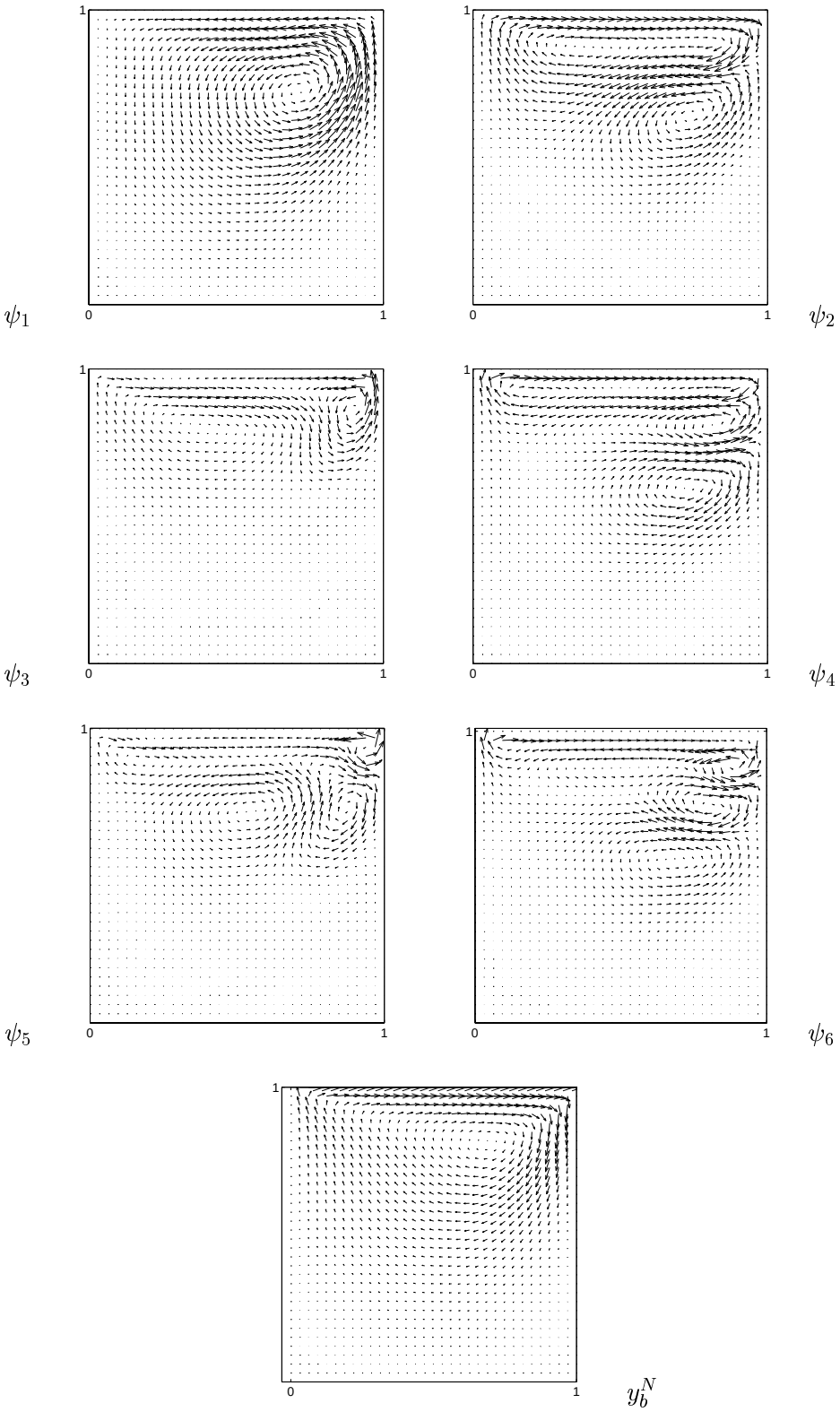

Figure 6.5: Example 1: POD basis functions $\psi_1 - \psi_6$ and y_b^N corresponding to u_4

Example 2. In the second example we consider the control of a cavity flow where the control action is a horizontal bottom wall movement. As a result, there evolves a second vortex counteracting the first vortex that is induced by the top wall movement. We set

$$\begin{aligned} y(x,t) &= (1,0)^T, && x \in \Gamma_{top},\ t \in (0,T), \\ y(x,t) &= (\mathtt{u}(t),0)^T, && x \in \Gamma_{bot},\ t \in (0,T), \quad \text{and} \\ y(x,t) &= (0,0)^T, && x \in \Gamma \setminus (\Gamma_{top} \cup \Gamma_{bot}),\ t \in (0,T), \end{aligned}$$

which implies $b(x) = (1,0)^T,\ x \in \Gamma_{bot}$, $c(x) = (1,0)^T,\ x \in \Gamma_{top}$, and $c(x) = (0,0)^T$, $x \in \Gamma \setminus (\Gamma_{top} \cup \Gamma_{bot})$ in the notation of Chapter 2. Thus, the control, $\mathtt{u}$, is the time-dependent boundary velocity at the bottom of the cavity.

The desired state, y^d, corresponds to an (arbitrarily) predefined bottom wall velocity

$$\begin{aligned} \mathtt{u}^d(t) &= 1.2 + 0.5\cos((t-2.5)\pi/2.5) - 0.1\cos((t-1)\pi) \\ &\quad + (0.1t-0.2)\cos((t-0.5)\pi/0.5) \end{aligned}$$

for all $t \in [0,T]$. Due to a time discretization of $\delta t = 0.03$ we get $n = 167$ (discrete) control variables, $u \in I\!R^n$. We choose the observation domain to be the entire cavity, $\Omega_{obs} = \Omega$, initialize the TRPOD algorithm with initial control $\mathtt{u}_0(t) \equiv 0.1$ for all $t \in [0,T]$, initial trust-region radius $\delta_0 = 2$, and set $\gamma = 0$.

Table 6.3 lists the values of the objective function, $f(u_k)$, and the model function, $m_k(u_k)$, at the beginning of each iteration k, the trust region radius, δ_k, and the step length, $\|s_k\|$, of the computed trial step. Furthermore, the norm of the model gradient, $\|g_k\|$, the quotient of actual reduction to the predicted reduction, ρ_k, and the number of POD basis elements, M, are given. We stopped the TRPOD algorithm after 5 iterations, where the objective function value is reduced from $f(u_0) = 0.229239$ to $f(u_5) = 0.000132$. Again, the dimensions of the POD reduced order models, M, are chosen according to the energy criterion (with $\bar{\epsilon} = 0.999$) combined with a singular value gap criterion.

k	$f(u_k)$	$m_k(u_k)$	δ_k	$\|s_k\|$	ρ_k	$\|g_k\|$	M
0	0.229239	0.188274	2	2.00	2.36	0.01815	5
1	0.149097	0.148143	4	4.00	1.27	0.01596	5
2	0.080427	0.080532	8	6.84	1.47	0.01374	5
3	0.006864	0.008835	16	2.87	0.93	0.00592	6
4	0.000246	0.000795	32	0.29	3.61	0.00041	8
5	0.000132						

Table 6.3: Example 2: TRPOD Results

In Figure 6.6, all control iterates of the optimization process are illustrated. It is shown how these iterates approach $\mathtt{u}^d$ that defines the desired state.

Figure 6.7 compares $\mathtt{u}^d$, $\mathtt{u}_5$, which is referred to as $\mathtt{u}_{opt}$ with TRPOD, and the solution of the reduced order control problem without trust-region modification. Again, we denote the latter by $\mathtt{u}_{opt}$ without TRPOD. Figure 6.8 illustrates the improvements in the objective function when the control $\mathtt{u}_{opt}$ with TRPOD is applied to the full system instead of the control $\mathtt{u}_{opt}$ without TRPOD.

Furthermore, Figure 6.9 depicts vector plots of the POD basis functions as well as $\bar{y}^N$ and y_b^N corresponding to the initial control ($k=0$). For a definition of these flow fields, we refer to Section 3.5.2. Figure 6.10 depicts the corresponding vector plots for the final control iterate of the TRPOD run ($k=5$). Here, we only give the first 6 POD basis functions although we employ 8 POD basis functions for the reduced order model in this iteration. Again, these figures illustrate the significant change in the POD basis due to the update mechanism inherent in the TRPOD algorithm. Here, the POD basis functions at iteration $k=0$ scarcely exhibit the influence of the control action at the bottom wall. But, the control action is clearly reflected in the POD basis functions shown in Fig. 6.10.

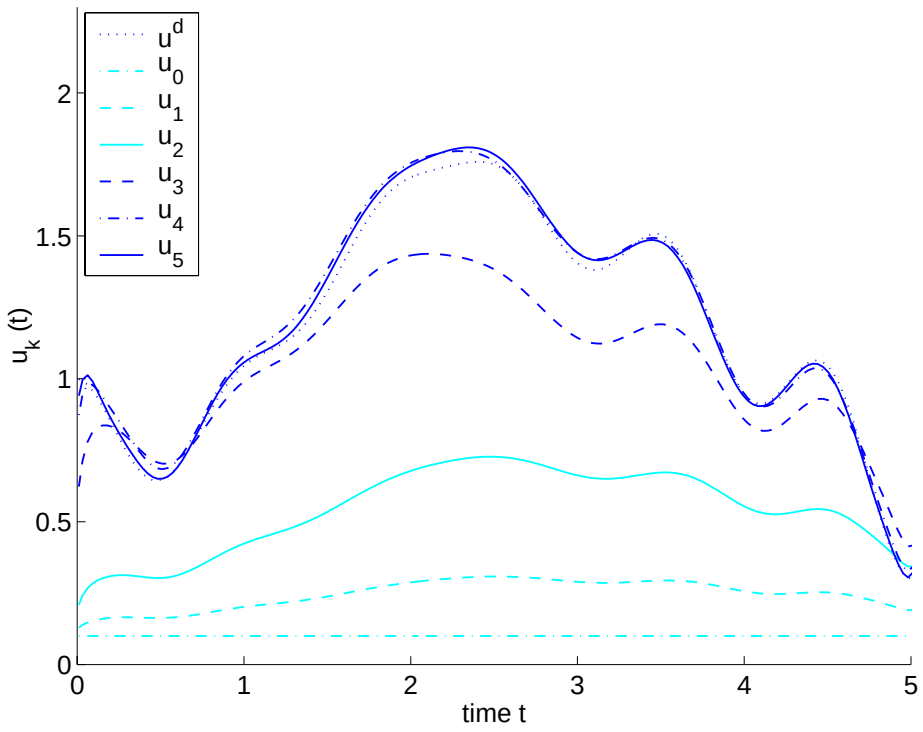

Figure 6.6: Example 2: Control Iterates

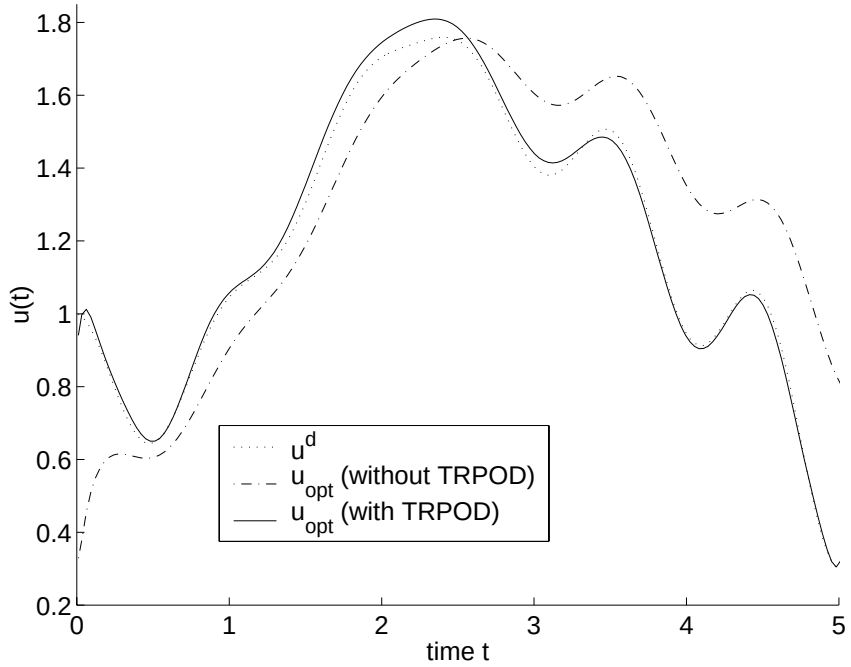

Figure 6.7: Example 2: Comparison of Controls

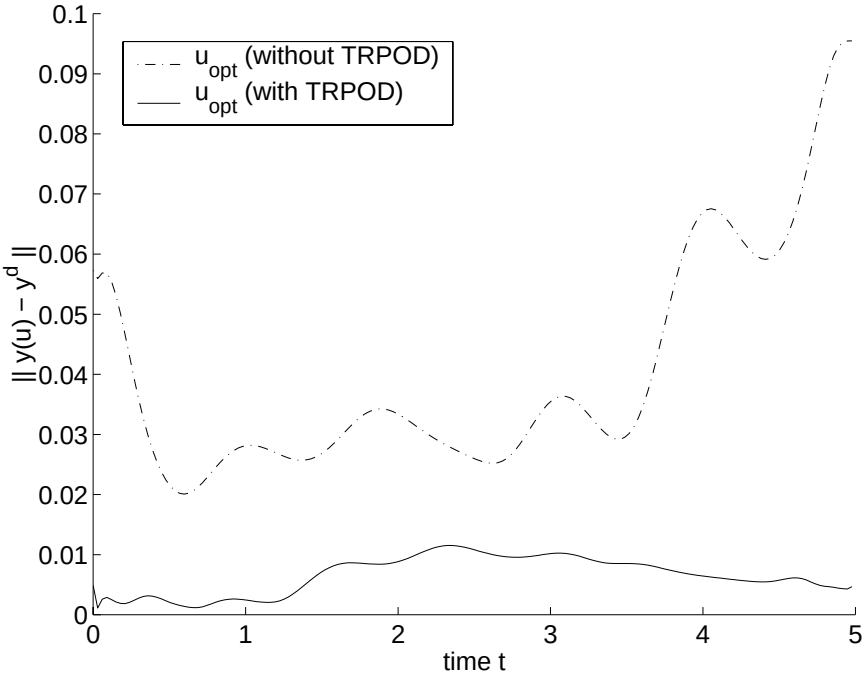

Figure 6.8: Example 2: Comparison of $\|y(\mathtt{u}) - y^d\|$

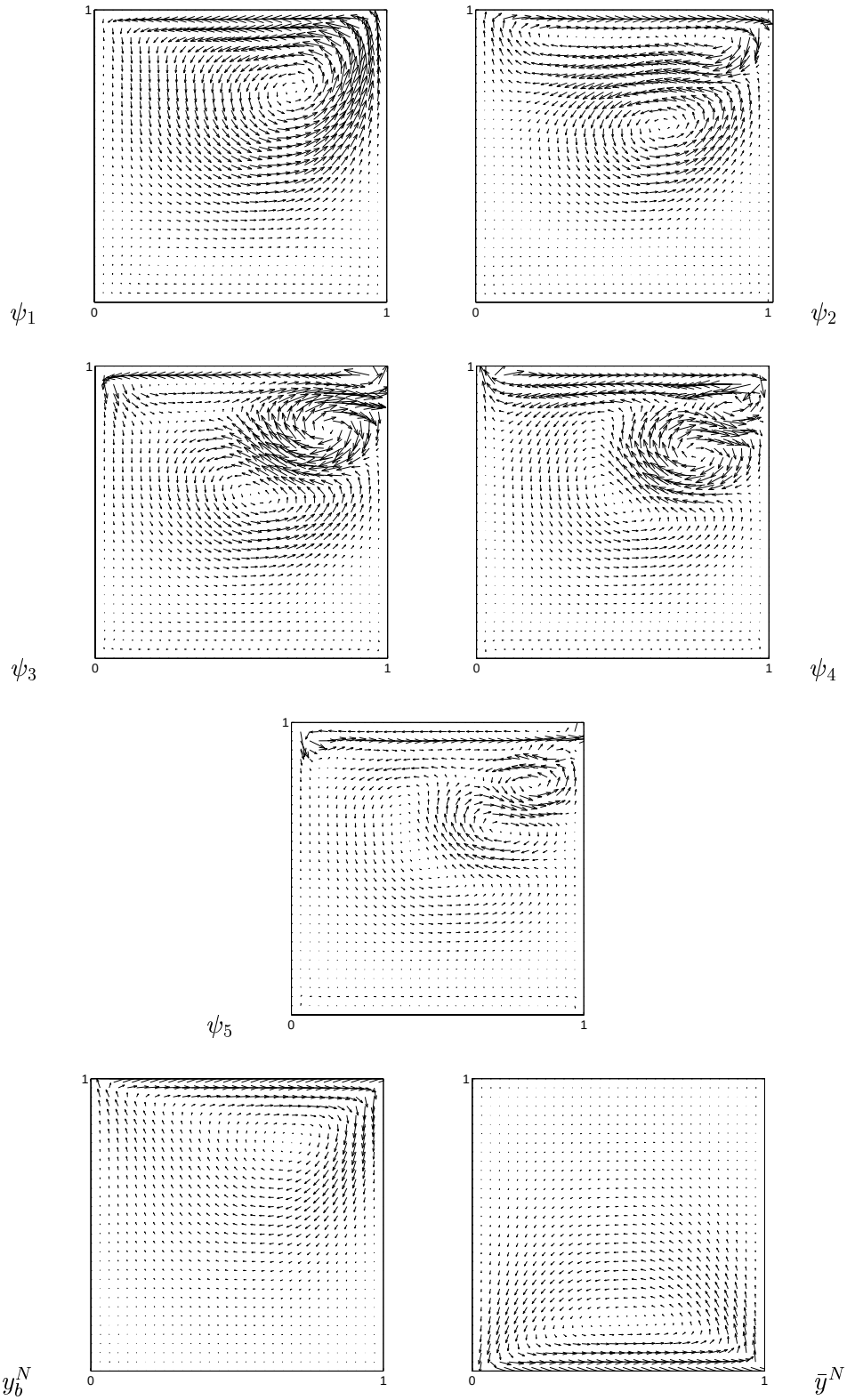

Figure 6.9: Example 2: POD basis functions $\psi_1 - \psi_5$, y_b^N and $\bar{y}^N$ corresponding to u_0

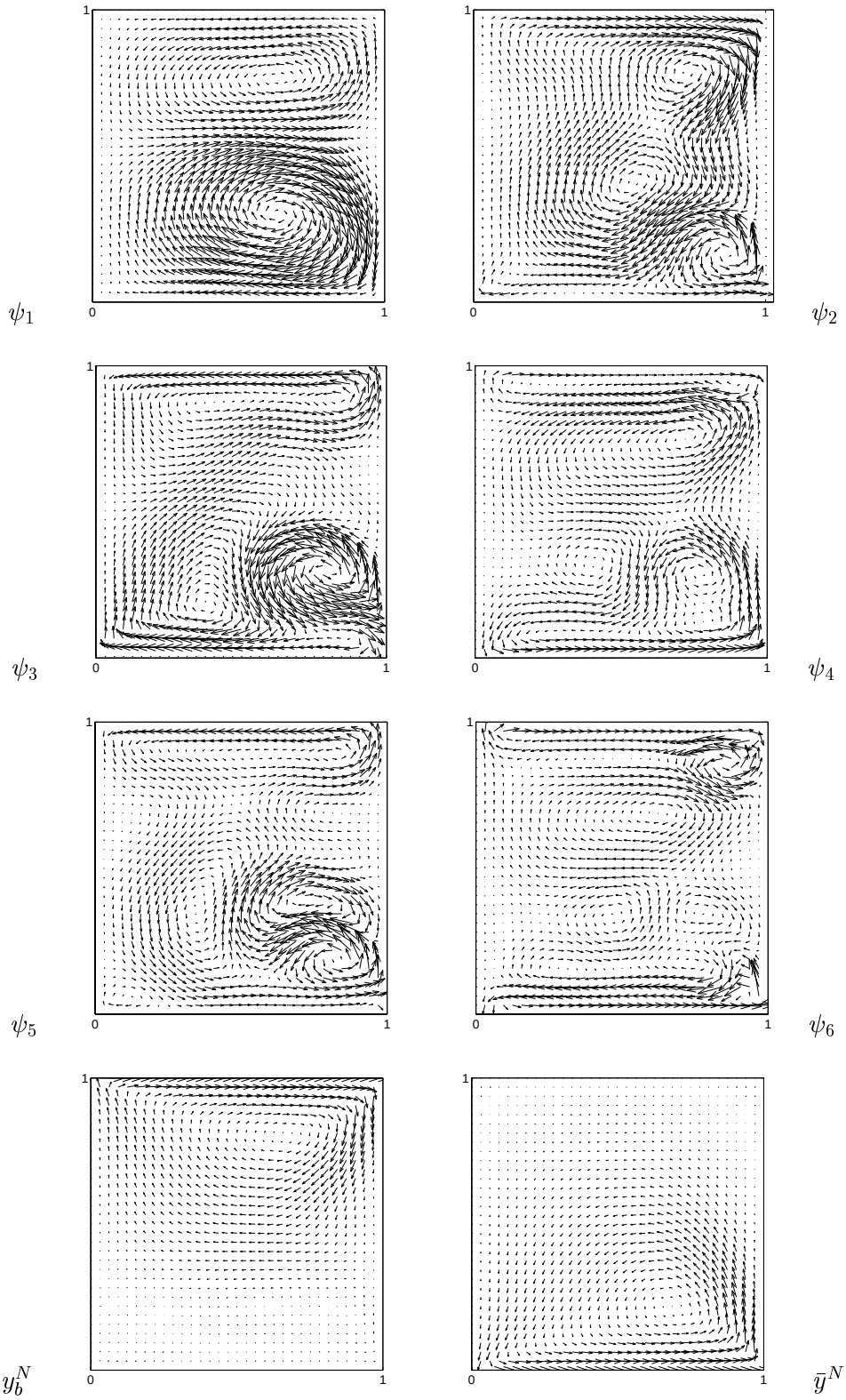

Figure 6.10: Example 2: POD basis functions $\psi_1 - \psi_6$, y_b^N and $\bar{y}^N$ corresponding to u_5

6.4.2 Results for the Hybrid TRPOD Algorithm

For the presentation of numerical results concerning the hybrid TRPOD algorithm, we consider an optimal control problem involving the linear heat equation on $\Omega = (0,1)$:

$$\text{Minimize} \quad \mathtt{f}(\mathtt{u}) = \tfrac{1}{2}\int_0^T (\tau(\mathtt{u};0,t) - q(t))^2\, dt + \tfrac{\gamma}{2}\int_0^T \mathtt{u}(t)^2\, dt \tag{6.47}$$

where $\tau(x,t) = \tau(\mathtt{u};x,t)$ is the solution of

$$\begin{aligned} \tau_t(x,t) - \kappa\,\tau_{xx}(x,t) &= 0, \qquad 0 < x < 1,\ 0 < t < T & (6.48)\\ \tau(x,0) &= \tau_0(x), \quad 0 < x < 1 & (6.49) \end{aligned}$$

with boundary conditions

$$\begin{aligned} \kappa\,\tau_x(1,t) &= \alpha(\tau(1,t) - \mathtt{u}(t)), \quad 0 < t < T & (6.50)\\ \kappa\,\tau_x(0,t) &= 0, \qquad 0 < t < T. & (6.51) \end{aligned}$$

In (6.47) – (6.51), $\tau(x,t)$ denotes the temperature in $x \in \Omega$ at time $t \in (0,T)$. Here, the parameters κ and α denote the heat conductivity and heat transfer coefficient, respectively, and $\gamma > 0$ is a regularization parameter. The cost functional is of tracking-type, where $q(t)$, $t \in (0,T)$ denotes a desired temperature profile at $x = 0$. The control, $\mathtt{u}(t)$ for $t \in (0,T)$, is applied at $x = 1$.

Following [65], the gradient of $\mathtt{f}$ is given by,

$$(\nabla \mathtt{f}(\mathtt{u}))(t) = \kappa\,\alpha\,\pi(1,t) + \gamma\,\mathtt{u}(t) \tag{6.52}$$

where $\pi(x,t)$ is the solution of the adjoint equation

$$\begin{aligned} -\pi_t(x,t) - \kappa\,\pi_{xx}(x,t) &= 0, \quad 0 < x < 1,\ 0 < t < T\\ \pi(x,T) &= 0, \quad 0 < x < 1 \end{aligned}$$

with boundary conditions

$$\begin{aligned} \kappa\,\pi_x(1,t) &= -\alpha\pi(1,t), \qquad 0 < t < T\\ \kappa\,\pi_x(0,t) &= q(t) - y(0,t), \quad 0 < t < T. \end{aligned}$$

We choose Problem (6.47) for the illustration of the ideas of the hybrid TRPOD algorithm, since the state equation as well as the adjoint equation can be solved easily.

For the computation of the following numerical results we use a finite element discretization with piecewise linear basis functions for the spatial discretization and a backward Euler method for the time discretization. We choose $T = 1$, discretize with $\delta t = 0.01$, $\delta x = 0.02$ and set the parameter values: $\kappa = 0.3$, $\alpha = 1 \cdot 10^{-3}$, $\gamma = 5 \cdot 10^{-3}$. The desired temperature profile is given by $q(t) = t$, the initial control is $\mathtt{u}_0 \equiv 0.001$. Furthermore, we set $\delta_0 = 10$. In the following, $u \in I\!R^n$, $n = 101$, denotes the discretized control and $f : I\!R^n \to I\!R$ the discretized cost functional in (6.47).

In Table 6.4–Table 6.6 we compare the performance of the basic TRPOD algorithm, with the TRPOD algorithm based on scaled model functions and the hybrid TRPOD algorithm for the solution of the control problem given in (6.47).

In all tables we present at each iteration, k, the objective function value, $f(u_k)$, the current trust-region radius, δ_k, the norm of the computed trial step, $\|s_k\|$, the value of ρ_k, and the POD subspace dimension, M, based on an energy criterion ($\bar{\epsilon}=0.9999$). We note that we employed the trust-region update rule (6.38).

For the basic TRPOD algorithm we also give the model function value, $m_k(u_k)$, and the norm of the model gradient, $\|g_k\|$. Furthermore, the value of $\zeta_k = \|\nabla f(u_k) - g_k\|/\|g_k\|$ according to (5.26) is shown. For this purpose, we have to compute the gradient of f via (6.52) in order to compute the quantity ζ_k, but we do not use $\nabla f(u_k)$ besides this.

In the case that scaled model functions are used, we renounce to give $a_k(u_k)$ (since it agrees with $f(u_k)$), but we give the scaling factor $\sigma_k(u_k)$ instead. Here, we have to compute the gradient of f in order to build the new model a_k. Since $\nabla f(u_k)$ and $\nabla a_k(u_k)$ coincide, we also present the norm of the true gradient, $\|\nabla f(u_k)\|$.

The numerical results given in Tables 6.4–6.6 illustrate the previous discussion. The basic TRPOD algorithm leads to good results. Although the gradient error condition, $\zeta_k \leq \zeta < 1-\eta_2 = 0.25$, is not satisfied for $k \geq 2$, the trial steps computed with the models m_k lead to a certain decrease in the true objective. We achieve an objective function value of $f(u_9) = 0.003016$. Furthermore, it is obvious that the gradient error deteriorates as indicated by the value of ζ_k, when $\|g_k\|$ approaches 0.

The TRPOD algorithm with scaled model functions leads to hardly better results. Here, we obtain $f(u_6) = 0.002945$ at the final iterate.

The results for the hybrid TRPOD algorithm are similar. In this case we switch from the TRPOD algorithm without scaling to scaled model functions as soon as the trust-region radius is smaller than a prescribed value, $\underline{\delta} = 0.2$, and an unsuccessful iteration is met ($\delta_k < \underline{\delta}$ and $\rho_k < \eta_1 = 0.25$ at $k=4$). Nevertheless, the results for the hybrid algorithm show that scaling of the POD based model function m_k can lead to an improved model a_k. Here, we refer to the values of ρ_k in iterations $k=4,5$ in Table 6.6. While the model m_k at $k=4$ yields $\rho_k = 0.06$, the scaled model function a_k at $k=5$ yields $\rho_k = 1.03$ where it also uses the additional information $\nabla f(u_k)$. In both cases the POD based reduced order model is the same. The final objective function value for the hybrid TRPOD algorithm approach is $f(u_8) = 0.002938$.

In Figure 6.11 we compare the controls computed with the basic TRPOD algorithm and the hybrid TRPOD algorithm with the optimal control computed with the finite element model. The corresponding objective function value for this case is $f(u_{opt}) = 0.002926$.

In Figure 6.12, the corresponding optimal states at $x=0$ are compared to the desired temperature profile, q.

k	$f(u_k)$	$m_k(u_k)$	δ_k	$\|s_k\|$	ρ_k	$\|g_k\|$	ζ_k	M
0	0.173936	0.173936	10.00	9.56	1.00	0.037498	0.0255	3
1	0.003814	0.003836	20.00	1.66	1.08	0.000980	0.1961	3
2	0.003061	0.003160	40.00	0.53	0.18	0.000500	0.5534	3
3	0.003061	0.003160	0.27	0.26	0.49	0.000500	0.5534	3
4	0.003022	0.003087	0.13	0.13	0.06	0.000308	0.8402	3
5	0.003022	0.003087	0.06	0.06	0.27	0.000308	0.8402	3
6	0.003018	0.003074	0.03	0.03	0.24	0.000236	1.1060	3
7	0.003018	0.003074	0.01	0.01	0.25	0.000236	1.1060	3
8	0.003017	0.003072	0.01	0.01	0.26	0.000230	1.1352	3
9	0.003016							

Table 6.4: Basic TRPOD

k	$f(u_k)$	$\sigma_k(u_k)$	δ_k	$\|s_k\|$	ρ_k	$\|\nabla f(u_k)\|$	M
0	0.173936	1.0000	10.00	9.19	1.00	0.037463	3
1	0.005422	0.9624	20.00	2.13	0.99	0.001964	3
2	0.003294	0.9870	40.00	1.30	1.83	0.000636	3
3	0.003013	0.9705	80.00	0.32	0.61	0.000487	3
4	0.002974	0.9810	40.00	0.35	0.65	0.000319	3
5	0.002962	0.9831	20.00	0.10	13.93	0.000288	3
6	0.002945						

Table 6.5: TRPOD with Scaled Model Functions

k	$f(u_k)$	$m_k(u_k)$	δ_k	$\|s_k\|$	ρ_k	$\|g_k\|$	ζ_k	M
0	0.173936	0.173936	10.00	9.56	1.00	0.037498	0.0255	3
1	0.003814	0.003836	20.00	1.66	1.08	0.000980	0.1961	3
2	0.003061	0.003160	40.00	0.53	0.18	0.000500	0.5534	3
3	0.003061	0.003160	0.27	0.26	0.49	0.000500	0.5534	3
4	0.003022	0.003087	0.13	0.13	0.06	0.000308	0.8402	3
	Switch to Scaled Model Functions							
k	$f(u_k)$	$\sigma_k(u_k)$	δ_k	$\|s_k\|$	ρ_k	$\|\nabla f(u_k)\|$		M
5	0.003022	0.9789	0.13	0.13	1.03	0.000235	-	3
6	0.003004	0.9801	0.26	0.25	4.45	0.000219	-	3
7	0.002971	0.9734	0.52	0.49	2.96	0.000146	-	3
8	0.002938							

Table 6.6: Hybrid TRPOD

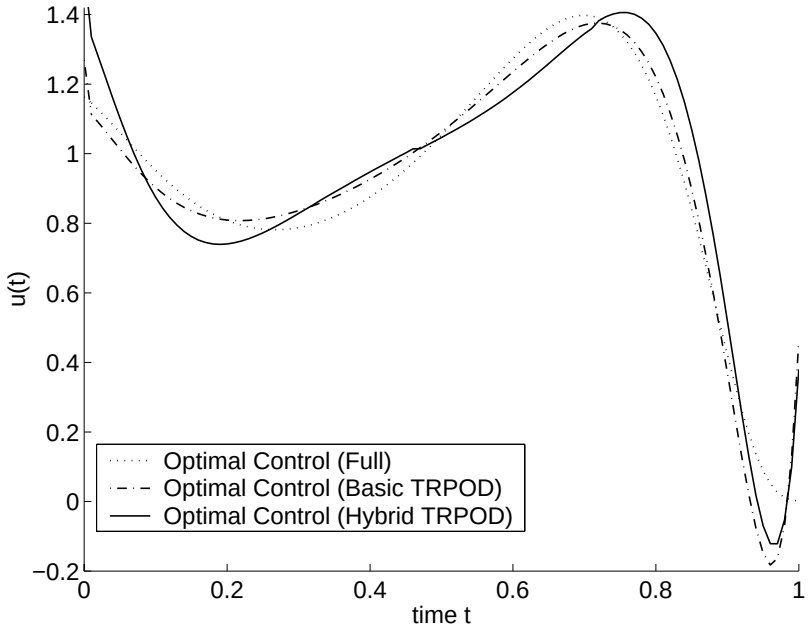

Figure 6.11: Control of Heat Equation: Comparison of Controls

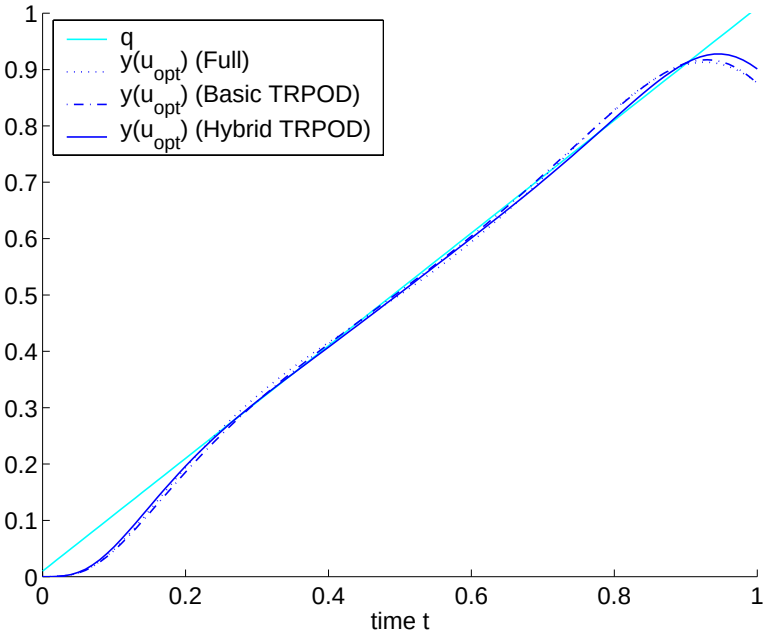

Figure 6.12: Control of Heat Equation: Optimal State at $x = 0$

Chapter 7

An Application in Food Processing

7.1 Introduction

In the previous chapters we presented various aspects of POD based reduced order modelling and the use of these reduced order models during an optimization process. Next, we introduce and discuss an industrial application of this technique in the field of food processing.

One of the important procedures for the sterilization of food products is thermal processing in a batch retort (*autoclave*). The product is filled into containers and these containers are heated in the autoclave by steam or hot water in order to destroy harmful microorganisms. However, not only the microorganisms are heat sensitive. Severe heat treatment during in-container sterilization can produce substantial negative changes in the nutritional and sensory quality of the food.

In practice, the sterilization process can be controlled through the temperature of the autoclave during the processing time. Here, the autoclaves often work with constant temperatures for the heating and cooling phase of the sterilization process. These heating temperatures, and also the holding times at constant temperatures, are chosen empirically such that the sterility requirements on the food product are satisfied in any case. But, this procedure often leads to an unwanted degradation of the quality of the food.

Therefore, research efforts in food industry concentrate on using appropriate mathematical models for this process, such that the competing goals of achieving a safe product and retaining high quality can be optimized.

In the following section, we present an optimal control problem that models the goals of the sterilization process. In order to solve the related optimization problem it is necessary to use a heat transfer model to represent the evolution of the temperature inside the container. In Section 7.3, we present different approaches of various complexity for modelling the heat transfer inside the container. Certainly, an appropriate heat transfer model has to take into account the specific problem formulation under consideration.

But, with increasing complexity of the heat transfer model also the computational complexity for the solution of the corresponding control problem increases. Here, we discuss reduced order modelling ideas that have the potential to be applied in the field of food processing.

7.2 The Optimal Control Problem

In this section, we derive an optimal control problem that models the competing goals of achieving a safe product while also striving for high nutritional quality of the food product. We consider in-container sterilization where prepackaged food is heated in an autoclave, in order to destroy existing microorganisms. In order to formulate an appropriate control problem for the sterilization process, we first derive a condition that models the sterility requirements presecribed by law. Furthermore, we present an objective function modelling the retention of nutrients. We refer to Kleis [69], Kleis/Sachs [70] and Justen [61] where mathematical modelling for the sterilization of prepackaged food and solution concepts for the corresponding control problems are investigated.

In the following, let $\Omega \subset \mathbb{R}^3$ denote the domain of the container, e.g., a can or a glass jar, of the food product, and $\Gamma = \partial\Omega$ its boundary.

The concentration of microorganisms, $C(x,t)$, inside the container can be decribed by a linear differential equation

$$\frac{\partial}{\partial t}C(x,t) = -K(\tau(x,t))\, C(x,t), \quad C(x,0) = C_0(x) \tag{7.1}$$

where $\tau(x,t)$ denotes the absolute temperature at point $x \in \Omega$ at time t, and $C_0(x)$ denotes the initial concentration of microorganisms at $x \in \Omega$. In (7.1), the function K depends on τ and is given by Arrhenius' law (see Kessler [67])

$$K(\tau) = K_{ref} \exp\left(-\frac{E_a}{R}\left(\frac{1}{\tau} - \frac{1}{\tau_{ref}}\right)\right) \tag{7.2}$$

where K_{ref} denotes the value of K at the reference temperature τ_{ref}, R is the universal gas constant and E_a is the activation energy. Here, the values of E_a, τ_{ref} (and K_{ref}) depend on the specific microorganisms that are considered to be most dangerous and that are to be destroyed during the sterilization process.

A sterility condition for food products can be derived by means of the analytic solution of (7.1). Provided that the temperature, τ, is a continuous function, we obtain the solution

$$C(x,T) = C_0(x) \exp\left(-\int_0^T K(\tau(x,t))\, dt\right) \tag{7.3}$$

for $x \in \Omega$ at the final processing time, T.

Since (7.1) describes a reduction process that depends on the temperature distribution inside the container, we note that at a given point $\bar{x} \in \Omega$, according to (7.2) the value of

$-K(\tau(\bar{x},t))$ in (7.3) decreases, if $\tau(\bar{x},t)$ increases at that location. Therefore, it is possible to restrict the analysis of the reduction of microorganisms to a single point $x_c \in \Omega$, if x_c represents a worst-case situation. We denote by x_c the coldest point inside the container which is often assumed to be located in the geometrical center of the container. For a discussion of this assumption, we refer to Kleis [69].

Inserting $K(\tau)$ given by (7.2) at x_c in (7.3) and using the relation $10^z = e^{z \ln 10}$ (for given $z \in \mathbb{R}$) yields

$$\begin{aligned} C(x_c,T) &= C_0(x_c) \exp\left(- \int_0^T K(\tau(x_c,t))\, dt \right) \\ &= C_0(x_c) \exp\left(- \int_0^T K_{ref} 10^{-\frac{E_a}{R \ln 10}\left(\frac{1}{\tau(x_c,t)} - \frac{1}{\tau_{ref}}\right)}\, dt \right) \\ &= C_0(x_c) \exp\left(- \int_0^T K_{ref} 10^{\frac{\tau(x_c,t)-\tau_{ref}}{z(\tau(x_c,t))}}\, dt \right) \end{aligned} \tag{7.4}$$

where $z(\tau(x_c,t))$ is defined by, see [70],

$$z(\tau(x_c,t)) = \tfrac{R}{E_a} \tau_{ref}\, \tau(x_c,t)\, \ln 10. \tag{7.5}$$

Equation (7.4) leads to the following definition of the sterility constraint which is concerned with the *bacteriological effect* of the thermal processing. A product satisfies the sterility requirements prescribed by law, if for the total processing time, T, the concentration of microorganisms, $C(x_c,t)$, at the coldest point, x_c, is reduced by a certain factor with respect to the initial concentration (see Kessler [67])

$$\frac{C(x_c,T)}{C_0(x_c)} = \exp\left(- \int_0^T K_{ref} 10^{\frac{\tau(x_c,t)-\tau_{ref}}{z_{ref}}}\, dt \right) \;\leq\; 10^{-\beta_{law}}. \tag{7.6}$$

Here, β_{law} is a given reference value for the microorganisms under consideration. Furthermore, we note that $z(\tau)$ according to (7.5) has been replaced by $z_{ref} = (R/E_a)\tau_{ref}^2 \ln 10$ in (7.4) in order to obtain the sterility condition (7.6).

We note that in the food research literature, e.g., see Ball/Olson [11] or Kessler [67], and in food industry, condition (7.6) is often reformulated in order to use the *F-value* concept. By defining

$$F(\tau) = \int_0^T 10^{\frac{\tau(x_c,t)-\tau_{ref}}{z_{ref}}}\, dt \tag{7.7}$$

it is easy to see that (7.6) is equivalent to

$$F(\tau) \geq \frac{\beta_{law}}{K_{ref}} \ln 10. \tag{7.8}$$

Since the differential equation in (7.1) describes a general temperature dependent reduction process, an expression similar to the left hand side of inequality (7.6) can be

derived that describes the destruction of nutrients. Merely, suitable reference values for the parameters in the Arrhenius equation (7.2) have to be chosen.

Therefore, if we aim at maximizing the concentration of nutrients, a suitable measure for the nutritional quality of a food product is given by

$$\tilde{J}(\tau) = \int_0^T 10^{\frac{\tau(x_b,t)-\tau_q}{z_q}}\, dt, \tag{7.9}$$

where τ_q, z_q denote reference values for the specific *chemical effect* of the thermal processing. Here, for the analysis of the retention of nutrients we focus on a point x_b at the boundary of the container, because this is affected most during the heating phase.

According to the above discussion, for the computation of the bacteriological and chemical effects it is necessary to know the temperature distribution inside the container. If the temperature is known, the F-value (at x_c) according to (7.7) as well as $\tilde{J}(\tau)$ (at x_b) given by (7.9) can be computed.

Let $u(t)$, $t \in (0,T)$, denote the temperature of the autoclave, where the containers are heated, during the thermal processing. Thus, with the cost functional

$$J(\tau, u) = \tilde{J}(\tau) + \frac{\gamma}{2}\int_0^T u(t)^2\, dt$$

for some $\gamma > 0$ we can formulate the optimal control problem

$$\text{Minimize} \quad J(\tau, u) \tag{7.10}$$

subject to the constraints

$$F(\tau) \geq \frac{\beta_{law}}{K_{ref}} \ln 10 \tag{7.11}$$

and

$$u_{min} \leq u(t) \leq u_{max}(t) \qquad t \in [0,T]. \tag{7.12}$$

Here, $\tau = \tau(u)$ is the temperature distribution inside the container that corresponds to the control u. The last term in the cost functional represents an energy term. The constraints on the control u are physically given upper and lower bounds for the temperature of the autoclave, see Kleis/Sachs [70].

So far, we did not specify a heat transfer model that serves to determine the temperature evolution inside the food container corresponding to temperature changes of the heating medium in the autoclave. This is the topic of the next section.

7.3 Modelling the Heat Transfer

As we have seen from the derivation of the optimal control problem, it is important to have a reasonable model for describing the development of the temperature inside the food container. In the following, we will give a review of some heat transfer models of various complexity that can be used in the context of sterilization of prepackaged food.

7.3.1 The Ball-Formula

In food engineering practice the *Ball-Formula*, originating from the 1960s, is still very popular (see Ball/Olson [11]). Depending on two parameters only, it provides a simple approach for modelling heat transfer in food products empirically.

Let $u(t) = \bar{u}$ for all $t \in (0, T)$ denote a constant and uniform autoclave temperature and τ_0 an initial uniform temperature distribution of the product inside the container, such that $\tau(x_c, t) = \tau_0$ and $\bar{u} > \tau_0$ holds.

Considering the scaled temperature difference

$$\tilde{\tau}(x_c, t) = \frac{\bar{u} - \tau(x_c, t)}{\bar{u} - \tau_0} \tag{7.13}$$

the Ball-Formula for determining $\tilde{\tau}(x_c, t)$ reads

$$\tilde{\tau}(x_c, t) = j\, e^{-(t/f)\ln 10}. \tag{7.14}$$

For heating processes, the parameters j, f in (7.14) are strictly positive and depend, e.g., on the geometry of the food container and some other material constants of the container and the product to be heated. According to (7.13) the absolute temperature, $\tau(x_c, t)$, is given by

$$\tau(x_c, t) = j\,(\tau_0 - \bar{u}) e^{-(t/f)\ln 10} + \bar{u}. \tag{7.15}$$

Consequently, for given parameters j, f and an initial temperature difference between product and autoclave temperature, $\bar{u} - \tau_0 > 0$, the absolute temperature $\tau(x_c, t)$ approaches the autoclave temperature exponentially from below.

In industrial practice, the parameters j, f are determined empirically, e.g., by fitting measured data for a specific type of product. Based on these parameters, formula (7.14) can be used for the calculation of the temperature development during the heating phase at a given point x_c for this specific type of product.

We remark, that the Ball-Formula according to (7.14) leads to reliable results for sufficiently large time values t only, see Ball/Olson [11]. Furthermore, it neglects any spatial effects which are important to be modelled in heat conduction processes. Therefore, (7.14) cannot be used for modelling cooling processes, or in general, changes in the autoclave temperature. The problem is an initial lag in the cooling curve, since the conductivity of the process cannot be modelled. To compensate this lag portion, this part of the curve has to be approximated by a hyperbola if cooling phases of the autoclave are considered. Therefore, the Ball-Formula approach does not seem to be suited for appropriately modelling heat transfer in a control problem formulation of the food sterilization process.

At this point, we note that the Ball-Formula can be derived by truncating a Fourier series representation of the solution of a conductive heat equation after the first term. This approach has been analyzed in [34] and is currently under further investigations. Using the Fourier series representation it seems to be possible to compensate the deficiencies of the Ball-Formula method by including additional terms of the series expansion.

7.3.2 A Nonlinear Heat Equation

The heat transfer from the surrounding heating medium in the autoclave to a homogeneous and solid food product can be generally modelled by a nonlinear heat equation

$$\rho c(\tau)\tfrac{\partial}{\partial t}\tau - \nabla \cdot (k(\tau)\nabla\tau) = 0 \quad \text{in } \Omega \times (0,T) \tag{7.16}$$

with initial condition

$$\tau(\,\cdot\,,0) = \tau_0(\,\cdot\,) \quad \text{in } \Omega \tag{7.17}$$

and boundary condition

$$k(\tau)\tfrac{\partial}{\partial n}\tau = \alpha(u-\tau) \quad \text{on } \Gamma \times (0,T). \tag{7.18}$$

In (7.16)–(7.18), ρ is the density, $c(\tau)$ the heat capacity and $k(\tau)$ the heat conductivity of the product to be heated. The heat transfer coefficient is denoted by α. We note that, in general, the heat capacity, c, and the heat conductivity, k, are functions of the temperature, τ.

In order to derive the boundary condition (7.18) it is assumed that there is a uniform temperature in the autoclave such that the heat flux into the food container is proportional to the temperature difference between the container and the heating medium, see Kleis/Sachs [70].

The advantage of (7.16)–(7.18) for modelling the temperature development inside a food container is that temperature dependent material constants can be included explicitly. Therefore, this model is another step into a more precise formulation for the food sterilization problem.

The optimal control of sterilization of prepackaged food based on (7.16)–(7.18) has been analyzed and numerically solved in Kleis [69], Kleis/Sachs [70]. There, a finite element discretization has been employed for the discretization of the heat equation.

7.3.3 Heat Transfer Models involving the Navier–Stokes Equations

There are also more complex approaches for modelling the temperature inside a food container.

For example, a heat transfer model for homogeneous, liquid food products is given by a coupled system of the heat equation and the Navier–Stokes equations. In this case, buoyancy effects due to temperature differences have to be included (*Boussinesq approximation*). The following approach to the sterilization of liquid food products has been analyzed in Justen [61]. We also refer to Abergel/Temam [1] where a similar problem in the context of flow control by boundary temperature control for the driven cavity is analyzed.

In [61] a heat transfer model similar to (7.16)–(7.18) is considered that also includes temperature changes inside the container due to buoyancy effects

$$\rho c(\tau)\tfrac{\partial}{\partial t}\tau - \nabla\cdot(k(\tau)\nabla\tau) + \rho c(\tau)(y\cdot\nabla)\tau = 0 \quad \text{in } \Omega\times(0,T), \tag{7.19}$$

$$\tau(\,\cdot\,,0) = \tau_0(\,\cdot\,) \quad \text{in } \Omega, \tag{7.20}$$

$$k(\tau)\tfrac{\partial}{\partial n}\tau = \alpha(u-\tau) \quad \text{on } \Gamma\times(0,T). \tag{7.21}$$

The velocity field y in (7.19) is given by the Navier–Stokes equations

$$\tfrac{\partial}{\partial t}y - \nu\Delta y + (y\cdot\nabla)y + \nabla p = f^{be}(\tau) \quad \text{in } \Omega\times(0,T), \tag{7.22}$$

$$\operatorname{div} y = 0 \quad \text{in } \Omega\times(0,T) \tag{7.23}$$

with homogeneous Dirichlet boundary conditions, where $f^{be}(\tau)$ in (7.22) denotes body forces due to buoyancy effects.

Furthermore, due to the specific application under consideration, Justen [61] considered two simplifications of the coupled system (7.19)–(7.21), (7.22)–(7.23).

First, the nonlinear convection–diffusion equation (7.19) is substituted by a linear convection–diffusion equation by assuming that the heat capacity and heat conductivity are constant and do not depend on the temperature, τ.

Second, the buoyancy effects in (7.22) are neglected due to the fact that the food sterilization is performed in an autoclave with additional rotation of the food containers.

Following this idea we obtain the new system of partial differential equations (see Justen [61])

$$\tfrac{\partial}{\partial t}\tau - \kappa\Delta\tau + (y\cdot\nabla)\tau = 0 \quad \text{in } \Omega\times(0,T), \tag{7.24}$$

$$\tau(\,\cdot\,,0) = \tau_0(\,\cdot\,) \quad \text{in } \Omega, \tag{7.25}$$

$$\tfrac{\partial}{\partial n}\tau = \tilde{\alpha}(u-\tau) \quad \text{on } \Gamma\times(0,T), \tag{7.26}$$

$$\tfrac{\partial}{\partial t}y - \nu\Delta y + (y\cdot\nabla)y + \nabla p = f^{rot}(r) \quad \text{in } \Omega\times(0,T), \tag{7.27}$$

$$\operatorname{div} y = 0 \quad \text{in } \Omega\times(0,T), \tag{7.28}$$

$$y(\,\cdot\,,0) = y_0(\,\cdot\,) \quad \text{in } \Omega, \tag{7.29}$$

$$y = 0 \quad \text{on } \Gamma\times(0,T). \tag{7.30}$$

Due to the assumption that the heat capacity and heat conductivity are constant we have $\kappa = k/(c\rho)$ and $\tilde{\alpha} = \alpha/k$ in (7.24) and (7.26), cf. also (7.16)–(7.18). In (7.27), the body forces f^{rot} represent centrifugal and gravitational forces depending on the rotational speed, $r\in\mathbb{R}$, that is usually measured in rotations per minute. The momentum equation in the Navier–Stokes system is independent of the temperature τ, since it is expected that the effect of the rotation dominates the buoyancy effects. Furthermore, in [61] the

rotational speed r is used as an additional control variable in an appropriate control problem formulation in the sense of (7.10).

The new system (7.24)–(7.26), (7.27)–(7.30) has an important advantage compared to the coupled system (7.19)–(7.21), (7.22)–(7.23), because it includes a decoupling in the computation of y and τ. First, it is possible to compute a flow field, y, for a given rotational speed, $r \in \mathbb{R}$. Then, for a given flow field, y, the convection-diffusion equation (7.24)–(7.26) can be solved. Here, the control u, i.e. the temperature of the autoclave, enters in the boundary conditions (7.26).

Finally, we note that the numerical computations for the sterilization of food products in [61] are based on finite element methods for the convection-diffusion equation (7.24)–(7.26) with prescribed flow fields only. Incorporating a solution procedure for the Navier–Stokes equations during the solution of the control problem was not workable, because of computational complexity reasons.

7.4 Starting Points for Reduced Order Modelling

Based on the various heat transfer models presented in the previous section, we will outline a POD based reduced order modelling approach in the context of food sterilization.

The corresponding reduced order models can be applied during the solution of the optimal control problem (7.10)–(7.12). Consequently, according to the discussion in Chapter 6, a TRPOD-like algorithm should be applied for the solution of the corresponding optimization problem.

According to the presentation in Section 7.3, an accurate finite element discretization of the nonlinear heat equation as well as the heat equation coupled with the Navier–Stokes equations leads to large-scale discretized optimization problems, cf. Kleis/Sachs [70]. In order to reduce the computational complexity for the solution of these optimization problems, it is possible to employ reduced order modelling techniques based on POD.

At this point, we note that POD based reduced order models for the heat equation have been investigated, e.g., in Atwell/King [9, 10] or Diwoky/Volkwein [30]. Similar reduced order models can be employed for modelling the sterilization of solid food products according to Subsection 7.3.2, see also [34].

Furthermore, POD based reduced order models for steady-state convection–diffusion equations with prescribed flow fields are considered in Park/Lee [86]. We also refer to Park/Cho [84] and Ly/Tran [78] were POD methods for CVD reactors are studied. In these papers, the flow fields for a convection-diffusion equation are obtained by solving the steady-state Navier–Stokes equations for compressible or incompressible flows, and reduced order models for the temperature evolution are derived.

Since we primarily focussed on flow control involving reduced order models for the time-dependent Navier–Stokes equations so far, we outline a POD based reduced order

modelling approach for the sterilization of liquid food products according to the material in Subsection 7.3.3. As pointed out there, in [61] it was not workable to solve the complex heat transfer model with rotational forces because of the computational complexity of this problem.

A remedy is given by substituting a finite element model for the Navier–Stokes equations by a POD based reduced order model according to Section 3.5. In this case, it is possible to solve the convection–diffusion equation

$$\frac{\partial}{\partial t}\tau - \kappa\Delta\tau + (y\cdot\nabla)\tau = 0 \quad \text{in } \Omega\times(0,T), \tag{7.31}$$

with initial condition (7.25) and boundary conditions (7.26) as it was done in [61]. But, the flow field, y, is approximated by a reduced order solution, y^M, instead of a finite element solution. The corresponding reduced order solution, y^M, is given by solving a POD based reduced order model of the type

$$\begin{aligned} \dot{\mathbf{y}}^M(t) + \nu\mathcal{A}\,\mathbf{y}^M(t) + \mathcal{N}(\mathbf{y}^M,\mathbf{y}^M) &= \mathcal{F}(r,t) \qquad \text{for all } t\in(0,T) \\ \mathbf{y}^M(0) &= \mathbf{y}_0^M \end{aligned}$$

for given control $r\in\mathbb{R}$, such that

$$y^M(x,t) = \sum_{j=1}^{M} \mathbf{y}_j^M(t)\,\psi_j(x). \tag{7.32}$$

This approach allows an efficient computation of an approximate flow field, if a corresponding reduced order model is available.

Besides this, in a finite element discretization of (7.31) the reduced order modelling approach yields an additional simplification. In order to illustrate this aspect, we consider a finite element discretization of the convection–diffusion equation (7.31)

$$\mathcal{M}\,\dot{\boldsymbol{\tau}}^N(t) + \kappa\,\mathcal{S}\boldsymbol{\tau}^N(t) + \mathcal{C}(y^N)\boldsymbol{\tau}^N(t) = 0 \tag{7.33}$$

for $t\in(0,T)$, neglecting boundary and initial conditions. Here, $\mathcal{M}$, $\mathcal{S}$ denote mass and stiffness matrix as usual, and $\mathcal{C}(y^N)\boldsymbol{\tau}^N(t)$ corresponds to the term $(y\cdot\nabla)\,\tau$ for a given flow field $y^N(x,t)$, $x\in\Omega$, $t\in(0,T)$, see Justen [61].

The computationally most expensive part during the solution of (7.33) is that terms of the type $\mathcal{C}(y_i^N)$ with $y_i^N(x) = y^N(x,t_i)$ have to be recomputed at every time step t_i. By employing the representation (7.32) the corresponding term in (7.33) for a POD based reduced order solution reads

$$\mathcal{C}(y^M) = \sum_{j=1}^{M} \mathbf{y}_j^M(t)\,\mathcal{C}(\psi_j). \tag{7.34}$$

Thus, expression (7.34) can be easily evaluated at time t_i, if the corresponding terms $\mathcal{C}(\psi_j)$, $j=1,\ldots,M$, have been computed and the reduced order solution $y^M(x,t)$ given by (7.32) is available.

For this reason, POD based reduced order modelling for the Navier–Stokes equations in the heat transfer model (7.24)–(7.26), (7.27)–(7.30) for liquid food products, can lead to heat transfer models of workable computational complexity. These can be used for the computation of controls for the sterilization of prepackaged food in an autoclave.

We are aware of the fact that the numerical implementation of the proposed solution concepts is not an easy task. It still requires a lot of research efforts and has not been completely done, yet. Some preliminary numerical results have been given in [34], see also the TRPOD results for the heat equation in Section 6.4.2.

However, the numerical results concerning the control of the Navier–Stokes equations in Chapter 6 indicate that POD based reduced order modelling provides an approach to efficient solution concepts for complex heat transfer models for optimal control problems in food processing.

Conclusions

The discretization of optimal control problems governed by partial differential equations typically leads to large-scale optimization problems. We consider flow control involving the time-dependent Navier–Stokes equations as state equation which is stamped by exactly this property.

In order to avoid the difficulties of dealing with large-scale (discretized) state equations during the optimization process, a reduction of the number of state variables can be achieved by employing a reduced order modelling technique. Using the snapshot proper orthogonal decomposition method, one obtains a low-dimensional model for the computation of an approximate solution to the state equation. In fact, often a small number of POD basis functions suffices to obtain a satisfactory level of accuracy in the reduced order solution. However, the small number of degrees of freedom in a POD based reduced order model also constitutes its main weakness for optimal control purposes. Since a single reduced order model is based on the solution of the Navier–Stokes equations for a specified control, it might be an inadequate model when the control (and consequently also the actual corresponding flow behaviour) is altered, implying that the range of validity of a reduced order model, in general, is limited. Thus, it is likely to meet unreliable reduced order solutions during a control problem solution based on one single reduced order model.

In order to get out of this dilemma, we propose to use a trust-region proper orthogonal decomposition (TRPOD) approach. By embedding the POD based reduced order modelling technique into a trust-region framework with general model functions, we obtain a mechanism for updating the reduced order models during the optimization process, enabling the reduced order models to represent the flow dynamics as altered by the control. In fact, a rigorous convergence theory for the TRPOD method is obtained which justifies this procedure also from a theoretical point of view.

Benefiting from the trust-region philosophy, the TRPOD method guarantees to save a lot of computational work during the control problem solution, since the original state equation only has to be solved if we intend to update our model function in the trust-region framework. The optimization process itself is completely based on reduced order information only.

Appendix A

The Flow Solver FEATFLOW

The starting point for the POD based reduced order modelling approach with snapshot input data is the availability of a solver for the partial differential equation under consideration.

Since we are interested in optimal control problems governed by the Navier-Stokes equations an appropriate flow solver is required. The numerical results of this thesis are based on flow computations with the finite element software FEATFLOW[1] [113]. This appendix serves to introduce some aspects of the solution techniques of the software package that have been used for our applications, and that can be seen as typical ingredients of a finite element flow solver. We follow the description in Turek [113, 115] and the references therein.

As pointed out in Chapter 2, for the solution of the Navier-Stokes equations special attention has to be paid to the treatment of the nonlinearity in the momentum equation

$$\frac{\partial}{\partial t} y - \nu \, \Delta y + (y \cdot \nabla) \, y + \nabla p = \mathtt{f} \qquad \text{in } \Omega \times (0,T)$$

and the treatment of the incompressibility condition

$$div \, y = 0 \qquad \text{in } \Omega \times (0,T)$$

where a coupling between the velocity field, y, and the pressure, p, is caused by the latter. For the sake of simplicity, we consider homogeneous Dirichlet boundary conditions, i.e. $y|_\Gamma = 0$ for all $t \in (0,T)$, in the following, and we denote $y|_{t=0} = y_0$.

Our presentation of the solution concepts here addresses the spatial discretization via finite elements in Section A.1, the time discretization scheme in Section A.2 and the decoupling of velocity and pressure variables in Section A.3. If these solution approaches are applied to the Navier–Stokes equations it is finally necessary to have efficient solution procedures for systems of linear and nonlinear equations. In Section A.4 and Section A.5 we address the corresponding techniques that are implemented in the FEATFLOW software.

[1] Information on FEATFLOW can also be found under *http://www.featflow.de*.

A.1 The Finite Element Discretization

Starting from the variational formulation (II) given in Problem 2.4, we consider a spatial discretization of

$$\begin{aligned} \tfrac{\partial}{\partial t}(y,\phi)_{L^2} + \nu\,\mathbf{a}(y,\phi) + \mathbf{c}(y,y,\phi) + \mathbf{b}(\phi,p) &= (f,\phi)_{L^2}, \\ \mathbf{b}(y,\psi) &= 0, \\ y(0) &= y_0, \end{aligned}$$

for $\phi \in H_0^1(\Omega)$, $\psi \in L_0^2(\Omega)$, by means of a finite element approach. Here, we only address the FEATFLOW specific choice of the finite element discretization for this variational formulation that differs from approaches that usually can be found in the literature, e.g., see Cuvelier et al. [26], Girault/Raviart [40] or Thomasset [110].

FEATFLOW employs quadrilateral elements with piecewise rotated bilinear shape functions for the velocity approximation and piecewise constant shape functions for the pressure approximation. This $\tilde{Q}_1/Q_0$–Element has been introduced and analyzed in Rannacher/Turek [94].

The most characteristic property of the rotated bilinear shape function for the velocity approximation is that a small number of degrees of freedom is retained while the shape functions include aspects of quadratic shape functions. In order to illustrate this idea, we consider a partition of the domain Ω into disjoint quadrilateral elements $\mathbf{Q}$ where each $\mathbf{Q}$ is determined by its four vertices v_1, v_2, v_3, v_4. For computational purposes, each $\mathbf{Q}$ is usually transformed to the unit square $\hat{\mathbf{Q}}_{[0,1]} = [0,1]\times[0,1]$, that serves as a reference element.

We recall that a bilinear function $\phi^{(lin)}$ on $\hat{\mathbf{Q}}_{[0,1]}$ is determined by the values of

$$\phi^{(lin)}(\hat{x}_1,\hat{x}_2) = w_0 + w_1\,\hat{x}_1 + w_2\,\hat{x}_2 + w_3\,\hat{x}_1\,\hat{x}_2$$

at the nodal points at the four vertices $\hat{v}_1 = (0,0)^T$, $\hat{v}_2 = (1,0)^T$, $\hat{v}_3 = (1,1)^T$, $\hat{v}_4 = (0,1)^T$ of the reference element for suitable weights w_j, $j=0,\ldots,3$. Here, $\hat{x}_1$, $\hat{x}_2$ denote the local coordinates for $\hat{\mathbf{Q}}_{[0,1]}$. Similarly, a biquadratic function on $\hat{\mathbf{Q}}_{[0,1]}$

$$\begin{aligned} \phi^{(quad)}(\hat{x}_1,\hat{x}_2) &= w_0 + w_1\,\hat{x}_1 + w_2\,\hat{x}_2 + w_3\,\hat{x}_1\,\hat{x}_2 + w_4\,\hat{x}_1^2 \\ &\quad + w_5\,\hat{x}_2^2 + w_6\,\hat{x}_1^2\,\hat{x}_2 + w_7\,\hat{x}_1\,\hat{x}_2^2 + w_8\,\hat{x}_1^2\,\hat{x}_2^2 \end{aligned}$$

is determined by its values at 9 nodal points, i.e. the 4 vertices, the 4 midpoints of the sides and the center of the reference element, for suitable weights w_j, $j=0,\ldots,8$, cf. e.g. Quarteroni/Valli [91].

The $\tilde{Q}_1/Q_0$–Element of Rannacher/Turek [94] uses 4 nodal points for the velocity approximation like in the bilinear case. But, its reference element is $\hat{\mathbf{Q}}_{[-1,1]} = [-1,1]\times[-1,1]$. The nodal points $\hat{v}_1 = (-1,0)^T$, $\hat{v}_2 = (0,1)^T$, $\hat{v}_3 = (1,0)^T$, $\hat{v}_4 = (0,-1)^T$ are placed

at the midpoints of the element sides. Nevertheless, we keep the notation $\hat{v}_j$ although the nodal points are no longer located in the vertices of the reference element $\hat{\mathbf{Q}}_{[-1,1]}$ (see Figure A.1).

In general, the rotated bilinear shape functions on $\hat{\mathbf{Q}}_{[-1,1]}$ can be represented as

$$\phi^{(rot)}(\hat{x}_1, \hat{x}_2) = w_0 + w_1\,\hat{x}_1 + w_2\,\hat{x}_2 + w_3\,(\hat{x}_1^2 - \hat{x}_2^2)$$

with suitable weights w_j, $j=0,\ldots,3$ and local coordinates $\hat{x}_1, \hat{x}_2$ for $\hat{\mathbf{Q}}_{[-1,1]}$. Consequently, these local shape function $\phi^{(rot)}$ also contain quadratic terms. Specifically, we obtain

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

such that $\phi_i^{(rot)}(\hat{v}_j) = \delta_{ij}$ for $i, j = 1, \ldots, 4$ holds.

We finally note that the transformation $\Phi : \hat{\mathbf{Q}}_{[-1,1]} \to \mathbf{Q}$ that maps the reference element $\hat{\mathbf{Q}}_{[-1,1]}$ to $\mathbf{Q}$ is given by

$$\Phi(\hat{x}_1, \hat{x}_2) = v_1 + \frac{1}{2}(v_2 + v_4 - 2v_1)(\hat{x}_1 + 1) + \frac{1}{2}(v_2 - v_4)\hat{x}_2.$$

Thus, we have $\Phi(\hat{v}_j) = v_j$, $j = 1, \ldots, 4$.

As pointed out above, FEATFLOW employs piecewise constant shape functions with one nodal point in the center of the element for the pressure approximation. The combination of piecewise rotated bilinear shape functions for the velocity and piecewise constant shape functions for the pressure leads to a stable numerical approximation of the velocity and pressure fields, in the sense that this finite element pair satisfies the *Babuska-Brezzi*

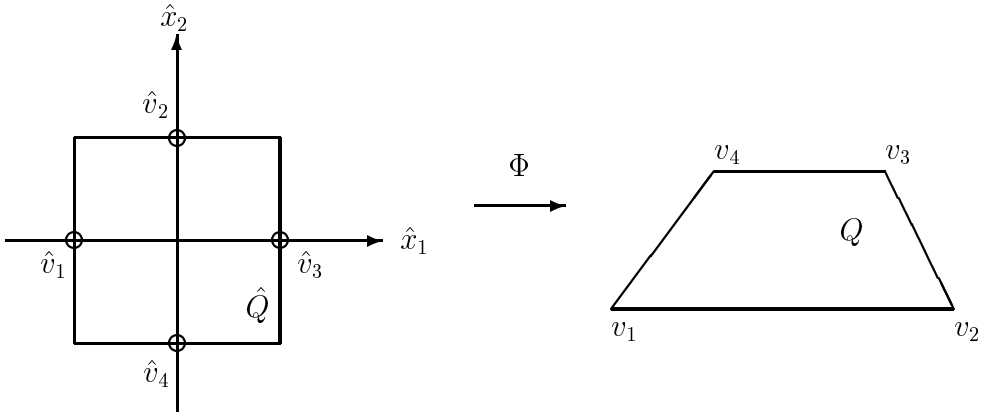

Figure A.1: Transformation from $\hat{\mathbf{Q}}_{[-1,1]}$ to $\mathbf{Q}$

condition (or *inf-sup condition*) with a mesh-independent constant which is not satisfied for other first-order elements ($Q1/Q0$, $Q1/Q1$).

Based on the finite element discretization of the variational formulation (II) we obtain a large-scale system of ordinary differential equations

$$\mathcal{M}\dot{\mathbf{y}}^N(t) + \nu\mathcal{S}\mathbf{y}^N(t) + \mathcal{C}(\mathbf{y}^N(t))\mathbf{y}^N(t) + \mathcal{B}\mathbf{p}^N(t) = \mathcal{F}(t) \quad (A.1)$$

$$\mathcal{B}^T\mathbf{y}^N(t) = 0 \quad (A.2)$$

for all $t \in (0,T)$ with initial value $\mathbf{y}^N(0) = \mathbf{y}_0^N$, where $\mathcal{M}$ denotes the finite element mass matrix, and $\mathcal{S}$ the stiffness matrix due to the discretization of the bilinear form $\mathbf{a}(\cdot,\cdot)$. The term $\mathcal{C}(\cdot)$ corresponds to the discretization of the trilinear form $\mathbf{c}(\cdot,\cdot,\cdot)$, $-\mathcal{B}^T$ denotes the divergence matrix and $\mathcal{F}(t)$ the discretized body forces of the Navier–Stokes equations. The bold face characters with superscript N denote the vectors of expansion coefficients of the velocity and pressure field solutions in the finite element basis representation.

A.2 The Time Discretization Scheme

The FEATFLOW solution scheme that handles the numerical difficulties concerning the nonlinearity and coupling of velocity and pressure variables is strongly linked with the time discretization of the Navier–Stokes equations.

Let $\mathbf{y}_i = \mathbf{y}(t_i)$, $\mathbf{p}_i = \mathbf{p}(t_i)$ denote current solutions for the velocity and pressure, respectively, at time t_i according to (A.1) and (A.2). For ease of notation we again drop the superscript N that indicates finite element approximations.

The solutions $\mathbf{y}_{i+1}$, $\mathbf{p}_{i+1}$ at time $t_{i+1} = t_i + \delta t$ with time step δt can be computed using a standard time stepping scheme. Omitting the boundary conditions, we then have

$$\begin{aligned}\mathcal{M}\mathbf{y}_{i+1} + \theta\delta t(\nu\,\mathcal{S}\mathbf{y}_{i+1} + \mathcal{C}(\mathbf{y}_{i+1})\mathbf{y}_{i+1} + \mathcal{B}\mathbf{p}_{i+1}) &= \theta\delta t\,\mathcal{F}_{i+1} + (1-\theta)\delta t\,\mathcal{F}_i + \mathcal{M}\mathbf{y}_i \\ &\quad - (1-\theta)\delta t\,(\nu\,\mathcal{S}\mathbf{y}_i + \mathcal{C}(\mathbf{y}_i)\mathbf{y}_i + \mathcal{B}\mathbf{p}_i) \\ \mathcal{B}^T\mathbf{y}_{i+1} &= 0\end{aligned}$$

where $\theta = 1$ yields the Backward-Euler scheme and $\theta = 0.5$ the Crank-Nicolson scheme which possess well-known accuracy and stability properties, see e.g. Lambert [74].

To combine both second order accuracy and favourable stability properties, a modification of the above time stepping scheme is incorporated in the FEATFLOW software package. The time step $\delta t = t_{i+1} - t_i$ is treated as a macro time step and three time substeps of different length are defined that lead to similar subproblems but with different weights (*fractional-step-θ-method*). Note that in Turek [114] the term fractional-step-θ-method is used to denote a mere time-stepping technique which is in contrast to operator splitting approaches (see Section A.3) that are also connected with this term, cf. e.g. Glowinski [41], Kloucek/Rys [71].

For

$$\theta = 1 - \sqrt{2}/2, \quad \theta' = 1 - 2\theta, \quad \text{and} \quad \tilde{\theta} = \alpha\theta\delta t$$

with

$$\alpha = (1 - 2\theta)/(1 - \theta), \quad \beta = 1 - \alpha$$

the following three problems have to be solved successively where we set

$$N(\mathbf{y}_i)\mathbf{y}_i = \nu\,\mathcal{S}\,\mathbf{y}_i + \mathcal{C}(\mathbf{y}_i)\,\mathbf{y}_i$$

(for given $\mathbf{y}_i$) for notational convenience. The first problem to be solved for $\mathbf{y}_{i+\theta}$, $\mathbf{p}_{i+\theta}$ is

$$\begin{aligned} [\mathcal{M} + \tilde{\theta}N(\mathbf{y}_{i+\theta})]\mathbf{y}_{i+\theta} + \theta\delta t\mathcal{B}\,\mathbf{p}_{i+\theta} &= [\mathcal{M} - \beta\theta\delta t N(\mathbf{y}_i)]\mathbf{y}_i + \theta\delta t\mathcal{F}_i \\ \mathcal{B}^T\mathbf{y}_{i+\theta} &= 0. \end{aligned}$$

Second, the problem

$$\begin{aligned} [\mathcal{M} + \tilde{\theta}N(\mathbf{y}_{i+1-\theta})]\mathbf{y}_{i+1-\theta} + \theta'\delta t\mathcal{B}\,\mathbf{p}_{i+1-\theta} &= [\mathcal{M} - \alpha\theta'\delta t N(\mathbf{y}_{i+\theta})]\mathbf{y}_{i+\theta} + \theta'\delta t\mathcal{F}_{i+1-\theta} \\ \mathcal{B}^T\mathbf{y}_{i+1-\theta} &= 0 \end{aligned}$$

has to be solved for $\mathbf{y}_{i+1-\theta}$, $\mathbf{p}_{i+1-\theta}$. Finally, the macro time step is completed by solving

$$\begin{aligned} [\mathcal{M} + \tilde{\theta}N(\mathbf{y}_{i+1})]\mathbf{y}_{i+1} + \theta\delta t\mathcal{B}\,\mathbf{p}_{i+1} &= [\mathcal{M} - \beta\theta\delta t N(\mathbf{y}_{i+1-\theta})]\mathbf{y}_{i+1-\theta} + \theta\delta t\mathcal{F}_{i+1-\theta} \\ \mathcal{B}^T\mathbf{y}_{i+1} &= 0 \end{aligned}$$

for $\mathbf{y}_{i+1}$, $\mathbf{p}_{i+1}$.

We note that in all cases the pressure is treated fully implicit and the body forces either fully implicit or fully explicit. The parameters α, β describe weighting parameters similar to θ in the standard time stepping scheme. With time substeps $\delta t_1 = \delta t_3 = \theta\delta t \approx 0.293\delta t$, $\delta t_2 = \theta'\delta t \approx 0.414\delta t$ and $\alpha \approx 0.586$, $\beta \approx 0.414$ a stable and second order accurate time discretization scheme is obtained, see Müller-Urbaniak [81], Rannacher [93]. FEATFLOW also employs an adaptive time step control for the determination of the next macro time step.

Following the above approach, for a given velocity field, $\mathbf{y}_i$, time step size δt and appropriate constants θ_1, θ_2, θ_3, θ_4 problems of the type

$$[\mathcal{M} + \theta_1\delta t N(\mathbf{y})]\mathbf{y} + \delta t\mathcal{B}\,\mathbf{p} = [\mathcal{M} - \theta_2\delta t N(\mathbf{y}_i)]\mathbf{y}_i + \theta_3\delta t\mathcal{F}_{i+1} + \theta_4\delta t\mathcal{F}_i \qquad \text{(A.3)}$$

$$\mathcal{B}^T\mathbf{y} = 0 \qquad \text{(A.4)}$$

have to be solved in order to compute $\mathbf{y}=\mathbf{y}_{i+1}$, $\mathbf{p}=\mathbf{p}_{i+1}$. At this, we note that

$$\mathcal{M} + \theta_1\delta t N(\cdot) = \mathcal{M} + E \qquad \text{(A.5)}$$

can be interpreted as a perturbation, $\mathcal{M}+E$, of the mass matrix, $\mathcal{M}$, for sufficiently small time steps δt.

A.3 Decoupling of Velocity and Pressure Variables

In the representation (A.3), (A.4) the velocity and pressure variables are still coupled. In order to resolve this difficulty FEATFLOW uses a *discrete projection scheme* similar to *Van Kan's scheme*, cf. Van Kan [62], Shen [100], Prohl [90], which is an operator splitting approach that decouples the computation of $\mathbf{y}$ and $\mathbf{p}$.

Given current iterates $\mathbf{y}_i$, $\mathbf{p}_i$, first an intermediate velocity field, $\tilde{\mathbf{y}}_{i+1}$, is computed where the incompressibility constraint is neglected, i.e.

$$[\mathcal{M} + \theta_1 \delta t N(\tilde{\mathbf{y}}_{i+1})]\tilde{\mathbf{y}}_{i+1} = [\mathcal{M} - \theta_2 \delta t N(\mathbf{y}_i)]\mathbf{y}_i + \theta_3 \delta t \mathcal{F}_{i+1} + \theta_4 \delta t \mathcal{F}_i - \delta t \mathcal{B}\mathbf{p}_i \qquad \text{(A.6)}$$

has to be solved for $\tilde{\mathbf{y}}_{i+1}$.

Since we neglected the incompressibility constraint, the question arises how a correction, $\bar{\mathbf{p}}_{i+1}$, of the pressure variable and thus also a correction, $\bar{\mathbf{y}}_{i+1}$, of the intermediate velocity field should be chosen, in order to account for that deficiency. Obviously, the correction, $\bar{\mathbf{y}}_{i+1}$, should satisfy

$$\mathcal{B}^T(\tilde{\mathbf{y}}_{i+1} + \bar{\mathbf{y}}_{i+1}) = 0. \qquad \text{(A.7)}$$

According to (A.5), FEATFLOW uses the inverse of the lumped mass matrix, $\mathcal{M}_l$, as an approximation to the inverse operator of the right hand side of (A.5) for small step sizes, δt, such that

$$\mathcal{M}_l\, \tilde{\mathbf{y}}_{i+1} \approx [\mathcal{M} - \theta_2 \delta t N(\mathbf{y}_i)]\mathbf{y}_i + \theta_3 \delta t \mathcal{F}_{i+1} + \theta_4 \delta t \mathcal{F}_i - \delta t \mathcal{B}\mathbf{p}_i$$

is assumed. Hence, $\bar{\mathbf{p}}_{i+1}$ has to be chosen such that (A.7) can be approximated by

$$\begin{aligned}
&\mathcal{B}^T[\mathcal{M}_l^{-1}([\mathcal{M} - \theta_2 \delta t N(\mathbf{y}_i)]\mathbf{y}_i + \theta_3 \delta t \mathcal{F}_{i+1} + \theta_4 \delta t \mathcal{F}_i - \delta t \mathcal{B}\mathbf{p}_i) + \bar{\mathbf{y}}_{i+1}] \\
&= \mathcal{B}^T[\mathcal{M}_l^{-1}([\mathcal{M} - \theta_2 \delta t N(\mathbf{y}_i)]\mathbf{y}_i + \theta_3 \delta t \mathcal{F}_{i+1} + \theta_4 \delta t \mathcal{F}_i - \delta t \mathcal{B}(\mathbf{p}_i + \bar{\mathbf{p}}_{i+1}))] \\
&= \mathcal{B}^T(\tilde{\mathbf{y}}_{i+1} - \delta t \mathcal{M}_l^{-1} \mathcal{B}\, \bar{\mathbf{p}}_{i+1}) = 0.
\end{aligned}$$

Consequently, the desired pressure correction, $\bar{\mathbf{p}}_{i+1}$, can be computed by solving the following discrete variant of a *Pressure-Poisson problem*

$$\mathcal{B}^T \mathcal{M}_l^{-1} \mathcal{B}\, \bar{\mathbf{p}}_{i+1} = \tfrac{1}{\delta t} \mathcal{B}^T \tilde{\mathbf{y}}_{i+1}. \qquad \text{(A.8)}$$

The solution of the Pressure-Poisson problem is followed by an update step for the velocity and pressure field to obtain new iterates at the next time level

$$\mathbf{p}_{i+1} = \mathbf{p}_i + \bar{\mathbf{p}}_{i+1}, \qquad \text{(A.9)}$$

$$\mathbf{y}_{i+1} = \tilde{\mathbf{y}}_{i+1} - \delta t \mathcal{M}_l^{-1} \mathcal{B} \bar{\mathbf{p}}_{i+1}. \qquad \text{(A.10)}$$

At this, steps (A.8), (A.9), (A.10) can be seen as a projection of the not necessarily divergence-free velocity field, $\tilde{\mathbf{y}}_{i+1}$, onto a divergence-free subspace such that the new iterate, $\mathbf{y}_{i+1}$, satisfies the (discrete) incompressibility constraint.

A.4 Solution of Nonlinear Equations

Solving (A.6) requires the solution of a nonlinear equation for the velocity field. For this purpose there is an *adaptive fixed point defect correction method* implemented in FEATFLOW. This Newton-like method with additional step size determination serves to solve systems of the type (A.6) and is outlined below.

For given $\mathbf{y}_i$, $\mathbf{p}_i$ we define the residual that depends on the unknown $\mathbf{y}$ as

$$R(\mathbf{y}) = [\mathcal{M} + \theta_1 \delta t N(\mathbf{y})]\,\mathbf{y} - [\mathcal{M} - \theta_2 \delta t N(\mathbf{y}_i)]\mathbf{y}_i - \theta_3 \delta t \mathcal{F}_{i+1} - \theta_4 \delta t \mathcal{F}_i + \delta t \mathcal{B}\mathbf{p}_i \quad \text{(A.11)}$$

such that $R(\tilde{\mathbf{y}}_{i+1})=0$ is equivalent to (A.6). Furthermore, we set $\mathbf{y}^0 = \mathbf{y}_i$.

Consequently, Newton's method for solving $R(\mathbf{y})=0$ requires to solve linear systems of the type

$$R'(\mathbf{y}^j)\,\mathbf{s}^j = -\,R(\mathbf{y}^j) \quad \text{(A.12)}$$

for $\mathbf{s}^j$, with

$$R'(\mathbf{y}^j)\,\mathbf{s}^j = [\mathcal{M} + \theta_1 \delta t N(\mathbf{y}^j)]\,\mathbf{s}^j + \theta_1 \delta t\, \mathcal{C}(\mathbf{s}^j)\,\mathbf{y}^j, \quad \text{(A.13)}$$

for the computation of a complete Newton step. Here, the superscript j indicates the iteration index during the solution of the nonlinear equations. The Newton-like method implemented in FEATFLOW discards the last part of the right hand side of (A.13) and uses the linear system

$$[\mathcal{M} + \theta_1 \delta t N(\mathbf{y}^j)]\,\mathbf{s}^j = -\,R(\mathbf{y}^j) \quad \text{(A.14)}$$

for the computation of a step $\mathbf{s}^j$.

Additionally, an exact step size selection is performed using the merit function

$$m(\omega) = \tfrac{1}{2}\,\|\bar{R}(\mathbf{y}^j, \mathbf{s}^j, \bar{\omega}, \omega)\|^2 \quad \text{(A.15)}$$

with

$$\begin{aligned} \bar{R}(\mathbf{y}^j, \mathbf{s}^j, \bar{\omega}, \omega) \;=\; & [\mathcal{M} + \theta_1 \delta t N(\mathbf{y}^j + \bar{\omega}\,\mathbf{s}^j)]\,(\mathbf{y}^j + \omega\,\mathbf{s}^j) \\ & -[\mathcal{M} - \theta_2 \delta t N(\mathbf{y}_i)]\mathbf{y}_i - \theta_3 \delta t \mathcal{F}_{i+1} - \theta_4 \delta t \mathcal{F}_i + \delta t \mathcal{B}\mathbf{p}_i. \end{aligned} \quad \text{(A.16)}$$

We note that $\bar{R}$ given by (A.16) corresponds to a linearization of the residual (A.11) for a given parameter $\bar{\omega}$, which is usually chosen as $\bar{\omega} = \omega^{j-1}$ if $j > 1$. The step size ω^j that minimizes the merit function in (A.15) is explicitly given by

$$\omega^j = \frac{-\bar{R}(\mathbf{y}^j, \mathbf{s}^j, \bar{\omega}, 0)^T [\mathcal{M} + \theta_1 \delta t N(\mathbf{y}^j + \bar{\omega}\,\mathbf{s}^j)]\,\mathbf{s}^j}{\|[\mathcal{M} + \theta_1 \delta t N(\mathbf{y}^j + \bar{\omega}\,\mathbf{s}^j)]\,\mathbf{s}^j\|^2}. \quad \text{(A.17)}$$

Thus, the next iterate $\mathbf{y}^{j+1}$ can be computed as

$$\mathbf{y}^{j+1} = \mathbf{y}^j + \omega^j\,\mathbf{s}^j. \quad \text{(A.18)}$$

Therefore, FEATFLOW performs the following sequence of steps in order to solve nonlinear equations of the type (A.6). First, for given $\mathbf{y}^j$ equation (A.14) is solved to obtain $\mathbf{s}^j$. Then, a step size according to (A.17) is computed and the iterate $\mathbf{y}^j$ is updated according to (A.18). This is repeated until the residual is sufficiently small or a fixed maximal number of iteration steps have been performed. Then, $\tilde{\mathbf{y}}_{i+1} = \mathbf{y}^{j+1}$ is set and the computations at the next time level can be started.

A.5 Solution of Linear Equations

According to the decoupling strategy for the velocity and pressure variables, and the Newton-like method described in Section A.4, the solution of the discretized Navier-Stokes equations reduces to solving linear systems. These arise either during the solution of nonlinear equations or correspond to the Pressure-Poisson problem that has to be solved. Consequently, we have to solve

$$[\mathcal{M} + \theta_1 \delta t(\nu \mathcal{S} + \mathcal{C}(\mathbf{y}))]\,\mathbf{s} = -R(\mathbf{y}) \tag{A.19}$$

in order to obtain $\mathbf{s}$, or

$$\mathcal{B}^T \mathcal{M}_l^{-1} \mathcal{B}\,\mathbf{p} = \tfrac{1}{\delta t} \mathcal{B}^T \mathbf{y} \tag{A.20}$$

has to be solved for $\mathbf{p}$.

For the solution of the linear systems (A.19), (A.20), FEATFLOW employs multigrid techniques that are adjusted to the special choice of the $\tilde{Q}_1/Q_0$–Element for the finite element discretization of the Navier–Stokes equations.

Bibliography

[1] F. Abergel and R. Temam. On some control problems in fluid mechanics. *Theor. Comput. Fluid Dyn.*, 1(6):303–325, 1990.

[2] R.A. Adams. *Sobolev Spaces.* Academic Press, New York, 1975.

[3] K. Afanasiev and M. Hinze. Adaptive control of a wake flow using proper orthogonal decomposition. Preprint No. 648/1999, Technische Universität Berlin, 1999.

[4] N.M. Alexandrov, J.E. Dennis, R.M. Lewis, and V. Torczon. A trust-region framework for managing the use of approximation models in optimization. *Struct. Optim.*, 15(1):16–23, 1997.

[5] N.M. Alexandrov and M.Y. Hussaini. *Multidisciplinary design optimization - State of the art.* Proceedings of the ICASE/NASA Langley workshop on multidisciplinary design optimization. SIAM, Philadelphia, 1997.

[6] N.M. Alexandrov, R.M. Lewis, C.R. Gumbert, L.L. Green, and P.A. Newman. Optimization with variable-fidelity models applied to wing design. ICASE Report No. 99-49, ICASE, NASA Langley Research Center, Hampton, 1999.

[7] B.G. Allan. A reduced order model of the linearized incompressible Navier-Stokes equations for the sensor/actuator placement problem. ICASE Report No. 2000-19, ICASE, NASA Langley Research Center, Hampton, 2000.

[8] E. Arian, M. Fahl, and E.W. Sachs. Trust-region proper orthogonal decomposition for flow control. ICASE Report No. 2000-25, ICASE, NASA Langley Research Center, Hampton, 2000.

[9] J.A. Atwell and B.B. King. Proper orthogonal decomposition for reduced basis controllers for parabolic equations. ICAM Report 99-01-01, Virginia Polytechnic Institute and State University, Blacksburg, 1999.

[10] J.A. Atwell and B.B. King. Reduced order controllers for spatially distributed systems via proper orthogonal decomposition. ICAM Report 99-07-01, Virginia Polytechnic Institute and State University, Blacksburg, 1999.

[11] C.O. Ball and F.C. Olson. *Sterilization in food technology.* McGraw-Hill, New York, 1957.

[12] H.T. Banks, M.L. Joyner, B. Wincheski, and W.P. Winfree. Evaluation of material integrity using reduced order computational methodology. Techreport CRSC-TR-99-30, North Caroliona State University, Raleigh, 1999.

[13] M. Berggren. Numerical solution of a flow-control problem: Vorticity reduction by dynamic boundary action. *SIAM J. Sci. Comput.*, 19(3):829–860, 1998.

[14] G. Berkooz, P. Holmes, and J.L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annu. Rev. Fluid Mech.*, 25:539–575, 1993.

[15] T.R. Bewley, P. Moin, and R. Temam. DNS-based predictive control of turbulence: An optimal benchmark for feedback algorithms. *submitted to J. Fluid Mech.*, 2000.

[16] A. Booker, J.E. Dennis, P.D. Frank, D.B. Serafini, V. Torczon, and M.W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. ICASE Report 98-47, ICASE, NASA Langley Research Center, Hampton, 1998.

[17] J. Borggaard and J. Burns. Asymptotically consistent gradients in optimal design. In N.M. Alexandrov and M.Y. Hussaini, editors, *Multidisciplinary design optimization - State of the Art. Proceedings of the ICASE/NASA Langley workshop on multidisciplinary design optimization*, pages 303–314, Philadelphia, 1997. SIAM.

[18] J. Borggaard and J. Burns. A PDE sensitivity equation method for optimal aerodynamic design. *J. Comput. Phys.*, 136(2):366–384, 1997.

[19] R.G. Carter. Numerical optimization in Hilbert space using inexact function and gradient evaluations. ICASE Report No. 89-45, ICASE, NASA Langley Research Center, Hampton, 1989.

[20] R.G. Carter. On the global convergence of trust region algorithms using inexact gradient information. *SIAM J. Numer. Anal.*, 28(1):251–265, 1991.

[21] R.G. Carter. Numerical experience with a class of algorithms for nonlinear optimization using inexact function and gradient information. *SIAM J. Sci. Comput.*, 14(2):368–388, 1993.

[22] K.J. Chang, R.T. Haftka, G.L. Giles, and P.-J. Kao. Sensitivity-based scaling for approximating structural response. *J. of Aircraft*, 30:283–288, 1993.

[23] A.R. Conn, N.I.M. Gould, and P.L. Toint. *Trust-region methods.* SIAM, Philadelphia, 2000.

[24] J.K. Cullum and R.A. Willoughby. *Lanczos algorithms for large symmetric eigenvalue computations.* Birkhäuser, Boston, 1985.

[25] J.K. Cullum, R.A. Willoughby, and M. Lake. A Lanczos algorithm for computing singular values and vectors of large matrices. *SIAM J. Sci. Stat. Comput.*, 4:197–215, 1983.

[26] C. Cuvelier, A. Segal, and A.A. van Steenhoven. *Finite element methods and Navier-Stokes equations.* D. Reidel, Dordrecht, 1986.

[27] J.E. Dennis and R.B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Englewood Cliffs, 1983.

[28] J.E. Dennis and V. Torczon. Managing approximation models in optimization. In N.M. Alexandrov and M.Y. Hussaini, editors, *Multidisciplinary design optimization - State of the Art. Proceedings of the ICASE/NASA Langley workshop on multidisciplinary design optimization*, pages 330–347, Philadelphia, 1997. SIAM.

[29] M. Desai and K. Ito. Optimal control of Navier-Stokes equations. *SIAM J. Control Optim.*, 32(5):1428–1446, 1994.

[30] F. Diwoky and S. Volkwein. Nonlinear boundary control for the heat equation utilizing proper orthogonal decomposition. *To appear in Proceedings of the workshop 'Fast solution of discretized optimization problems', WIAS, Berlin*, 2000.

[31] W.S. Edwards, L.S. Tuckerman, R.A. Friesner, and D.C. Sorensen. Krylov methods for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 110(1):82–102, 1994.

[32] M. Gad el Hak, A. Pollard, and J.-P. Bonnet. *Flow control. Fundamentals and practices.* Springer, Berlin, 1998.

[33] M. Fahl. Computation of POD basis functions for fluid flows with Lanczos methods. Forschungsbericht 99-13, Universität Trier, 1999.

[34] M. Fahl, C. Schwarz, and E.W. Sachs. Modeling heat transfer for optimal control problems in food processing. *To appear in Proceedings of the 2000 IEEE CCA/CASD*, 2000.

[35] H.O. Fattorini and S.S. Sritharan. Existence of optimal controls for viscous flow problems. *Proc. R. Soc. Lond.*, Ser. A 439(1905):81–102, 1992.

[36] H.O. Fattorini and S.S. Sritharan. Necessary and sufficient conditions for optimal controls in viscous flow problems. *Proc. R. Soc. Edinb*, Sect. A 124(2):211–251, 1994.

[37] K. Fukunaga. *Introduction to statistical pattern redognition.* Academic Press, Boston, 1990.

[38] A.V. Fursikov, M.D. Gunzburger, and L.S. Hou. Boundary value problems and optimal boundary control for the Navier-Stokes system: The two-dimensional case. *SIAM J. Control Optim.*, 36(3):852–894, 1998.

[39] E. Gallopoulos and Y. Saad. Efficient solution of parabolic equations by Krylov approximation methods. *SIAM J. Sci. Stat. Comput.*, 13(5):1236–1264, 1992.

[40] V. Girault and P.-A. Raviart. *Finite element methods for Navier-Stokes equations.* Springer, Berlin, 1986.

[41] R. Glowinski. *Numerical methods for nonlinear variational problems.* Springer, New York, 1984.

[42] G.H. Golub and C.F. Van Loan. *Matrix computations.* John Hopkins University Press, Baltimore, 1989.

[43] M.D. Gunzburger. *Flow control.* Springer, New York, 1995.

[44] M.D. Gunzburger. A prehistory of flow control and optimization. In M.D. Gunzburger, editor, *Flow control*, pages 185–195, New York, 1995. Springer.

[45] M.D. Gunzburger and S. Manservisi. The velocity tracking problem for Navier-Stokes flows with bounded distributed controls. *SIAM J. Control Optim.*, 37(6):1913–1945, 1999.

[46] P.C. Hansen. *Rank-deficient and discrete ill-posed problems. Numerical aspects of linear inversion.* SIAM, Philadelphia, 1998.

[47] M. Heinkenschloss. Formulation and analysis of a sequential quadratic programming method for the optimal Dirichlet boundary control of Navier-Stokes flow. *Appl. Optim.*, 15:178–203, 1998.

[48] M. Heinkenschloss. A trust-region method for norm constrained problems. *SIAM J. Numer. Anal.*, 35(4):1594–1620, 1998.

[49] N.J. Higham. Matrix nearness problems and applications. In M.J.C. Glover and S. Barnett, editors, *Applications of matrix theory*, pages 1–27, Oxford, 1989. Clarendon Press.

[50] M. Hinze and K. Kunisch. Second order methods for optimal control of time-dependent fluid flow. Bericht Nr. 165, Spezialforschungsbereich F 003 - Kontrolle und Optimierung, Karl-Franzens-Universität Graz, 1999.

[51] M. Hinze and K. Kunisch. Three control methods for time-dependent fluid flow. Preprint No. 660/2000, Technische Universität Berlin, 2000.

[52] P. Holmes, J.L. Lumley, and G. Berkooz. *Turbulence, coherent structures, dynamical systems and symmetry.* Cambridge University Press, Cambridge, 1996.

[53] L.S. Hou and S.S. Ravindran. A penalized Neumann control approach for solving an optimal Dirichlet control problem for the Navier-Stokes equations. *SIAM J. Control Optim.*, 36(5):1795–1814, 1998.

[54] L.S. Hou and S.S. Ravindran. Numerical approximation of optimal flow control problems by a penalty method: Error estimates and numerical results. *SIAM J. Sci. Comput.*, 20(5):1753–1777, 1999.

[55] A. Iollo, S. Lanteri, and J.-A. Désidéri. Stability properties of POD-Galerkin approximations for the compressible Navier-Stokes equations. *Theoret. Comput. Fluid Dynamics*, 13:377–396, 2000.

[56] K. Ito and S.S. Ravindran. A reduced basis method for control problems governed by PDEs. In W. Desch, F. Kappel, and K. Kunisch, editors, *Control and estimation of distributed parameter systems*, pages 153–168, Basel, 1998. Birkhäuser.

[57] K. Ito and S.S. Ravindran. A reduced-order method for simulation and control of fluid flows. *J. Comput. Phys.*, 143(2):403–425, 1998.

[58] K. Ito and J.D. Schroeter. Reduced order feedback synthesis for viscous incompressible flows. Techreport CRSC-TR-98-41, North Carolina State University, Raleigh, 1998.

[59] R.D. Joslin, M.D. Gunzburger, R.A. Nicolaides, G. Erlebacher, and M.Y. Hussaini. Self-contained automated methodology for optimal flow control. *AIAA J.*, 35(5):816–824, 1997.

[60] M.L. Joyner and H.T. Banks. Nondestructive evaluation using a reduced order computational methodology. ICASE Report No. 2000-10, ICASE, NASA Langley Research Center, Hampton, 2000.

[61] P. Justen. *Optimal control of thermally coupled Navier-Stokes equations in food industry.* PhD thesis, Universität Trier, Shaker Verlag, Aachen, 2000.

[62] J. Van Kan. A second-order accurate pressure-correction scheme for viscous incompressible flow. *SIAM J. Sci. Stat. Comput.*, 7(3):870–891, 1986.

[63] C.T. Kelley. *Iterative methods for optimization.* SIAM, Philadelphia, 1999.

[64] C.T. Kelley and E.W. Sachs. Truncated Newton methods for optimization with inaccurate functions and gradients. Techreport CRSC-TR-99-20, North Carolina State University, Raleigh, 1999.

[65] C.T. Kelley and E.W. Sachs. A trust region method for parabolic boundary control problems. *SIAM J. Optim.*, 9(4):1064–1081, 1999.

[66] G.M. Kepler, H.T. Tran, and H.T. Banks. Reduced order model compensator control of species transport in a CVD reactor. Techreport CRSC-TR-99-15, North Carolina State University, Raleigh, 1999.

[67] H.G. Kessler. *Lebensmittel- und Bioverfahrenstechnik - Molkereitechnologie.* Verlag A. Kessler, München, 1996.

[68] M. Kirby. Minimal dynamical systems from PDEs using Sobolev eigenfunctions. *Phys. D*, 57(3/4):466–475, 1992.

[69] D. Kleis. *Augmented Lagrange SQP methods and application to the sterilization of prepackaged food.* PhD thesis, Universität Trier, Shaker Verlag, Aachen, 1997.

[70] D. Kleis and E. W. Sachs. Optimal control of the sterilization of prepackaged food. *SIAM J. Optim.*, 10(4):1180–1195, 2000.

[71] P. Kloucek and F.S. Rys. Stability of the fractional step θ-scheme for the nonstationary Navier-Stokes equations. *SIAM J. Numer. Anal.*, 31(5):1312–1335, 1994.

[72] K. Kunisch and S. Volkwein. Control of Burgers equation by a reduced - order approach using proper orthogonal decomposition. *J. Optimization Theory Appl.*, 102(2):345–371, 1999.

[73] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for parabolic problems. Bericht Nr. 171, Spezialforschungsbereich F 003 - Kontrolle und Optimierung, Karl-Franzens-Universität Graz, 1999.

[74] J. D. Lambert. *Numerical methods for ordinary differential systems.* John Wiley & Sons, Chichester, 1991.

[75] R.M. Lewis and V. Torczon. Pattern search algorithms for bound constrained minimization. *SIAM J. Optim.*, 9(4):1082–1099, 1999.

[76] J.L. Lumley. Coherent structures in turbulence. In R.E. Meyer, editor, *Transition and Turbulence*, pages 215–242, New York, 1981. Academic Press.

[77] H.V. Ly and H.T. Tran. Modeling and control of physical processes using proper orthogonal decomposition. Techreport CRSC-TR-98-37, North Carolina State University, Raleigh, 1998.

[78] H.V. Ly and H.T. Tran. Proper orthogonal decomposition for flow calculations and optimal control in a horizontal CVD reactor. Techreport CRSC-TR-98-13, North Carolina State University, Raleigh, 1998.

[79] M. Moonen and B. De Moor. *SVD and signal processing III. Algorithms, architectures and applications.* Elsevier, Amsterdam, 1995.

[80] J.J. Moré. Recent developments in algorithms and software for trust region methods. In A. Bachem, M. Grötschel, and B. Korte, editors, *Mathematical programming - The state of the art*, pages 258–287, Berlin, 1983. Springer.

[81] S. Müller-Urbaniak. *Eine Analyse des Zwischenschritt–Θ–Verfahrens zur Lösung der instationären Navier–Stokes–Gleichungen.* PhD thesis, Universität Heidelberg, 1993.

[82] J. Nocedal and S.J. Wright. *Numerical optimization.* Springer, New York, 1999.

[83] C.C. Paige. Bidiagonalization of matrices and solution of linear equations. *SIAM J. Numer. Anal.*, 11:197–209, 1974.

[84] H.M. Park and D.H. Cho. Low-dimensional modeling of flow reactors. *Int. J. Heat Mass Transfer*, 39(16):3311–3323, 1996.

[85] H.M. Park and M.W. Lee. An efficient computational method of boundary optimal control problems for the Burgers equation. *Comput. Methods Appl. Mech. Eng.*, 166(3-4):289–308, 1998.

[86] H.M. Park and M.W. Lee. Solution of an inverse heat transfer problem by means of empirical reduction of modes. *Z. Angew. Math. Phys.*, 51:17–38, 2000.

[87] B.N. Parlett. *The symmetric eigenvalue problem.* Prentice–Hall, Englewood Cliffs, 1980.

[88] J. Peterson. The reduced basis method for incompressible viscous flow calculations. *SIAM J. Sci. Stat. Comput.*, 10(4):777–786, 1989.

[89] M.J.D. Powell. Convergence properties of a class of minimization algorithms. In O.L. Mangasarian, R.R. Meyer, and S.M. Robinson, editors, *Nonlinear programming 2*, pages 1–27, New York, 1975. Academic Press.

[90] A. Prohl. *Projection and quasi-compressibility methods for solving the incompressible Navier-Stokes equations.* B.G. Teubner, Stuttgart, 1997.

[91] A. Quarteroni and A. Valli. *Numerical approximation of partial differential equations.* Springer, Berlin, 1994.

[92] M. Rajaee, S.K.F. Karlsson, and L. Sirovich. Low-dimensional description of free-shear-flow coherent structures and their dynamical behaviour. *J. Fluid Mech.*, 258:1–29, 1994.

[93] R. Rannacher. Finite element methods for the incompressible Navier–Stokes equations. Preprint No. 1999-14, IWR Universität Heidelberg, 1999.

[94] R. Rannacher and S. Turek. Simple nonconforming quadrilateral Stokes element. *Numer. Methods Partial Differential Equations*, 8(2):97–111, 1992.

[95] S.S. Ravindran. Proper orthogonal decomposition in optimal control of fluids. Technical memorandum TM-1999-209113, NASA Langley Research Center, Hampton, 1999.

[96] J.F. Rodriguez, J.E. Renaud, and L.T. Watson. Convergence of trust region augmented Lagrangian methods using variable fidelity approximation data. *Struct. Optim.*, 15:141–156, 1998.

[97] Y. Saad. On the rates of convergence of the Lanczos and the block-Lanczos methods. *SIAM J. Numer. Anal.*, 17:687–706, 1980.

[98] Y. Saad. *Numerical methods for large eigenvalue problems.* Manchester University Press, Manchester, 1992.

[99] D.B. Serafini. *A framework for managing models in nonlinear optimization of computationally expensive functions.* PhD thesis, Rice University, Houston, 1998.

[100] J. Shen. On error estimates of the projection methods for the Navier-Stokes equations: Second-order schemes. *Math. Comput.*, 65(215):1039–1065, 1996.

[101] L. Sirovich. Turbulence and the dynamics of coherent structures, Part I: Coherent structures. *Q. Appl. Math.*, 45(3):561–571, 1987.

[102] L. Sirovich, B.W. Knight, and J.D. Rodriguez. Optimal low-dimensional dynamical approximations. *Q. Appl. Math.*, 48(3):535–548, 1990.

[103] D.C. Sorensen. Newton's method with a model trust-region modification. *SIAM J. Numer. Anal.*, 19:409–426, 1982.

[104] S.S. Sritharan. *Optimal control of viscous flows.* SIAM, Philadelphia, 1998.

[105] T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.*, 20:626–637, 1983.

[106] G.W. Stewart and J. Sun. *Matrix perturbation theory.* Academic press, Boston, 1990.

[107] K.Y. Tang, W.R. Graham, and J. Peraire. Active flow control using a reduced order model and optimum control. AIAA 96-1946, 1996.

[108] R. Temam. *Navier-Stokes equations.* North-Holland, Amsterdam, 1979.

[109] R. Temam. *Navier-Stokes equations and nonlinear functional analysis.* SIAM, Philadelphia, 1995.

[110] F. Thomasset. *Implementation of finite element methods for Navier-Stokes equations.* Springer, New York, 1981.

[111] P.L. Toint. Global convergence of a class of trust-region methods for nonconvex minimization in Hilbert space. *IMA J. Numer. Anal.*, 8(2):231–252, 1988.

[112] V. Torczon. On the convergence of pattern search algorithms. *SIAM J. Optim.*, 7(1):1–25, 1997.

[113] S. Turek. *FEATFLOW - Finite element software for the incompressible Navier-Stokes equations: User manual.* University of Heidelberg, 1995.

[114] S. Turek. A comparative study of some time-stepping techniques for the incompressible Navier-Stokes equations: From fully implicit nonlinear schemes to semi-implicit projection methods. *Int. J. Numer. Methods Fluids*, 22(10):987–1011, 1996.

[115] S. Turek. *Efficient solvers for incompressible flow problems. An algorithmic and computational approach.* Springer, Berlin, 1999.

[116] S. Volkwein. Optimal control of a phase-field model using proper orthogonal decomposition. *To appear in Z. Angew. Math. Mech.*

[117] C.-J. Wu and H.-J. Shi. An optimal theory for an expansion of flow quantities to capture the flow structures. *Fluid Dynam. Res.*, 17:67–85, 1995.

Tabellarische Zusammenfassung des Bildungswegs

	Marco Fahl geb. am 21. Nov. 1970 in Nastätten
1977 – 1981	Loreleyschule, St. Goarshausen
1981 – 1990	Wilhelm-Hofmann-Gymnasium, St. Goarshausen
10/1990 – 11/1996	Studium der Wirtschaftsmathematik an der Universität Trier
02/1997 – 02/1998	Erfüllung der allgemeinen Wehrpflicht
03/1998 – 02/2001	Stipendiat im Graduiertenkolleg *Mathematische Optimierung* der Deutschen Forschungsgemeinschaft an der Universität Trier